NVIDIA Cable Validation Tool (CVT) v2.0.1

Table of Contents

1 About This Document ... 21
2 Technical Support ... 22
3 Document Revision History ... 23
4 Release Notes ... 24
4.1 Changes and New Features in This Release ... 24
4.1.1 Unsupported Functionalities/Features ... 24
4.2 Installation Notes ... 24
4.2.1 Supported Fabric Types ... 24
4.2.2 Supported Devices ... 25
4.2.3 Supported Systems and Deployments ... 26
4.2.3.1 Pre-Requisites for Agents ... 26
4.3 Bug Fixes in This Release ... 27
4.4 Known Issues in This Release ... 29
5 Introduction ... 31
5.1 Symptoms and Remediation ... 32
5.2 Features Supported in Different Fabrics ... 34
6 Topology Files ... 36
6.1 P2P File ... 36
6.1.1 Unified Topology Format ... 36
6.1.2 Nodes ... 36
6.1.3 Links ... 38
6.1.3.1 Internal Links ... 39
6.1.4 DC Floor Layout ... 39
6.1.5 Patch Sheet ... 40
6.1.5.1 Column description ... 40
6.1.6 Server Profile ... 41
6.1.7 Legacy P2P Format (will be deprecated) ... 42
6.1.7.1 P2P Examples ... 42
6.1.7.2 Legend Sheet ... 43
6.2 Topo File ... 44
6.3 Dot File ... 44
6.4 Optional Files ... 44
6.4.1 DC Floor Layout File ... 44

6.4.2 HCA Mapping File 45
7 Collector 46
7.1 Deploying the Module 46
7.1.1 Server Resource Requirements per Cluster Size 46
7.1.2 Deploying the Module as Standalone 47
7.1.2.1 Setting Docker Environment 47
7.1.2.1.1 Environment Variable Configuration File 47
7.1.2.1.2 Setting Environmental Variables with Docker Run 51
7.1.3 Deploying the Module as a UFM Enterprise Plugin 52
7.1.3.1 Copy Files to the Plugin 52
7.1.3.2 Overriding the Apache Configuration 52
7.1.3.3 Troubleshooting 52
7.1.3.3.1 Collector hostname resolution and agent reachability 52
7.1.3.3.2 Symptom 52
7.1.3.3.3 Cause 53
7.1.3.3.4 Fix 1 — Add the collector hostname to /etc/hosts (labs without DNS) 53
7.1.3.3.5 Fix 2 — Set the collector IP explicitly in cvt_env.conf (recommended) 53
7.1.3.3.6 Apply and verify 54
7.1.3.3.7 Related settings 54
7.2 CVT as a Service 55
7.2.1 Overview 55
7.2.2 Architecture 55
7.2.3 Service Components 56
7.2.3.1 CVT Service (cvt-service) 56
7.2.3.2 Apache Web Server (apache) 56
7.2.3.3 CV Controller (cv_controller) 56
7.2.3.4 Cron Service (cron) 57
7.2.4 Installation and Deployment 57
7.2.4.1 Docker Deployment for CVT Service 57
7.2.5 Configuration 58
7.2.5.1 Environment Variables 58
7.2.5.1.1 Configuration File (/<cable_bringup_root>/config/cvt_env.conf) 58
7.2.5.1.2 Environment Variable Configuration 58
7.2.5.1.3 Web Service Configuration 58
7.2.5.1.4 UFM Plugin Mode 58
7.2.5.1.5 NetQ Plugin Mode 58
7.2.5.1.6 Supervisord Configuration 58

7.2.6 Service Management ... 59
7.2.6.1 CVT Service Management with Supervisor ... 59
7.2.6.2 CVT Service Commands ... 60
7.2.6.3 Using CVT Controller API ... 60
7.2.6.4 Legacy Mode Operation ... 60
7.2.6.4.1 Stop Service and Run Legacy CLI ... 60
7.2.6.4.2 Legacy CLI Features ... 61
7.2.6.4.3 Common Legacy Mode Workflow ... 61
7.2.6.4.4 When to Use Legacy Mode ... 61
7.2.7 Automatic Features ... 62
7.2.7.1 Topology Loading ... 62
7.2.7.1.1 Load Last Topology ... 62
7.2.7.1.2 Load Specific File ... 62
7.2.7.1.3 Supported Formats ... 63
7.2.7.2 Automatic Validation ... 63
7.2.7.2.1 Signal Handling ... 63
7.2.7.3 Monitoring and Logging ... 63
7.2.7.3.1 Log Files ... 63
7.2.7.3.2 Log Rotation ... 63
7.2.7.3.3 Health Monitoring ... 64
7.2.8 Troubleshooting ... 64
7.2.8.1 Common Issues ... 64
7.2.8.2 Debug Mode ... 65
7.2.8.2.1 Log Analysis ... 65
7.2.9 Best Practices ... 66
7.3 Device Status ... 66
7.3.1 What Is Device Status? ... 66
7.3.1.1 Device Statuses at a Glance ... 67
7.3.1.2 Lifecycle Overview ... 67
7.3.1.2.1 Allowed Transitions Reference ... 68
7.3.1.3 Typical Device Journey ... 69
7.3.1.4 How Status Affects CVT Operations ... 69
7.3.1.5 Managing Device Status in the UI ... 70
7.3.1.5.1 Viewing Device Status ... 70
7.3.1.5.2 Changing the Status of a Single Device ... 70
7.3.1.5.3 Changing the Status of Multiple Devices (Bulk Update) ... 70
7.3.1.5.4 Viewing Status History ... 71
7.3.1.6 Agent Deployment and Device Status ... 71
7.3.1.6.1 Standard Deployment ... 71
7.3.1.6.2 Force Deployment ... 71

7.3.1.7 Topology Upload and the Managed Column ... 71
7.3.1.7.1 Unified PTP Files ... 71
7.3.1.7.2 DOT Files and Legacy ETH PTP ... 72
7.3.1.8 Automatic Transitions ... 72
7.3.1.8.1 Connectivity Checks ... 72
7.3.1.8.2 Agent Deployment ... 72
7.3.1.8.3 Validation Lifecycle ... 72
7.3.1.8.4 Crash Recovery ... 72
7.3.1.9 Migration from Excluded Devices ... 73
7.3.1.10 Common Workflows ... 73
7.3.1.10.1 Bringing a New Device into Production ... 73
7.3.1.10.2 Excluding a Device from CVT ... 73
7.3.1.10.3 Putting a Device into Maintenance ... 73
7.3.1.10.4 Troubleshooting a Production Device ... 73
7.3.1.11 Tips and Best Practices ... 74
7.4 Hierarchical Anomalies ... 74
7.4.1 XDR BER Anomaly Handling ... 76
7.4.1.1 Anomaly Clearing Behavior ... 77
7.5 Bringup CLI ... 77
7.5.1 Running Bringup CLI ... 77
7.5.2 Bringupcli Usage ... 78
7.5.3 Bringupcli Commands ... 78
7.5.3.1 Load Topology Commands ... 78
7.5.3.1.1 load_topo ... 78
7.5.3.1.2 load_ptp ... 78
7.5.3.1.3 load_ip ... 79
7.5.3.1.4 load ... 79
7.5.3.1.5 load_clusters ... 80
7.5.3.2 Validations Commands ... 80
7.5.3.2.1 show_clusters ... 80
7.5.3.2.2 show_switches ... 80
7.5.3.2.3 check_switch_status ... 81
7.5.3.2.4 start_validation ... 81
7.5.3.2.5 stop_validation ... 82
7.5.3.3 Troubleshooting ... 82
7.5.3.4 Set Credentials Commands ... 82
7.5.3.4.1 show_creds ... 82
7.5.3.4.2 set_default_creds ... 83
7.5.3.4.3 set_node_creds ... 83
7.5.3.4.4 remove_node_creds ... 83

7.5.3.4.5 set_credential_profile 83
7.5.3.4.6 remove_credential_profile 84
7.5.3.5 Web User Commands 84
7.5.3.5.1 add_web_login 84
7.5.3.5.2 delete_web_user 84
7.5.3.5.3 show_web_users 84
7.5.3.5.4 update_web_user 85
7.5.3.6 Other Commands 85
7.5.3.6.1 show_switch_history 85
7.5.3.6.2 amber_show_latest 85
7.5.3.6.3 show_incoming_reports 86
7.5.3.6.4 show_statistics 86
7.5.3.6.5 show_switches 86
7.5.3.6.6 add_certificate 86
7.5.3.6.7 version 87
7.5.3.6.8 exit 87
7.5.3.6.9 help 87
7.5.4 Tool Initializing Using Bringup CLI 87
7.5.5 Running bringup GUI 87
7.6 CVT Configuration 87
7.6.1 CVT Environment Configuration (cvt_env.conf) 87
7.6.1.1 Configuring CVT 88
7.6.1.1.1 Precedence 88
7.6.1.1.2 Changing Configuration 88
7.6.1.2 Network Configuration 88
7.6.1.3 Agent Configuration 89
7.6.1.4 Collector Configuration 90
7.6.1.5 SSH Configuration 92
7.6.1.6 Apache Configuration 93
7.6.1.7 Application Configuration 93
7.6.1.8 Data Management 93
7.6.1.9 Database Management 94
7.7 Update Certificate 94
7.8 Apache Configuration for CVT Collector 94
7.8.1 Apache Configuration for CVT Collector 94
7.8.1.1 Overview 94
7.8.1.2 Configuration Options 94
7.8.1.2.1 CVT_APACHE_PROFILE 95
7.8.1.2.2 CVT_APACHE_LOG_LEVEL 95
7.8.1.2.3 CVT_APACHE_ACCESS_LOG 95

7.8.1.3 Profile Details 96
7.8.1.3.1 Small Profile (Default) 96
7.8.1.3.2 Medium Profile 96
7.8.1.3.3 Large Profile 96
7.8.1.3.4 XLarge Profile 97
7.8.1.4 Usage 97
7.8.1.4.1 Via cvt_env.conf (Recommended) 97
7.8.1.4.2 Via Environment Variables 97
7.8.1.4.3 Auto-Detection 97
7.8.1.5 Configuration Files 98
7.8.1.6 Customizing Profiles 98
7.8.1.7 Troubleshooting 98
7.8.1.7.1 Apache fails to start with "already running" 98
7.8.1.7.2 Check current configuration 98
7.8.1.8 Best Practices for Large Clusters 99
7.9 Agent Deployment and SSH Configuration 99
7.9.1 SSH Configuration and Usage in Agent Deployment/Uninstall 99
7.9.1.1 SSH Architecture 99
7.9.1.1.1 System Components 99
7.9.1.1.2 Device Support Matrix 100
7.9.1.2 SSH Configuration 100
7.9.1.2.1 Environment Variables 100
7.9.1.2.2 Key Configuration Details 100
7.9.1.3 Authentication Methods 101
7.9.1.3.1 1. Password Authentication 101
7.9.1.3.2 2. SSH Key Authentication 101
7.9.1.3.3 Authentication Priority (for hosts only) 101
7.9.1.3.4 Switch Authentication Requirements 101
7.9.1.4 Agent Deployment Process 101
7.9.1.4.1 Linux Devices Deployment Flow 101
7.9.1.4.2 Deployment Script Features 102
7.9.1.5 Agent Uninstall Process 102
7.9.1.5.1 Linux Devices Uninstall Flow 102
7.9.1.6 Credential Management 102
7.9.1.6.1 Setting Credentials 102
7.9.1.6.2 Credential Priority 103
7.9.1.7 Troubleshooting 103
7.9.1.7.1 Common SSH Issues 103
7.9.1.7.2 SSH Key Issues 103
7.9.1.7.3 Deployment Script Issues 104
7.9.1.7.4 Debugging Steps 104

7.9.1.8 Best Practices 105
7.9.1.8.1 Security 105
7.9.1.8.2 Performance 105
7.9.1.8.3 Operational 105
7.9.1.9 Container Considerations 105
7.9.1.10 Troubleshooting Agent Deployment Issues on mlnx_os switches 106
7.10 High Availability (HA) Mode Support for CVT Plugin 107
7.10.1 Overview 107
7.10.2 HA Architecture 107
7.10.3 Failover Configuration 107
7.10.4 Required Configuration Settings 108
7.10.5 Complete HA Configuration Example 108
7.10.6 Failover Behavior 108
7.10.7 Important Considerations 109
7.10.8 Summary 109
7.11 Cluster Sizing Guide 109
7.11.1 Overview 109
7.11.2 Sizing Methodology 110
7.11.3 Quick Configuration Reference 110
7.11.3.1 🎯 Simple 3-Variable Configuration 110
7.11.4 Cluster Sizing Table 110
7.11.5 Detailed Recommendations by Scale 112
7.11.5.1 Small Clusters (1-1,000 devices) 112
7.11.5.1.1 Medium Clusters (1,000-10,000 devices) 112
7.11.5.2 Large Clusters (10,000-25,000 devices) 112
7.11.5.3 Hyperscale Clusters (25,000+ devices) 113
7.11.6 Network Bandwidth Analysis 113
7.11.6.1 Bandwidth Requirements by Cluster Size 113
7.11.7 Memory Usage Patterns 114
7.11.7.1 Memory Breakdown by Component 114
7.11.8 Performance Optimization Guidelines 114
7.11.8.1 Quick Performance Tuning 114
7.11.8.2 CPU Optimization 114
7.11.8.2.1 Memory Optimization 114
7.11.8.2.2 Network Optimization 114
7.11.9 Device Type Considerations 115
7.11.9.1 Switches vs Hosts Performance Characteristics 115

7.11.10 Example Configurations 115
7.11.10.1 Small Deployment (<1,000 devices) 115
7.11.10.2 Medium Deployment (1,000-10,000 devices) 115
7.11.10.3 Large Deployment (10,000-30,000 devices) 115
7.11.10.4 Hyperscale Deployment (30,000+ devices) 116
7.11.11 Configuration Guidelines 116
7.11.12 Scaling Considerations 116
7.11.12.1 Vertical Scaling Limits 116
7.11.12.2 Horizontal Scaling Options 116
7.11.13 Device Mix Impact on Sizing 117
7.11.13.1 Typical Cluster Compositions 117
7.11.13.2 Sizing Adjustments by Device Mix 117
7.11.13.3 Monitoring and Alerting 117
7.11.13.3.1 Key Metrics to Monitor 117
7.11.13.3.2 Success Indicators 118
7.11.13.3.3 Scaling Triggers 118
7.11.13.3.4 Performance Expectations by Cluster Size 118
7.11.13.4 Document Relationship 118
7.11.14 Simple Tuning Guide 119
7.11.14.1 Simple CVT Performance Tuning Guide 119
7.11.14.1.1 Easy Configuration - Just 3 Variables 119
7.11.14.1.2 How to Decide Values 119
7.11.14.1.3 Quick Start by Deployment Size: 120
7.11.14.1.4 How to Know If Your Settings Are Good 120
7.11.14.1.5 Expected Performance 121
7.11.14.1.6 Monitoring and Troubleshooting 121
7.11.14.1.7 System Tuning for Large Deployments 122
7.11.14.1.8 Agent Deployment Time Estimates 122
7.11.14.1.9 Bottom Line 123
7.12 CVT as UFM Plugin 123
7.12.1 CVT Issues Integration with UFM Alarms 123
7.12.1.1 Overview 123
7.12.1.1.1 Issues reported to UFM Events 123
7.12.1.1.2 Not reported to UFM Events 123
7.12.1.2 Prerequisites 124
7.12.1.3 3. Configuration file: cvt_env.conf 124
7.12.1.4 Enabling the feature 124
7.12.1.5 [ufm events] variables 124
7.12.1.5.1 Severity override keys (supported syndromes) 125
7.12.1.6 Viewing CVT events in UFM 125

7.12.2 UFM Topology Support 125
7.12.2.1 UFM Topology Support 125
7.12.2.1.1 Feature Summary 125
7.12.2.1.2 Architecture 126
7.12.2.1.3 User Workflows 126
7.12.2.1.4 UI Walkthrough 127
7.12.2.1.5 REST API Quick Reference 127
7.12.2.1.6 Generated PTP File Format 128
7.12.2.1.7 Data Transformation Rules 130
7.12.2.1.8 Configuration 130
7.12.2.1.9 Prerequisites 130
8 Cables Agent 131
8.1 Open Agent Port 131
8.2 Cable Agent Status 131
8.2.1 REST APIs to check agent status 131
8.2.2 Check if Agent Container is running 132
8.2.2.1 On InfiniBand 132
8.2.2.2 On Ethernet, NVOS, and Hosts 132
8.3 Collecting Amber File 133
8.3.1 Ethernet Switches, InfiniBand Switches (NDR, HDR) and Hosts 133
8.3.2 XDR Switches 133
8.3.3 NV Switches 134
9 Cable Validation Controller 136
9.1 Overriding CV Controller Port 136
9.2 CV Controller GUI 137
10 Bringup GUI 139
10.1 Dashboards 140
10.1.1 Operational Dashboard 140
10.1.2 AI Factory Dashboard 140
10.1.3 Operational Dashboard 141
10.1.3.1 Operational Dashboard 141
10.1.3.1.1 Overview 141
10.1.3.1.2 Opening the dashboard 142
10.1.3.1.3 Recommended workflow 142
10.1.3.1.4 Validation toolbar 142
10.1.3.1.5 Operational summary 143
10.1.3.1.6 KPI cards 146

10.1.3.1.7 Issue charts 147
10.1.3.1.8 Device Status 148
10.1.3.1.9 Recent Activity 149
10.1.3.1.10 Understanding common dashboard states 150
10.1.3.1.11 Related page 150
10.1.4 AI Factory Dashboard 151
10.1.4.1 AI Factory Dashboard 151
10.1.4.1.1 Overview 151
10.1.4.1.2 Opening the dashboard 151
10.1.4.1.3 Recommended workflow 151
10.1.4.1.4 Validation toolbar 152
10.1.4.1.5 Build-up Summary 152
10.1.4.1.6 Status KPI cards 153
10.1.4.1.7 Device Status 154
10.1.4.1.8 Device Status Activity 154
10.1.4.1.9 Build-Up Progress 155
10.1.4.1.10 Build-Up Velocity 155
10.1.4.1.11 Understanding Common Dashboard States 156
10.1.4.1.12 Related Pages 156
10.2 Circuits Overview 157
10.2.1 Circuits View 157
10.2.1.1 Rx/Tx Power Lane 160
10.2.1.2 Circuit History and Remediation 161
10.2.1.2.1 Issues and Remediation Workflow 161
10.2.1.2.2 Accessing History and Remediation 161
10.2.1.3 Amber Simulation 165
10.2.2 Flapping Circuits View 167
10.2.2.1 Flapping History 169
10.3 Devices View 169
10.3.1 Agent Status 171
10.3.2 Device Selection and Navigation 171
10.3.2.1 Expanding the Devices Table 171
10.3.2.2 Expanding the Device Details 172
10.3.3 Device Tabs 173
10.3.3.1 Device Overview Tab 173
10.3.3.2 Device Health Tab 173
10.3.3.3 Agent Status Tab 174
10.3.3.4 App Health Tab 175
10.3.3.5 Device Recipe 176
10.3.3.6 How it works 176

10.3.3.7 Where it appears in the UI 176
10.3.3.8 Device statuses 177
10.3.3.9 Important notes 177
10.3.4 Recipe Assignment Row 178
10.4 Rack View 178
10.4.1 Hierarchical Selection Workflow 178
10.4.2 Select a Rack Directly 180
10.4.3 Rack-Level Summary 180
10.4.4 Front View 180
10.4.4.1 Ethernet 181
10.4.4.1.1 SN5600 SN5600D Switch 181
10.4.4.1.2 SN5610 Switch 182
10.4.4.1.3 SMC SuperServer 182
10.4.4.1.4 XE9680 Server 183
10.4.4.1.5 SN2201 Switch 183
10.4.4.1.6 SN2201_M Switch 183
10.4.4.1.7 SN6600 Switch 184
10.4.4.1.8 SN6600-LD Switch 184
10.4.4.2 InfiniBand 184
10.4.4.2.1 MQM8700 Switch 184
10.4.4.2.2 MQM9700 Switch 184
10.4.4.2.3 MQM9701 / QM9701 Switch 185
10.4.4.3 XDR 185
10.4.4.3.1 Quantum-3 XDR Switch 185
10.4.4.4 NVLink 185
10.4.4.4.1 Scaleout Switch 185
10.4.4.4.2 Non-Scaleout Switch 186
10.4.4.4.3 Non-Scaleout DGX Switch 186
10.4.4.4.4 N6100_LD Switch 186
10.4.4.4.5 HGX Compute Tray 186
10.4.4.4.6 DGX Compute Tray 186
10.4.4.4.7 2U GB200 Compute Tray 187
10.4.4.4.8 Dell Server 9712a 187
10.4.4.5 DGX-B300 Server 187
10.4.4.6 Default 188
10.4.4.7 Port Status Coloring 189
10.4.4.8 Status LEDs 190
10.4.4.9 Device Summary 191
10.4.4.9.1 Device Model 191
10.4.5 Front / Rear View Toggle 193

10.4.6 Rear View — Cable Cartridge Visualization ... 193
10.4.6.1 Availability ... 193
10.4.6.2 Cable Cartridge Layout ... 194
10.4.6.3 Cable Cartridge Status Coloring ... 195
10.4.6.4 Cable Cartridge Details (CC-Level Popover) ... 195
10.4.6.5 Slot-Level Details (Slot Popover) ... 196
10.4.6.6 Recommended Actions ... 199
10.4.7 Shuffle Box Visualization ... 200
10.4.8 Rack Guide ... 200
10.4.9 Custom Device Configuration ... 201
10.4.9.1 How It Works ... 201
10.4.9.2 File Location ... 201
10.4.9.3 Configuration Schema ... 201
10.4.9.4 Field Reference ... 202
10.4.9.4.1 Top-Level Fields ... 202
10.4.9.4.2 Device Fields ... 202
10.4.9.4.3 View Configuration Fields (frontView) ... 203
10.4.9.4.4 Port Group Fields ... 203
10.4.9.4.5 LED Group Fields ... 204
10.4.9.5 Example ... 204
10.4.9.6 Validation Rules ... 205
10.5 Tests ... 206
10.5.1 Test Types and Workflows ... 206
10.5.1.1 Golden BER Test ... 206
10.5.1.1.1 Create New Test ... 207
10.5.1.1.2 Test Details ... 208
10.5.1.2 Amber Collection ... 210
10.5.1.2.1 Create New Test ... 211
10.5.1.2.2 Test Details ... 213
10.5.1.2.3 Downloading Amber Files ... 214
10.5.1.3 Advanced Flapping Monitoring ... 214
10.5.1.3.1 Create New Test ... 216
10.5.1.3.2 Test Details ... 218
10.5.2 Managing Tests ... 219
10.5.3 Test Results, Storage, and Retention ... 219
10.6 Reports ... 219
10.6.1 Cables Issues ... 219
10.6.2 Summary Report ... 221
10.6.2.1 Download Circuits as CSV ... 222

10.6.2.2 Go to Circuits 222
10.6.3 Device Health Report 223
10.6.3.1 Topology Levels 224
10.6.3.2 Navigating Deeper 224
10.6.3.3 Location Indicator 226
10.6.3.4 Initial Level — Data Center Level (DC) 226
10.7 System Admin 227
10.7.1 Sidebar navigation 228
10.7.2 Section details 228
10.7.2.1 Setup & Validate 228
10.7.2.2 Agent Management (Reporting) 228
10.7.2.3 Collector Management (Service Management) 229
10.7.2.4 Users Management 229
10.7.3 Setup & Validate 229
10.7.3.1 Step 1 — Load Topology 229
10.7.3.1.1 P2P File 229
10.7.3.1.2 Topo File 233
10.7.3.1.3 Dot File 233
10.7.3.2 Topology Summary 233
10.7.3.3 Step 2 — Credentials 234
10.7.3.4 Step 3 — Pre-Check 236
10.7.3.5 Step 4 — Deploy Agents 238
10.7.3.6 Step 5 — Validate 238
10.7.4 Agent Management 240
10.7.4.1 Summary Report 240
10.7.4.2 Agent Status Report 241
10.7.4.3 Selection-based management 242
10.7.4.3.1 Manage Agents 242
10.7.4.3.2 Manage Device Status 243
10.7.4.3.3 Context menu 244
10.7.4.3.4 Connectivity last-run indicator 244
10.7.4.3.5 Real-time deployment monitoring 245
10.7.4.4 Recipe Management 248
10.7.4.4.1 How it works 248
10.7.4.4.2 Where it appears in the UI 249
10.7.4.5 Device-to-Recipe Mapping Section 249
10.7.4.6 Recipe Files Section 250
10.7.4.7 Recipe and Mapping File Formats 251
10.7.4.7.1 Recipe File 251
10.7.4.7.2 Mapping File 251

10.7.4.8 Device-to-Recipe Mapping Resolution 252
10.7.5 Collector Management 252
10.7.5.1 Service Controls 252
10.7.5.2 Resource Utilization 253
10.7.5.2.1 Collector Logs 254
10.7.6 Users Management 256
10.7.6.1 Upload File 257
10.7.6.2 Account types 258
10.7.6.3 Safeguards 258
10.8 Auto Refresh 259
10.8.1 Enable or Disable Auto-Refresh 259
10.8.2 Adjust the Refresh Interval 259
10.8.3 Manually Refresh the Page 259
10.9 Table 260
10.9.1 Filter 260
10.9.1.1 Supported Filter Types 260
10.9.2 Layout Manager 261
10.9.2.1 Layout Manager 261
10.9.2.1.1 Supported Tables 261
10.9.2.1.2 Layout Controls 262
10.9.2.1.3 Layout Types 262
10.9.2.1.4 Creating a Layout 262
10.9.2.1.5 Applying or Switching Layouts 263
10.9.2.1.6 Updating a Personal Layout 264
10.9.2.1.7 Resetting Changes 265
10.9.2.1.8 Viewing a Layout Summary 266
10.9.2.1.9 Deleting a Personal Layout 266
10.9.2.1.10 Additional Notes 267
10.10 Resource Filter 267
10.10.1 Filter Modes 269
10.10.1.1 Layout (Hierarchical Mode) 269
10.10.1.2 Data Center (DC Mode) 270
10.10.1.3 1.4.1.3. Rack Mode 270
10.10.2 Filter Summary and Results 271
11 Rest APIs 273
11.1 Bringup Server REST API 273
11.2 Overriding Collector Web server Port 273
11.3 Reports APIs 273

11.3.1 Get Validation Report ... 273
11.3.2 Get Validation Status ... 274
11.3.3 Get Topology Metadata ... 274
11.3.4 Get Summary Report ... 275
11.3.5 Get Circuit Report ... 275
11.3.6 Get Circuit Statistics ... 276
11.3.7 Get Device Health Report ... 277
11.4 Resources APIs ... 278
11.4.1 Get Clusters ... 278
11.4.2 Get Data Halls ... 278
11.4.3 Get Scalable Units ... 278
11.4.4 Get All Racks ... 279
11.4.5 Get Rack ... 279
11.4.6 Get Nodes ... 281
11.4.7 Get Resource Utilization ... 282
11.5 Filters APIs ... 282
11.5.1 Get Page Filters ... 282
11.5.2 Get Filter ... 282
11.5.3 Add Filter ... 283
11.5.4 Delete Filter ... 283
11.6 Commands APIs ... 283
11.7 Users Management APIs ... 284
11.7.1 Get All Users ... 284
11.7.2 Get User ... 284
11.7.3 Create User ... 284
11.7.4 Update User ... 285
11.7.5 Delete User ... 285
11.7.6 Get Account Types ... 285
11.8 Files Management APIs ... 285
11.8.1 Get Files ... 285
11.8.2 Upload File ... 286
11.8.3 Delete File ... 286
11.9 Exclude Devices APIs ... 287
11.9.1 Get Excluded Devices ... 287

11.9.2 Exclude Devices 287
11.9.3 Un-exclude Devices 287
11.10 Version APIs 287
11.10.1 Get Version 287
11.11 Agent Status APIs 288
11.11.1 Get Agents Summary 288
11.11.2 Get Agents Details 288
11.12 Golden BER Test APIs 289
11.12.1 Get Tests 289
11.12.2 Get Test Circuits 290
11.12.3 Create Test 290
11.12.4 Stop Test 291
11.12.5 Download The Amber File For A Specific Node 291
11.12.6 Download The Amber Files For The Failed Circuits 291
11.13 Amber Collection Test APIs 292
11.13.1 Get Tests 292
11.13.2 Get Test Details 292
11.13.3 Create Test 292
11.13.4 Download Amber tar File 293
11.14 Advanced Flapping Test APIs 293
11.14.1 Get Tests 293
11.14.2 Get Test Circuits 294
11.14.3 Create Test 294
11.14.4 Stop Advanced Flapping Test 295
11.15 Cable Validation Controller APIs 295
11.15.1 Start Bringup 295
11.15.2 Stop Bringup 295
11.16 Credential APIs 296
11.16.1 Get Devices Credentials 296
11.16.2 Upload Credentials File 296
11.16.3 Description – Uploads and loads credentials from a credential file (JSON or INI format). 296
11.16.4 Set Node Credentials 297
11.16.5 Delete Node Credentials 297
11.16.6 Set Default Credentials 298

11.16.7 Delete Default Credentials ... 298
11.16.8 Set Profile Credentials ... 298
11.16.9 Delete Profile Credentials ... 299
11.17 Help API ... 299
11.18 Amber Simulation APIs ... 300
11.18.1 Get Amber Simulation ... 300
11.18.2 Load Amber Simulation ... 300
11.18.3 Stop Amber Simulation ... 301
11.19 Prometheus Endpoint ... 301
11.19.1 Cable Validation Tool - Prometheus Metrics Endpoint ... 301
11.19.1.1 Overview ... 301
11.19.1.2 Features ... 301
11.19.1.2.1 🎯 Key Capabilities ... 301
11.19.1.2.2 📊 Metrics Categories ... 302
11.19.1.3 API Endpoints ... 302
11.19.1.3.1 Base URL ... 302
11.19.1.3.2 Available Endpoints ... 302
11.19.1.3.3 Authentication ... 302
11.19.1.4 Sample Output ... 302
11.19.1.4.1 Prometheus Format ... 302
11.19.1.4.2 JSON Format ... 303
11.19.1.5 Metrics Reference ... 304
11.19.1.5.1 Power Metrics ... 304
11.19.1.5.2 BER Metrics ... 304
11.19.1.5.3 Temperature Metrics ... 305
11.19.1.5.4 Status Metrics ... 305
11.19.1.5.5 Counter Metrics ... 305
11.19.1.5.6 Validation Metrics ... 305
11.19.1.6 Labels ... 306
11.19.1.6.1 Topology Labels (All Metrics) ... 306
11.19.1.6.2 Cable Labels (Status Metrics Only) ... 307
11.19.1.7 Performance Characteristics ... 307
11.19.1.7.1 Update Frequency ... 307
11.19.1.7.2 Performance Metrics ... 307
11.19.1.7.3 Optimization Features ... 307
11.19.1.8 Troubleshooting ... 307
11.19.1.8.1 Common Issues ... 307
11.19.1.8.2 Performance Tuning ... 309
11.19.1.9 Security Considerations ... 309

11.19.1.9.1 Access Control 309
11.19.1.9.2 Sensitive Data 309
11.19.1.9.3 Recommended Security 309
11.19.1.10 Integration Examples 310
11.19.1.10.1 Python Client Example 310
11.19.1.10.2 Validation Monitoring Examples 310
11.19.1.11 Architecture 311
11.19.1.11.1 Data Flow 311
11.19.1.11.2 Caching Strategy 311
11.19.1.11.3 Performance Optimizations 312
11.19.1.12 Monitoring Best Practices 312
11.19.1.12.1 Prometheus Configuration 312
11.19.1.12.2 Alerting Guidelines 312
11.19.1.13 Version Information 313
11.19.1.13.1 Release Notes 313
11.19.1.13.2 API Stability 314
11.19.1.14 Data Quality Improvements 314
11.19.1.14.1 Enhanced Counter Semantics 314
11.19.1.14.2 Validation Integration Benefits 314
11.19.1.15 Support 314
11.19.1.15.1 Troubleshooting 314
11.19.1.15.2 Performance Monitoring 314
11.19.1.15.3 Contact Information 314
11.20 Logs APIs 315
11.20.1 Introduction 315
11.20.2 API Requests 315
11.20.2.1 Collector Logs API: 315
11.20.2.1.1 Query Parameter 315
11.20.2.1.2 log_type values 315
11.20.2.1.3 Example 316
11.20.2.2 Agent Logs API 316
11.20.2.2.1 Query Parameter 316
11.20.2.2.2 Example 316
11.21 Deprecated REST API Migration Guide 317
11.21.1 1. Deprecated REST API Migration Guide 317
11.21.1.1 1.1. Overview 317
11.21.1.2 1.2. CommandProc Endpoints 317
11.21.1.2.1 1.2.1. 1 Generic Routes (Removed) 317
11.21.1.2.2 1.2.2. 2 Topology Loading 318
11.21.1.2.3 1.2.3. 3 Credential Management 318

11.21.1.2.4 1.2.4. 4 User Management 319
11.21.1.2.5 1.2.5. 5 Agent Deployment 319
11.21.1.2.6 1.2.6. 6 Validation 319
11.21.1.2.7 1.2.7. 7 Switch Status and Connectivity 319
11.21.1.2.8 1.2.8. 8 Commands Without a Replacement 320
11.21.1.2.9 1.2.9. 9 Server Shutdown 320
11.21.1.3 1.3. Device Exclude/Unexclude Endpoints 320
11.21.1.3.1 1.3.1. 1 Endpoint Migration 320
11.21.1.3.2 1.3.2. 2 New Device Status Endpoints 320
11.21.1.4 1.4. Summary 321
11.22 Dashboard REST APIs 321
11.22.1 Get Dashboard Summary 321
11.22.2 Get Port-Issue History 322
11.22.3 Get Build-Up Progress 323
11.22.4 Get Build-Up Velocity 324
11.22.5 Update Build-Up Timestamps 324
11.23 Event REST APIs 325
11.23.1 Get Recent Events 325
11.23.2 Get Event Details 326
11.24 Layouts APIs 327
11.24.1 Get Layouts 327
11.24.2 Create Layout 327
11.24.3 Get Layout 328
11.24.4 Update Layout 329
11.24.5 Delete Layout 329
11.24.5.1 Layout Name Requirements 330
11.24.5.2 Layout Snapshot Fields 330
12 Document Revision History 331

1 About This Document

This document describes NVIDIA Cable Validation Tool (CVT), connectivity and configuration options.

2 Technical Support

Customers who purchased NVIDIA products directly from NVIDIA are invited to contact us through the following methods:

- E-mail: enterprisesupport@nvidia.com
- Enterprise Support page: https://www.nvidia.com/en-us/support/enterprise

Customers who purchased NVIDIA M-1 Global Support Services, please see your contract for details regarding technical support.

Customers who purchased NVIDIA products through an NVIDIA-approved reseller should first seek assistance through their reseller.

3 Document Revision History

For the list of changes made to this document, refer to Document Revision History.

4 Release Notes

4.1 Changes and New Features in This Release

Feature	Description
NVL576 Solution Support	Added support for NVL576 systems, including compatibility and setup requirements.
SN2201 Copper-Interface Support	Enabled validation of copper interfaces on SN2201 switches and documents expected behavior.
VERA RUBIN Shuffle-Cable and Shuffle-Box Support	Added support for VERA RUBIN PTP patch-sheet structures, shuffle-cable paths, and shuffle-box rack assets.
On-demand Per-Circuit Telemetry Refresh	Enabled users to refresh telemetry for an individual circuit and view the updated circuit data.
IPv6 Support	Added support for IPv6 addressing in PTP configuration, agent deployment, and CVT validation.
Unified PTP Patch Sheet	Added the Patch sheet to the unified PTP template for defining patching information.
Dashboards	Added Operational and AI Factory dashboards for monitoring cable-validation operations, infrastructure health, AI Factory build-up progress, and production readiness.
Layout Manager	Lets users create and reuse named table layouts that preserve column width, order, visibility, pinning, and sorting in supported report tables.
Recipe Management	Added support for validating devices against a file containing the expected firmware and software versions.
NVOS Analysis on XDR Switch	Adds support for using NVOS command outputs, instead of the amber file, to analyze port status.
Identify Master NVSwitch within the rack	Added the ability to collect cluster health applications from every NVSwitch agent and reports any degradations in the Rack View.
Template download for device credentials and users	Added a download option for templates used to add management users and device credentials.

4.1.1 Unsupported Functionalities/Features

1. Commands REST APIs has been deprecated; see REST API migration section.
2. In the upcoming versions of the Cable Validation Tool, the Legacy Ethernet and InfiniBand P2P file formats will be deprecated. Moving forward, only the Unified Topology will be supported. For more information, refer to P2P File.

4.2 Installation Notes

4.2.1 Supported Fabric Types

The Cable Validation Tool is compatible with the following fabric types:

Fabric Type	Description
InfiniBand	Supported on MLNX-OS Switches with NDR and HDR port speeds
XDR	Supported on NVOS Q3400-RA Switches with XDR port speed
Ethernet	Supported on Cumulus Switches and Ethernet Hosts
NVLink	• Supported on NVOS Switches and compute nodes of GB200/GB300 rack types: 36x2, 36x1, and 72x1. • Supported on internal NVLink links in the rack • Hosts with external links as Ethernet or IB • The external InfiniBand links will report BER status, but neighboring information will need to be monitored from the neighboring switch side only.

4.2.2 Supported Devices

Type	Model
XDR Switches	• Q3400_RA
NDR Switches	• MQM9700
HDR Switches	• MQM8700
Ethernet Switches	• SN5600 • SN5610 • SN2201 - support also for copper interfaces
NVLink Switches	• N5200_LD • N5201_LD • N5110_LD • N5112_LD • N5113_LD • N5400_LD • N5500_LD • N5700_LD
Ethernet GPU Servers	• SYS-421GE-TNHR2-LC-TW008 (H100) • PowerEdge XE9680
NVLink GPU Servers	• GB200 NVL • DGX GB200 Compute Tray • Dell Server 9712a • DGX B300
Sonic Switches	• SN3700 • SN3700C • SN4600 • SN4600C • SN4700 • SN5400 • SN5600

4.2.3 Supported Systems and Deployments

The Cable Validation Tool supports the following system and rack deployments:

Deployment	Supported CVT scope
GB200	Supports GB200 NVLink rack configurations, including 36x1, 36x2, and 72x1 rack types. Includes internal NVLink validation and external Ethernet or InfiniBand connectivity.
GB300	Supports GB300 NVLink rack configurations, including supported rack-scale NVLink topology and external Ethernet or InfiniBand connectivity.
NVL576	Supports NVL576 solution deployments, including topology loading, device discovery, connectivity validation, and reporting.
Vera Rubin	Supports Vera Rubin deployments with shuffle-cable and shuffle-box topologies, including the associated unified topology and patch information.

> ***Deployment support depends on the corresponding switch, server, operating-system, and topology-file requirements documented in this guide. Refer to Supported Devices, P2P File, and Pre-Requisites for Agents for detailed configuration requirements.***

⚠ Devices not included in this list will be supported by CVT but may not be displayed correctly in the rack view.

4.2.3.1 Pre-Requisites for Agents

Device Type	Requirement	Details
All	Container Runtime	Docker installed and running on Server acting as collector and the Switches/Hosts being monitored.
	Network Access	Port 8251 open for communication between the collector and agents(switches and hosts being monitored).
	SSH Access	SSH access required for initial agent deployment on switches and hosts.
Servers	Architecture	x86_64 or aarch64
	Packages	Servers should have `mft` and `ofed` packages installed
Ethernet Servers	Packages	Server should have `lldpd` installed and `/var/run/lldpd.socket`
GB200 compute Nodes	Packages	Should have `nvidia-container-toolkit` and `nvidia-smi` packages installed
SONiC Switches	LLDP Container	SONiC LLDP container running (`docker ps | grep lldp`)
	SONiC Version	SONiC with `config_db.json` support

SONiC-Based Switches Support

The Cable Validation Tool (CVT) provides comprehensive support for SONiC (Software for Open Networking in the Cloud) based switches. CVT continuously monitors cable connections on SONiC switches, validates them against expected topology, and reports any discrepancies such as missing cables, wrong connections, or signal quality issues.

The CVT agent installer automatically detects SONiC switches by checking for the SONiC version file. No manual configuration of the agent type is required.

4.3 Bug Fixes in This Release

Ref #	Bug Fix
5069921	**Description**: CVT Collector Device Status is not updated after users Stops Validation on the Collector or if the agent was suddenly removed not by the collector
	Keywords: Device Status
	Discovered in Release: v1.9.1
5079289	**Description**: host was offline for a while but it came back. The CVT Collector OS can ping the host but CVT Collector Agent Status PING status = FALSE
	Keywords: Device Status
	Discovered in Release: v1.9.1
5079009	**Description**: Can't set device status to Maintenance or RMA.
	Keywords: Device Status
	Discovered in Release: v1.9.1
5071249	**Description**: devices with mismatch node name status stuck on validation_ready status while is should be not_provisioned
	Keywords: Device Status
	Discovered in Release: v1.9.1
5056427	**Description**: Incorrect Cable Type reported in CVT
	Keywords: Cable Type - Copper/Optical
	Discovered in Release: v1.9.1
5032833	**Description**: CVT reports wrong port if the port name case is different
	Keywords: Case mismatch for port name
	Discovered in Release: v1.9.1
5040592	**Description**: Support gb200 from other vendors
	Keywords: GB200 Product Name
	Discovered in Release: v1.9.1
5032830	**Description**: CVT does not report Signal & BER values for NDR switch with split profile
	Keywords: NDR split port
	Discovered in Release: v1.9.1

Ref #	Bug Fix
5004923	**Description**: RX/TX power loss logic - Account for threshold breach before variance
	Keywords: RX/TX power loss
	Discovered in Release: v1.8.2
5038390	**Description**: Log rotation at midnight causes UI to hang for hours
	Keywords: Log rotation
	Discovered in Release: v1.9.1
5083896	**Description**: False positive issues with uplink ports of sn2201 switches
	Keywords: sn2201
	Discovered in Release: v1.9.1
5038392	**Description**: Device Status shows Validating but the device does not have an agent installed.
	Keywords: device status
	Discovered in Release: v1.9.1
4989873	**Description**: start validation failed silently when running start validation without running pre-check
	Keywords: pre-check
	Discovered in Release: v1.9.1
5079240	**Description**: SPC6 Salamandra - All switch ports appeared down after running RDMA traffic.
	Keywords: spectrum6
	Discovered in Release: v1.9.1
5097885	**Description**: Time since last clear counter - sorting does not work properly.
	Keywords: sort and filter
	Discovered in Release: v1.9.1
5043909	**Description**: `RX_TX_POWER_MISMATCH` syndrome appear in the Reports Summary, not in the Circuits View.
	Keywords: reports, rx tx power mismatch
	Discovered in Release: v1.9.1
4957097	**Description**: Valid Duplicate Rack names are not recognized correctly CVT
	Keywords: rack name duplication
	Discovered in Release: v1.9.1
5179837	**Description**: file descriptor leak for agent logs
	Keywords: agent logs
	Discovered in Release: v1.9.1
5102756	**Description**: Report LLDP missing as miswiring / wrong neighbor (should show LLDP not working)
	Keywords: LLDP
	Discovered in Release: v1.9.1

Ref #	Bug Fix
5165458	**Description**: racks endpoint returns empty list for PTP without Data Hall and SU
	Keywords: rack view
	Discovered in Release: v1.9.1
5170410	**Description**: Unified topology load slow due to repeated HCA mapping warnings
	Keywords: unified ptp template
	Discovered in Release: v1.9.1
5165642	**Description**: Go To Circuit from Reports View does not work
	Keywords: report view, circuit view
	Discovered in Release: v1.9.1
5194267	**Description**: Device View: Add ability to expand devices table left and right
	Keywords: device view
	Discovered in Release: v1.9.1
5182931	**Description**: XDR: admin down and flapping links missing
	Keywords: flapping links, admin down
	Discovered in Release: v1.9.1
5183907	**Description**: Maintenance state does not fully exclude devices from CVT operations
	Keywords: Maintenance state
	Discovered in Release: v1.9.1
5186903	**Description**: IB should not throw LLDP related warning
	Keywords: neighbor, lldp, smpquery
	Discovered in Release: v1.9.1
5216695	**Description**: amber collection for sonic is dropped in case cable agent deployed before mst service started.
	Keywords: sonic switch
	Discovered in Release: v1.9.1
5200163	**Description**: UI pre-check fails with "No pre-check output available" due to async race
	Keywords: pre-check
	Discovered in Release: v1.9.1

4.4 Known Issues in This Release

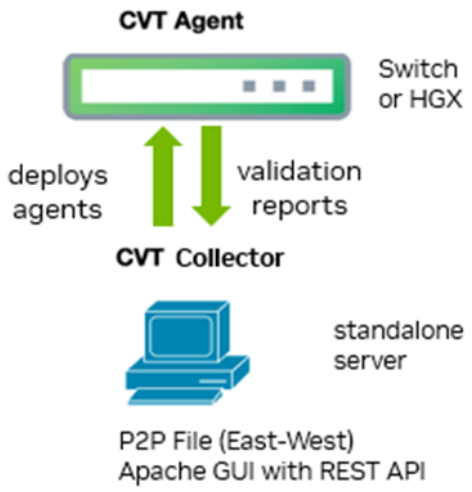
Reference Number	Issue
4877768	**Description**: In Circuit View, node names are automatically converted to lowercase.
	Keywords: PTP, Hostname, Lower Case
	Discovered in Release: v1.8.1
	Workaround: N/A

Reference Number	Issue
4995060	**Description:** [UFM] Cables connected to unmanaged switches are lost from CVT validation when loading UFM topology
	Keywords: UFM, Topology, Unmanaged Switches
	Discovered in Release: v1.9.1
	Workaround: N/A
4994136	**Description:** [UFM] hosts without hostname in topology file are missing from PTP
	Keywords: UFM, Topology, Hostname
	Discovered in Release: v1.9.1
	Workaround: N/A
5190302	**Description:** XDR: Transceiver reinsert/swap count always reports 0
	Keywords: Transceiver reinsert swap counter, circuit view
	Discovered in Release: v2.0.1
	Workaround: N/A

5 Introduction

The Cable Validation Tool (CVT) is designed to ensure the accuracy and quality of network cluster wiring. Its primary purpose is to validate the connectivity of physical links within the cluster and verify high-quality communication between the network components. By maintaining the integrity of these connections.
The Collector, also referred to as the "bring-up server," serves as the central component of the system and operates as a Docker container. It can be deployed on any machine connected to the management network of the switches. This deployment enables seamless communication with the switches. For large-scale systems, the Collector relies on dedicated agents installed on each switch. These agents are responsible for verifying the connections between the switches.

The Collector can be deployed in environments using IPv4 or IPv6 management-network connectivity, provided it can reach the nodes required for validation.



Collector Responsibilities
The Collector performs the following critical tasks:

- Deployment and Execution: It is installed and executed on a server with network access to all nodes requiring validation.
- Topology Validation: Reads the **Topology Files** (P2P, topo or dot), which serves as the authoritative source for validating the physical link connections in the fabric topology.
- Agent Management:
    - Deploys agents on all nodes and uninstalls them
    - Monitors agents' health
    - Supports external (unmanaged) agent deployments, with the Collector only monitoring their health
- Data Collection and Processing: Collects and processes agents' reports from all validated nodes
- User Interface and Reporting:
    - Displays validation results through a web page, along with recommended remediation steps.

    - Provides data visualization options, including aggregation, sorting, and filtering, and supports downloading reports in CSV format. REST APIs are also available for integration with other systems.
- IPv6 Support:
    - Supports IPv6 addressing for Collector-to-node communication, agent deployment, topology configuration, and validation.

Agent Responsibilities
Agents are installed on all switches and servers within the cluster. Their key functions include:

- deployed on all Switches and Servers
- Real-Time Monitoring:
    - Monitor node and link statuses every 10 seconds. Agents detect changes in link states and, when a change occurs, send an updated status report to the Collector.
    - If no changes are detected, agents send a periodic status report every 10 minutes.
    - Amberfile collection takes place upon state changes, which can take 40-50s.
- Event-Driven Reporting:
    - Upon receiving a "start_validation" message from the Collector, agents initiate status reporting.
    - Reports are triggered in two scenarios:
        - When a link status change is detected (ad-hoc report).
        - Every 10 minutes as part of routine reporting.

Further details on the Collector and Cable Agents, including their operational workflows, will be discussed in the subsequent chapters.

5.1 Symptoms and Remediation

The CVT tool supports the following symptoms/issues:

- Validate *Physical* Connections (Cable, End points) / Miswiring
    - Wrong-neighbor - the cable connects to a different device than Topo file (P2P, topo or dot) dictates
    - Wrong-port - the cable connects to the expected device but on the wrong port
    - Unknown-neighbor - the cable connects to a device not mentioned in the Topo file (P2P, topo or dot) or LLDP is not enabled/failing
    - Extra Cable - a cable was found to be connected but not part of the loaded Topo file (P2P, topo or dot)
    - Media Unplugged - a cable is missing in the port. A transceiver is not present in the port if optical cable is used.
    - NIC name-mismatch - The interface name of hosts in the ptp file does not match any actual interface names on the host
    - RX/TX power mismatch - The power on the Transmitter side and Receiver side does not match. Alerted when the power difference is greater than 3dBm
- Validate *Layer1 Link* Integrity (Bit Error Rate, Lane powers, Temperature)
- Flapping link - the link state has transitioned up->down->up on its own or due to external actions.
- Link Down, No Signal - fiber is not connected or broken

- ErrDisable-Rx - interface down events due to the Server NIC firmware bug issues (RX Disable)
- Err-Disable-Flap - Link Protection feature (5 flaps/10 sec.) due to excessive flaps
- Anomalous-Port - out-of-range parameters such as transceiver lane signal strength or transceiver temperature
- Underperforming (BER) Bit-Error-Rate
    - Effective BER errors should be 0 during the first 125 mins of the link being up
    - Raw BER should be ≤ 1e-6
    - Effective BER should be ≤ 1.5E-254 for ≤ 6hrs measurements and ≤ 1E-15 for ≥ 6hrs measurements
- Triage *Non-cable issues (Provisioning, CVT issues) requiring Escalations*
    - AdminDown - spectrum switch port is administratively shutdown
    - Negotiation Fail - detects interface issue due to speed and duplex mismatch between devices
    - No-report - agent communication is working but report not received
    - unreachable-device - agent not installed, not running, not reachable (e.g. port 8251 not open in switch configuration)

Syndrome	Description	non-admin users	Supported Fabric
Negotiation Fail	Negotiation of speed, fec or config issue	YES	IB, ETH, XDR, NVlink
AdminDown	Link is disabled administratively	YES	IB, ETH, XDR, NVlink
Wrong-neighbor	Port is connected to the wrong peer node	YES	IB, ETH, XDR
Wrong-port	Port is connected to the wrong port in the correct peer node	YES	IB, ETH, XDR
Extra-cable	port is connected but neighbor is not in the P2P topology	YES	IB, ETH, XDR
Flapping-link	On switches: Carrier transitions are monitored every 10 seconds. If it increments by more than 2 in 125 sec interval, a link flap alarm is raised	YES	IB, ETH, XDR, NVlink
Underperforming-link	High BER counters	YES	IB, ETH, XDR, NVlink
Anomalous-port (Signal, Temperature)	Some counters are not in range	YES	IB, ETH, XDR, NVlink
Unreachable-device	Cannot ping it and/or Agent not deployed or Agent communication is failing	NO	IB, ETH, XDR, NVlink
Media Unplugged	Port is down and no cable is plugged in or transceiver is not plugged in if its a optical cable	YES	IB, ETH, XDR
NIC Name-mismatch	Interface name of hosts in the ptp file does not match any actual interface names on the host	YES	ETH

Syndrome	Description	non-admin users	Supported Fabric
Unknown-neighbor	Port is up, however no peer info found one known instance is when the far end is not reachable	NO	IB, ETH, XDR
Link Down, No signal	Port is down, **while transceivers are plugged in**	YES	IB, ETH, XDR, NVlink
ErrDisable – Flap, Proto Down	Cumulus switch Port locally disabled by Link Protection (≥5 flaps/10s), defensive mechanism enabled by default.	YES	IB, ETH, XDR, NVlink
ErrDisable -Rx	Interface down events due to the Server NIC firmware bug issues (RX Disable)	YES	IB, ETH, XDR, NVlink
RX/TX power mismatch	The difference in power on the Transmitter and Receiver side is **greater than 3dBm**	YES	IB, ETH, XDR, NVlink
no report	Node is reachable but no agent report was received yet	NO	IB, ETH, XDR, NVlink

5.2 Features Supported in Different Fabrics

Feature	Description	Supported Fabric
Circuit View	Shows the links being monitored	IB, ETH, XDR, NVlink
Port Status	Shows the port is up or down	IB, ETH, XDR, NVlink
Link Syndrome	Shows the cable/port issues for the link	IB, ETH, XDR, NVlink
BER Stats	Shows the eff BER, raw BER, and grading based on BER	IB, ETH, XDR, NVlink
Report Anomalies	Shows the anomalies detected	IB, ETH, XDR, NVlink
Flapping Status	Shows the advanced flapping stats	IB, ETH, XDR, NVlink
Rack View	Shows the switches and hosts on the rack	IB, ETH, XDR, NVlink
System Admin	Allows to start the bringup service, load topology, set credentials and start validation. Shows overview of the validation session and reporting status of agents. Allows to manage GUI users and displays collector resource utilization.	IB, ETH, XDR, NVlink
Golden BER Test	Creates a test to monitor Bit Error Rates (BER) and analyze the interface counters	IB, ETH, XDR, NVlink
Amber Collection Test	Allows to collect amber files on demand	IB, ETH, XDR, NVlink
Advanced Flapping Test	Creates a flapping test to analyze the metrics that could lead to a flapping event	IB, ETH, XDR, NVlink

Feature	Description	Supported Fabric
SN2201 copper-interface support	Validates copper interfaces on SN2201 switches and reports their connectivity and validation status.	ETH
VERA RUBIN shuffle topology	Supports VERA RUBIN shuffle-cable paths, shuffle-box assets, and associated patch-sheet topology.	IB, ETH, XDR, NVlink
On-demand per-circuit telemetry refresh	Allows users to request an individual circuit telemetry refresh and view the latest available circuit data.	IB, ETH, XDR, NVlink

6 Topology Files

Topology files are the primary reference for validating physical link connections within a fabric topology. These files must be provided by the user, as the validation process cannot start without them.

The tool supports the following types of topology files:

- P2P File
- Topo File
- Dot File
- Optional Files

6.1 P2P File

6.1.1 Unified Topology Format

The Unified Topology workbook defines the physical and logical topology of the CVT environment. The template unified_ptp_template_2.0.1.xlsx supports the following sheets:

- Nodes — Defines switches, hosts, and other network devices, including node type, OS, model, rack placement, management state, credentials, and IP address.
- Links — Defines connections between nodes, including protocol, circuit ID, plane, rail, ports, shuffle IDs, connector information, cable part numbers, and source references.
- Patch — Defines physical cable segments for circuits that pass through passive cabling infrastructure. Use Circuit-ID and Segment-Index to associate and order segments. Termination
  fields describe the endpoints, data hall, scalable unit, rack, rack unit, model, blade/cassette, side, port, and connector type.
- DC Floor Layout — Defines data halls, scalable units, racks, rack types, rack groups, and rack indexes. This is required in some NVLink rack configurations.
- Server Profile — Maps custom NIC names to OS interface names, physical ports, RDMA names, and PCI addresses. This is required for applicable host and InfiniBand configurations.

Column names are case-insensitive. The order of columns does not affect processing. Requirement levels are defined as Mandatory, Mandatory in Some Cases, Recommended, and Optional.

6.1.2 Nodes

The Nodes sheet will focus on providing comprehensive information about the nodes within the cluster. The primary data points will include:

FabricID	Rack	Unit	TrayIndex	NodeName	NodeType	NodeOS	NodeModel	ServerProfile	Managed	CredentialProfile	IP

Column	Value Type	Description	Mandatory
`NodeName`	String	The identifier for the node within the network.	☒
`NodeType`	[Switch|Host]	The classification of the node (e.g., host, switch).	☒
`NodeModel`	String	The specific model of the node. **Mandatory for GB200/300**	☐
`NodeOS`	String	The OS running on the node. Supported OS are: mlnx-os, cumulus, nvos, and linux for hosts	☒
`Rack`	String	The physical rack where the node is located.	☐
`Unit`	Integer	The specific unit within the rack.	☐
`TrayIndex`	Integer	The tray index within the rack. Relevant and **mandatory for GB200/300**	☐
`CoolingType`	[Air|Liquid]	The node cooling type (e.g., Air, Liquid). For future use (not supported now)	☐
`FabricID`	String	The Site/Cluster/Fabric name/ID In case the topology includes multiple sites	☐
`ServerProfile`	String	Hosts are assigned server profiles to provide mapping between different interface naming conventions. It is a custom name you can create which will be referenced in the Server Profile Sheet.	☐
`Managed`	[yes|no]	Control if agent should be installed on this node. Set to 'yes' by default.	☒
`CredentialProfile`	String	A custom name assigned to a set of nodes with same credentials which is different from the default.	☐
`IP`	String	the ip address for the node	☐

Notes:

- If Rack/Unit information is missing, some features—specifically the Rack View and filtering capabilities—will not function in the CVT interface.
- IB hosts need to be added to the nodes sheet, so that their corresponding links to switches can be detected. However, agent installation on them is not supported, hence their 'Managed' column value should be 'no'.
- Column names are case-insensitive, and the order of columns does not affect functionality.
- The user can dismiss optional columns as described in the Mandatory column.

6.1.3 Links

The Links sheet will detail the connectivity information across the nodes within the cluster. It will consist of the following columns:

Protocol	Circuit-ID	Shuffle-ID	Plane	Rail	A-Node	A-Port	A-Module-PN	A-Connector	A-MPO-Connector	Z-Node	Z-Port	Z-Module-PN	Z-Connector	Z-MPO-Connector	Cable-PN	Cable-Connector	Source-Ref

Column	Value Type	Description	Mandatory
A-Node	String	The source node for the link. Must exist in the Nodes sheet	☒
A-Port	String/Integer	The port on the source node.	☒
Z-Node	String	The destination node for the link. Must exist in the Nodes sheet	☒
Z-Port	String	The port on the destination node.	☒
Protocol	(ib/ethernet/nvlink)	The protocol used for the connection	☒
Circuit-ID	String/Integer	Unique identifier for the circuit or end-to-end connection. Used to associate link and patch segments.	☐
Plane	String/Integer	Identifies the logical or physical plane to which the connection belongs.	☐
Rail	String/Integer	Identifies the network rail or lane associated with the connection.	☐
Cable-PN	String	Cable part number identifying the cable type or assembly used.	☐
Cable-Connector	String	Identifies the connector type used by the cable.	☐
Shuffle-ID	String/Integer	Shuffle Cable ID	☐
A-Connector	String/Integer	A side Connector of the shuffle cable	☐
A-MPO-Connector	String/Integer	A side MPO Connector of the shuffle cable	☐
Z-Connector	String/Integer	Z side Connector of the shuffle cable	☐
Z-MPO-Connector	String/Integer	Z side MPO Connector of the shuffle cable	☐

Column	Value Type	Description	Mandatory
A-Module-PN	String/Integer	A side module part number	☐
Z-Module-PN	String/Integer	Z side module part number	☐
Source-Ref	String/Integer	Line number from a original reference file from which this PTP is constructed	☐

In case of Host/Server ports, the A/Z-port would be the custom NIC name if available or the default NIC name. This value can be found in the output of `ip address show` command.
For IB hosts it is mandatory to have a server profile for this interface so that there is a mapping available for corresponding RDMA name of the port. RDMA name can be found from output of command: `sudo mst status -v`.

Notes:

- Column names are case-insensitive, and the order of columns does not affect functionality.
- The user can dismiss Optional Columns as explained in the Mandatory column.

6.1.3.1 Internal Links

Internal links (NVLink within the GB200/300 racks) will not be part of the Links sheet. CVT detects any internal links based on the RackType and uses the predefined JSON representation of these links to build the links. This approach ensures seamless integration and efficient configuration within the GB200/300 racks.

6.1.4 DC Floor Layout

The data center floor layout sheet is an optional sheet that is useful to provide a layout of the data center. This includes information about:

- Data Halls: The various halls within the data center.
- Scalable Units: Units designed to be scalable for future expansions. Add a default SU name for all if you are not using scalable unit concepts.
- Racks: Detailed information about the racks within the data center. This information is mandatory for GB200/300 racks.
- - Rack: Rack Name
- - Rack Type: Type of the rack. Can be GB200_72x1, GB200_36x2, or GB200_36x1 for GB200 racks (similar convention is used for GB300) or any other general rack type like ServerRack, NetworkRack for other general racks.
- - Rack Group: Used for grouping the racks, especially in GB200 racks. It is a unique number for each GB200 rack system. A GB200 72x1 system would have a unique group number for itself. In a GB200 36x2 system, the two racks will belong to the

same group_number. (Group number is just a made-up integer to facilitate the identification of rack systems correctly).

DataHall	ScalableUnit	Rack	RackType	Rack Group

6.1.5 Patch Sheet

The optional Patch sheet defines physical cable segments for circuits that pass through passive cabling infrastructure. Use Circuit-ID and Segment-Index to associate each segment with a circuit and preserve its order.

Circuit-ID	Segment-Index	Cable-ID	Cable-PN	Termination-1-Type	Termination-1-Name	Termination-1-DataHall	Termination-1-ScalableUnit	Termination-1-Rack	Termination-1-Rack-Unit	Termination-1-Model	Termination-1-Blade	Termination-1-Cassette	Termination-1-Side	Termination-1-Port	Termination-1-Connector-Type	Termination-2-Type	Termination-2-Name	Termination-2-DataHall	Termination-2-ScalableUnit	Termination-2-Rack	Termination-2-Rack-Unit	Termination-2-Model	Termination-2-Blade	Termination-2-Cassette	Termination-2-Side	Termination-2-Port	Termination-2-Connector-Type	Source-Ref

6.1.5.1 Column description

Column	Description	Mandatory
Circuit-ID	Identifies the circuit associated with the patch segment.	☐
Segment-Index	Defines the segment's order within the circuit path.	☐
Cable-ID	Unique identifier for the physical cable.	☐
Cable-PN	Part number identifying the cable type or assembly.	☐
Termination-1-Type	Type of the first termination, such as rack, blade, cassette, or port.	☐
Termination-1-Name	Name or identifier of the first termination.	☐
Termination-1-DataHall	Data hall containing the first termination.	☐
Termination-1-ScalableUnit	Scalable unit containing the first termination.	☐
Termination-1-Rack	Rack containing the first termination.	☐
Termination-1-Rack-Unit	Rack unit location of the first termination.	☐
Termination-1-Model	Hardware model of the first termination.	☐
Termination-1-Blade	Blade identifier for the first termination, when applicable.	☐

Column	Description	Mandatory
Termination-1-Cassette	Cassette identifier for the first termination, when applicable.	☐
Termination-1-Side	Side of the first termination, such as A or Z.	☐
Termination-1-Port	Port used by the first termination.	☐
Termination-1-Connector-Type	Connector type used by the first termination.	☐
Termination-2-Type	Type of the second termination.	☐
Termination-2-Name	Name or identifier of the second termination.	☐
Termination-2-DataHall	Data hall containing the second termination.	☐
Termination-2-ScalableUnit	Scalable unit containing the second termination.	☐
Termination-2-Rack	Rack containing the second termination.	☐
Termination-2-Rack-Unit	Rack unit location of the second termination.	☐
Termination-2-Model	Hardware model of the second termination.	☐
Termination-2-Blade	Blade identifier for the second termination, when applicable.	☐
Termination-2-Cassette	Cassette identifier for the second termination, when applicable.	☐
Termination-2-Side	Side of the second termination, such as A or Z.	☐
Termination-2-Port	Port used by the second termination.	☐
Termination-2-Connector-Type	Connector type used by the second termination.	☐
Source-Ref	Reference to the original source row or file from which the patch segment was created.	☐

Termination-* fields are mandatory only when required to identify the physical endpoints of a patch segment; blade, cassette, side, and connector fields are optional when not applicable.

6.1.6 Server Profile

The Server Profile sheet will be used to specify the interface configurations on Hosts, it does not have any significance for switches. The ServerProfile name mentioned in the Nodes sheet should have a corresponding entry here. Server Profile sheet will include below information:

Fabric ID	**CustomNICName**	**NICOSName**	**PhysicalPort**	**RDMAName**	**PCIAddress**

Column	Value Type	Description	Mandatory
FabricID	String	For future use (not supported now)	☐
CustomNICName	String	Custom name given to the NIC if it has been renamed from its default value. This value can be found listed in the output of the command `ip address show`	☒

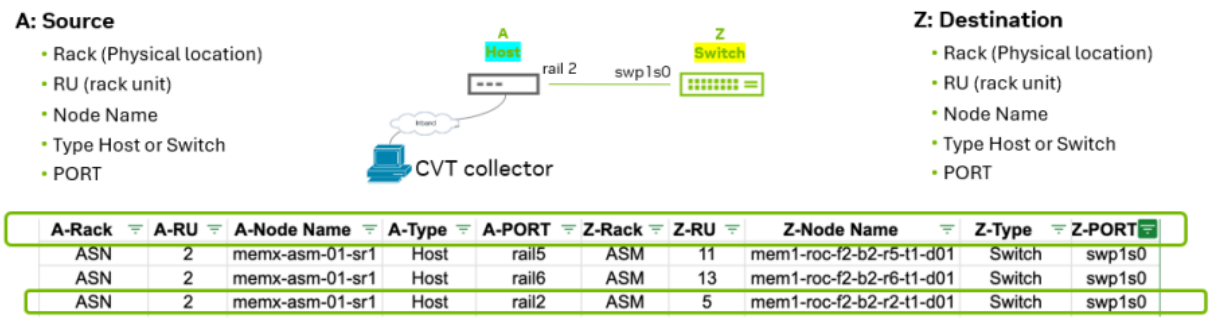
Column	Value Type	Description	Mandatory
NICOSName	String	Default name of the interface. It can be found in the output of the command `ip address show`. If NIC is renamed with a custom name, the default name can be found under the altname tag.	☒
PhysicalPort	String	The OSFP port/slot in IB networks	☐
RDMAName	String	RDMA name of IB ports like mlx5_0, etc. Mandatory for IB ports. Can be found in the output of the command `sudo mst status -v`	☒
PCIAddress	String	For future use (not supported now)	☐

> ⚠ DC Floor layout and Server Profile in Unified Topology are provided in the same Excel workbook and not passed as optional arguments.

6.1.7 Legacy P2P Format (will be deprecated)

The P2P file is an Excel file that details the physical link connections within the fabric. It may consist of multiple sheets, each containing the following columns:

- A-Node Name: Specify the name of the node on the "A" side of the connection.
- A-Type: Indicate the role of the "A" side node, either "Host" or "Switch" (applicable for Ethernet).
- A-Port: Provide the port name for the "A" side node.
- Z-Node Name: Specify the name of the node on the "Z" side of the connection.
- Z-Type: Indicate the role of the "Z" side node, either "Host" or "Switch" (applicable for Ethernet).
- Z-Port: Provide the port name for the "Z" side node.



A-Rack	A-RU	A-Node Name	A-Type	A-PORT	Z-Rack	Z-RU	Z-Node Name	Z-Type	Z-PORT
ASN	2	memx-asm-01-sr1	Host	rail5	ASM	11	mem1-roc-f2-b2-r5-t1-d01	Switch	swp1s0
ASN	2	memx-asm-01-sr1	Host	rail6	ASM	13	mem1-roc-f2-b2-r6-t1-d01	Switch	swp1s0
ASN	2	memx-asm-01-sr1	Host	rail2	ASM	5	mem1-roc-f2-b2-r2-t1-d01	Switch	swp1s0

6.1.7.1 P2P Examples

InfiniBand Example:
A sample sheet for InfiniBand connections in a P2P file:

Rack	U	Name	HCA/Port	Rack	Name	Name	Port
PXH	28	swx-proton03	1	PXX	30	sw-hdr-proton01	1/3
316	24	swx-proton04	3	PXX	27	sw-hdr-proton01	1/4

- The designated port can be a single number or a split port (e.g., 1/2).
- Mapping for HCA ports:
  - 1 → mlx5_0 P1
  - 2 → mlx5_1 P1
  - And so on.

The HCA mapping could be customized by the user, for more details see HCA Mapping File

XDR Example:
A sample sheet for NVOS connections in a P2P file:

Rack	U	Name	HCA/Port	Rack	Name	Name	Port
316	22	clx-abc-073	1	R113	22	bm-abc-t4	sw1p1
316	24	clx-abc-074	1	R113	22	bm-abc-t5	sw1p2

- Designated ports are represented as `sw<port_number>p<split_number>` for switches.
- Mapping for HCA ports: Same as InfiniBand (see above).

Ethernet Example:

A-Rack	A-U	A-Node Name	A-Type	A-Port	Z-Rack	Z-RU	Z-Node Name	Z-Type	Z-Port
ASN	2	memx-asm-01-sr1	Host	rail5	ASM	11	mem1-roc-f2-b2-r5-t1-d01	Switch	swp1s0
ASN	2	memx-asm-01-sr1	Host	rail6	ASM	13	mem1-roc-f2-b2-r6-t1-d01	Switch	swp1s1

Designated ports are represented as `swp<port_number>s<split_number>` for switches.

Please note that in all fabrics the tool relies on the header names to extract the information it needs. So user must have these names exactly as they appear above. If you make a syntax error then it will fail.

6.1.7.2 Legend Sheet

it is mandatory for the PTP file to incorporate a "Legend" sheet, which contains vital details regarding switch and host patterns. The below is an example:
Example:

Name	Model	Switch/HCA	Speed	Rate
`c-csi-mqm*`	MQM9700	Switch	4x 100G	NDR

Name	Model	Switch/HCA	Speed	Rate
`c-csi-0*`	HCA_2	HCA	4x 100G	NDR

6.2 Topo File

This file is specifically supported for InfiniBand and NVOS fabrics. It is generated using the `ibdiagnet` tool with a specific command.

The following is example for topo file:

```
MQM8700 sw-ufm-hdr01
   P2 -4x-50G-> HCA_mlx5_3 swx-ufm3-04 mlx5_0/P1
   P3 -4x-50G-> HCA_mlx5_3 swx-ufm3-04 mlx5_1/P1
   P40 -4x-50G-> MQM9700 sw-ufm-ndr05 P62
   P5 -4x-50G-> HCA_mlx5_3 swx-ufm3-05 mlx5_2/P1
MQM9700 sw-ufm-ndr05
   P61 -4x-50G-> MQM8700 sw-ufm-hdr01 P39
   P62 -4x-50G-> MQM8700 sw-ufm-hdr01 P40
```

6.3 Dot File

This file is specifically supported for Ethernet fabric

```
graph network {
"mose-gbnvl-l1" [ label="mose-gbnvl-l1" role="leaf" os="cumulus"]
"mose-gbnvl-l2" [ label="mose-gbnvl-l2" role="leaf" os="cumulus" ]
"mose-gbnvl-s1" [ label="mose-gbnvl-s1" role="spine" os="cumulus" ]
"mose-gbnvl-s2" [ label="mose-gbnvl-s2" role="spine" os="cumulus"]
"mose-gbnvl-c1" [ label="mose-gbnvl-c1" role="super_spine" os="cumulus"]
"mose-gbnvl-c2" [ label="mose-gbnvl-c2" role="super_spine" os="cumulus"]

"mose-gbnvl-l1":"swp1s0" -- "gb-nvl-yok-04-compute01":"enp3s0np0"
"mose-gbnvl-l1":"swp1s1" -- "gb-nvl-yok-04-compute02":"enp3s0np0"
"mose-gbnvl-l1":"swp2s0" -- "gb-nvl-yok-04-compute03":"enp3s0np0"
"mose-gbnvl-l1":"swp2s1" -- "gb-nvl-yok-04-compute04":"enp3s0np0"
"mose-gbnvl-l1":"swp3s0" -- "gb-nvl-yok-04-compute05":"enp3s0np0"
"mose-gbnvl-l1":"swp3s1" -- "gb-nvl-yok-04-compute06":"enp3s0np0"
"mose-gbnvl-l1":"swp4s0" -- "gb-nvl-yok-04-compute07":"enp3s0np0"
"mose-gbnvl-l1":"swp4s1" -- "gb-nvl-yok-04-compute08":"enp3s0np0"
"mose-gbnvl-l1":"swp5s0" -- "gb-nvl-yok-04-compute09":"enp3s0np0"
"mose-gbnvl-l1":"swp5s1" -- "gb-nvl-yok-03-compute01":"enp3s0np0"
}
```

6.4 Optional Files

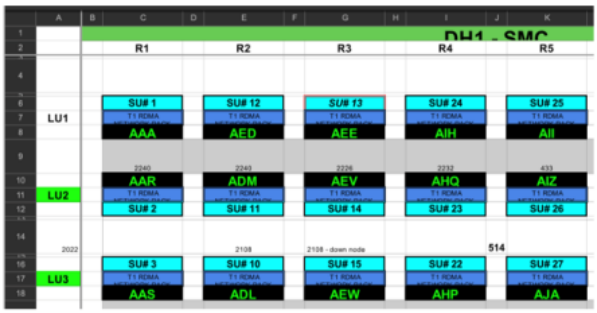
The tool also supports optional files that helps with customizing the topology file.

6.4.1 DC Floor Layout File

Supported for **Ethernet** fabrics, this CSV file allows users to aggregate CVT results based on their Data Center (DC) layout hierarchy, such as:

- Data Hall # or Pod #
- Scalable Unit #
- Row #
- Line Unit #

It defines the mapping of **racks** to a set of desired layout categories (column headers).
Example: below racks are mapped to SU's, LU's, R's, DH



DH	SU	RACK
DH1	SU1	AAA
DH1	SU1	AAB
DH1	SU1	AAC
DH1	SU1	AAD
DH1	SU1	AAE
DH1	SU1	AAF
DH1	SU1	AAG
DH1	SU1	AAH
DH1	SU1	AAI
DH1	SU2	AAR
DH1	SU2	AAJ
DH1	SU2	AAK
DH1	SU2	AAL

DC_layout.csv
(export table as CSV)

6.4.2 HCA Mapping File

Supported for **InfiniBand** and **NVOS** fabrics, this CSV file defines the relationship between port numbers and HCA names in the following format:
port,hca-name

- **port** : Interface number on the host
- **hca-name** : RDMA interface name, which could be mlx5_x, HCA-X, ibp*, or any other naming convention.

By default, if the user didn't provide the **hca_mapping** the tool will use the following assumption:

```
port,hca-name
1, mlx5_0
2, mlx5_1
3, mlx5_2
4, mlx5_3
5, mlx5_4
6, mlx5_5
7, mlx5_6
8, mlx5_7
```

The below is an example of the hca_mapping file:

```
port,hca-name
1,mlx5_1
2,mlx5_0
3,mlx5_2
4,mlx5_4
```

7 Collector

The collector is the main module that should be deployed and run on a host with management network access.

- Deploying the Module
- CVT as a Service
- Device Status
- Hierarchical Anomalies
- Bringup CLI
- CVT Configuration
- Update Certificate
- Apache Configuration for CVT Collector
- Agent Deployment and SSH Configuration
- High Availability (HA) Mode Support for CVT Plugin
- Cluster Sizing Guide
- CVT as UFM Plugin

7.1 Deploying the Module

7.1.1 Server Resource Requirements per Cluster Size

Fabric Size	CPU Requirements*	Memory Requirements	Disk Space Requirements	
			Minimum	Recommended
Up to 1000 nodes	16-core server	16 GB	100 GB	500 GB
1000-5000 nodes	32-core server	64 GB	250 GB	1000 GB (1 TB)
5000-10000 nodes	64-core server	64 GB	1000 GB	2000 GB (2 TB)
Above 10000 nodes	64-core server	64 GB	2000 GB	4000 GB (4 TB)

Supported architectures for collector installation: x86_64 and ARM64 (aarch64)

The following operating systems were tested with Docker Container:

Component	Type and Version
Supported OS	• RHEL8 • RHEL9 • RHEL10 • Ubuntu20.04 • Ubuntu22.04 • Ubuntu24.04 • Debian 10 • Oracle Enterprise Linux 8.10

The Cable Validation tool can be deployed in two methods:

- Deploying the Module#Deploying the Module as Standalone
- Deploying the Module#Deploying the Module as a UFM Enterprise Plugin

7.1.2 Deploying the Module as Standalone

Deploy the `cables_bringup` container on a host as described below:

1. `docker load -i <image_path>/cables_bringup_<version>.tar.gz`
2. `docker run --name cables_bringup -itd --network=host cables_bringup`
3. `docker exec -it cables_bringup /bin/bash`

7.1.2.1 Setting Docker Environment

There are three ways to set environment variables to help customize some of the settings on CVT.

1. Set the values in CVT environment variable configuration file [Least Priority]
2. Setting Environment variables when starting the docker container
3. Exporting environment variables manually inside the container [Highest Priority]

7.1.2.1.1 Environment Variable Configuration File

To enhance flexibility and usability, CVT supports environment variable management through a dedicated file. All variables listed below can be set in the configuration file, which includes default values for easy customization.
If an environment variable is defined in both the Docker environment and the configuration file, the Docker environment value takes precedence.

7.1.2.1.1.1 Step 1: Configuration File

The `cvt_env.conf` file is installed with CVT and comes preloaded with default values.
You can modify this file to match your environment requirements.

7.1.2.1.1.2 Step 2: Updating Variables

To update an environment variable:

1. Edit the `<cable_bringup_root>/config/cvt_env.conf` file.
2. Save your changes.
3. Restart the CVT collector for the changes to take effect.
   This can be done in the following ways:
   a. `supervisorctl restart cvt-service`
   b. `supervisorctl stop cvt-service`
      `bringupcli -k`

> ⚠ Note: A Docker container restart is not required—only the CVT collector needs to be restarted.

Sample cvt_env.conf file

```
# Cable Validation collector settings (same format in every copy of this file).
#
# Which file is used (first match wins):
#   • To change settings in production, edit the copy under your bringup config folder, for example:
#       <BRINGUP_ROOT_PATH>/config/cvt_env.conf
#     (BRINGUP_ROOT_PATH is often /cable_bringup_root or the path you set as BRINGUP_ROOT_PATH.)
#     In plugin mode, the file is at /config/cvt_env.conf
#   • If that file does not exist, as a fallback, the collector uses the built-in default file from the container:
#       /etc/cablevalidation/cvt_default_env.conf
#
# Override: any variable you set in the environment (for example in Docker or systemd) overrides
# the value from the configuration file that was loaded.

[Version]
# DO NOT EDIT THIS SECTION
# Developer note: when adding/removing/changing a variable, you must increment the version number.
# Version of the cvt_env.conf file
# This version is used to check if the cvt_env.conf file is compatible with the current version of the CVT
# If the version is not compatible, the original cvt_env.conf file will be saved as cvt_env.conf.save
# and the new cvt_env.conf file will be created with the current version
# The new cvt_env.conf file will be used to start the CVT. The custom values from old version will be preserved.
CVT_ENV_VERSION = 1.0.12
#
### Variable names are case-sensitive, and should be unique among sections.

[network]
### Network Configuration
# IP addresses used by the agents:
# if no Environment Variable is set, the IP address of the default interface will be used.
# if AGENTS_COLLECTOR_NAT_IP is set
#   - the agents (switch and host) will use this IP address
# otherwise
#   - if DEFAULT_AGENTS_INTERFACE_NAME is set, switch and host agents will use the IP address of the interface
#       specified by DEFAULT_AGENTS_INTERFACE_NAME
#   - if HOST_SPECIFIC_INTERFACE_NAME is set, host agents will use the IP address of the interface
#       specified by HOST_SPECIFIC_INTERFACE_NAME

# Collector External (NAT) IP address and port; define if there is a NAT between the collector and the agents
# this IP address and port are used by the agents to communicate the collector
# fetch images and send data/reports to the collector
# Leave empty if there is no NAT between the collector and the agents
# the NAT port is used for port-forwarding from K8s or other intermediate networks to the collector
# if the port is empty, CVT will use the https port of the collector (APACHE_HTTPS_PORT) or default 443.
AGENTS_COLLECTOR_NAT_IP =
AGENTS_COLLECTOR_NAT_PORT =

# Interface name of the collector over which all the agents (switch and host) will communicate
# The IP address of this interface will be used by all agents (switch and host) to communicate to
# the collector to fetch images and send data/reports
DEFAULT_AGENTS_INTERFACE_NAME =
# Use the following variable if you want to use a different interface of the collector for the host agents
# Define if the interface to connect with hosts is different from the one used for switch agents
# specified by DEFAULT_AGENTS_INTERFACE_NAME
# The IP address of this interface will be used by the host agents to communicate to the
# collector to fetch images and send data/reports
# Leave empty if you want to use the same interface as DEFAULT_AGENTS_INTERFACE_NAME
HOST_SPECIFIC_INTERFACE_NAME =

# ICMP reachability for collector-side connectivity checks (Linux ping: -c count, -W seconds per probe).
# Defaults match previous hardcoded behavior. Environment variables override this file.
PING_COUNT = 1
PING_TIMEOUT_SEC = 3

[agent]
### Agent Configuration (settings used by the agents themselves)
# set `true` if the switch hostname contains a dot (other than the domain part)
CV_DOT_IN_HOSTNAME =
# Time after which a full report is forced to be published.
# Value to be provided in minutes. Default is 720 minutes (12 hours).
# Interval less than 10 mins is not supported.
FULL_REPORT_PUBLISH_INTERVAL_MINUTES = 720
# Set to `true` to publish amber data on each agent iteration, regardless of changes.
# Default is `false` - amber is only published when there are changes or during forced full reports.
AMBER_PUBLISH_EACH_ITERATION = false
# Agent data collection interval in seconds. Default is 600 seconds (10 minutes).
# This controls how often the agent collects and processes port/link data.
AGENT_COLLECT_INTERVAL = 600
# Optional: add arbitrary extra fields/counters to be included in advanced stats.
# These fields are looked up first in the AMBER CSV record (by column name), and if not found,
# they are looked up as AdvancedStats attributes (e.g. transceiver_reinsert_cnt).
# NOTE: all fields must be numeric.
# Format: comma-separated list (whitespace is ignored). Example:
# CUSTOM_AMBER_FIELDS = transceiver_reinsert_cnt, transceiver_swap_cnt, Advanced_Status_Opcode
CUSTOM_AMBER_FIELDS = Advanced_Status_Opcode
# Enable/Disable nvlink addition and validation for gb200/300 nodes.
# Default is `true`. Set to `false` to disable nvlink validation.
NVLINK_VALIDATION = true
CVT_SKIP_ACP_REPORTS = false
# Interval to check ports in seconds (default 10 seconds)
# This controls how often the agent checks the ports for changes.
CHECK_PORTS_INTERVAL = 10
# Maximum interval between authentication failures in seconds (default 10 minutes)
# interval starts from 10 seconds and doubles every failure, till it reaches the maximum interval
MAX_AUTH_FAILURE_INTERVAL = 600

[agent deployment]
### Container Runtime Configuration for Agent Deployment
```

```
# These settings control how agents are deployed on target hosts.
# Supports Docker (default), containerd (via nerdctl), and Kubernetes.
#
# Container runtime to use for agent deployment.
# Leave empty for auto-detection (recommended).
# Supported values: docker, containerd, k8s
# - docker: Uses Docker daemon (default if available)
# - containerd: Uses nerdctl CLI with containerd
# - k8s: Deploys as Kubernetes Pod or DaemonSet
# NOTE: Switch deployments (SONiC, Cumulus) always use Docker regardless of this setting.
CVT_CONTAINER_RUNTIME =
#
# containerd namespace for nerdctl (only used when CVT_CONTAINER_RUNTIME=containerd)
# Default: default
CVT_CONTAINERD_NAMESPACE = default
#
# Kubernetes namespace for agent deployment (only used when CVT_CONTAINER_RUNTIME=k8s)
# The namespace will be created if it doesn't exist.
# Default: cvt-agent
CVT_K8S_NAMESPACE = cvt-agent
#
# Kubernetes Pod/DaemonSet name (only used when CVT_CONTAINER_RUNTIME=k8s)
# Default: cables-agent
CVT_K8S_APP_NAME = cables-agent

[collector]
### Collector Configuration (settings used by the collector to manage agents)
# Minutes of slack after AGENT_COLLECT_INTERVAL before a report is stale/inactive.
# Not heard from since interval+slack matches is_stale / inactive-agent handling.
MAX_INACTIVE_INTERVAL = 5
# Interval to check for new switches in minutes
CHECK_NEW_SWITCHES_INTERVAL = 15
# Time to wait for an agent to become active after start validation (minutes)
START_VALIDATION_TIMEOUT = 5
# Time to wait for an agent to become inactive in minutes
WAIT_TIME_INACTIVE_AGENTS = 1
### Worker Concurrency Settings
# Max number of workers to run in parallel for general operations (validation, connectivity, DNS)
CVT_MAX_WORKERS = 30
# Max number of workers for agent deployment (limited due to 384MB image transfers)
CVT_DEPLOYMENT_MAX_WORKERS = 30
### Timeout Settings
# Quick timeout for unreachable devices (seconds) - reduces wait time for failed connections
CVT_QUICK_TIMEOUT = 3
# Agent communication timeout (seconds) - timeout for individual HTTP requests to agents
AGENT_COMM_TIMEOUT = 30
### Batch Processing Settings
# Batching threshold - only use batching for deployments larger than this (reduces overhead)
CVT_BATCHING_THRESHOLD = 5000
# Batch size when batching is used (devices per batch)
CVT_BATCH_SIZE = 1000
### DNS Resolution Settings
# DNS resolver options for fast timeouts to avoid long waits on unresolvable hostnames
# This configures the system resolver behavior when load_topo performs parallel DNS resolution
# Format: timeout:X attempts:Y single-request
# - timeout:X : seconds to wait per DNS query (default: 1)
# - attempts:Y : number of retry attempts (default: 1, no retries)
# - single-request : send A and AAAA queries separately (improves performance)
CVT_DNS_RES_OPTIONS = timeout:1 attempts:1 single-request

[ssh]
### SSH Configuration
# SSH private key file path for passwordless authentication to HOST devices only.
# NOTE: SSH keys are NOT used for switch devices (switches use password authentication).
# Used by both SSH commands and SFTP file transfers during agent deployment to hosts.
# IMPORTANT: Path must be accessible inside the collector container, not the host system.
# If using Docker volumes, ensure the key file is mounted into the container.
#  - Leave empty to use standard SSH key discovery (recommended)
#  - When empty, SSH will automatically try default container locations like:
#    - ~/.ssh/id_rsa, ~/.ssh/id_dsa, ~/.ssh/id_ecdsa, ~/.ssh/id_ed25519
# Set to specific path only if you need to use a non-standard key location
# Examples:
# 1.  CV_SSH_KEY_FILE = /opt/collector/keys/host_key     (container path)
# 2.  CV_SSH_KEY_FILE = /home/collector/.ssh/custom_key  (container path)
CV_SSH_KEY_FILE =
# SSH connection timeout in seconds
# Applied to both SSH command execution and SFTP file transfers
# Increase for slow networks, decrease for faster failure detection
SSH_CONN_TIMEOUT = 20
# Enable automatic SSH key discovery from SSH agent and default locations
# NOTE: Only applies to HOST devices, not switches
# When enabled, the system will try to use keys from (inside container):
# - SSH agent (if running and accessible in container)
# - Default container locations (~/.ssh/id_rsa, ~/.ssh/id_dsa, etc.)
# Only used for host devices when no password is provided
SSH_LOOK_FOR_KEYS = true

[application]
# Topology loading at startup - specify what to load:
#   none  - do not load any topology file (default)
#   last  - load the last loaded topology from history
#   <path> - load a specific topology file (supports .topo, .dot, .xlsx, .json)
# Examples:
# 1.  STARTUP_TOPOLOGY = none
# 2.  STARTUP_TOPOLOGY = last
# 3.  STARTUP_TOPOLOGY = ./topologies/production.topo
# 4.  STARTUP_TOPOLOGY = /absolute/path/to/topology.xlsx
STARTUP_TOPOLOGY = none
# Credentials file to load at startup
# If not set, no credentials will be loaded
# Examples:
# 1.  STARTUP_CREDENTIALS_FILE = /opt/collector/credentials.ini
# 2.  STARTUP_CREDENTIALS_FILE = /absolute/path/to/credentials.json
# 3.  STARTUP_CREDENTIALS_FILE = none
# If STARTUP_CREDENTIALS_FILE is set to none, no credentials will be loaded
```

```
STARTUP_CREDENTIALS_FILE = none
# automatically start validation if a topology file is loaded
AUTO_START_VALIDATION = false

[apache]
### Apache Web Server Configuration
# Apache performance profile based on server resources and cluster scale
# This controls Apache's concurrency, connection handling, and optimization settings
# Options:
#   small   - 2-16 cores / 8-64GB RAM / < 1,000 agents (default)
#   medium  - 32-128 cores / 128-512GB RAM / 1,000-10,000 agents
#   large   - 200-300 cores / 1TB+ RAM / 10,000-20,000 agents
#   xlarge  - 400+ cores / 1.5TB+ RAM / 20,000+ agents
#   auto    - Auto-detect based on available CPU cores
CVT_APACHE_PROFILE = small
#
### Apache Logging Configuration
# Log level for Apache error logs. Higher levels reduce log volume.
# Options (from most to least verbose):
#   debug, info, notice, warn (default), error, crit, alert, emerg
# For high-scale production deployments, consider using 'error' or 'crit'
CVT_APACHE_LOG_LEVEL = warn
#
# Enable or disable Apache access logs (request logging)
# Set to 'false' to disable access logs and reduce disk I/O on high-traffic deployments
# Options: true (default), false
CVT_APACHE_ACCESS_LOG = true

[data management]
# set `true` if you want to poll for stats from CVT collector
ENABLE_STATS_POLLING = false
#
### Data Archiving Configuration
# Cron schedule for data directory archiving (compresses old date directories)
# Format: minute hour day month weekday
# Default: 5 0 * * * (daily at 00:05 UTC)
# Adjust to run during low-usage periods in your timezone
# Examples:
#   0 6 * * *   = 06:00 UTC (1am CT / 2am ET)
#   0 2 * * *   = 02:00 UTC (9pm ET previous day)
#   30 7 * * *  = 07:30 UTC (2:30am CT)
CVT_DATA_ARCHIVE_SCHEDULE = 5 0 * * *
#
# Maximum number of plain (uncompressed) data directories to keep
# Older directories are compressed to .tgz archives
CVT_MAX_PLAIN_DATA_DIRS = 7
#
# Maximum total data entries (plain + archived) to keep
# Oldest archives are deleted when this limit is exceeded
CVT_MAX_DATA_DIRS = 30

[database management]
# Housekeeping for collector DBs.
# circuit_history: delete rows older than this many days. Default 14. Set to 0 to disable age-based pruning.
CVT_CIRCUIT_HISTORY_MAX_AGE_DAYS = 14
#
# circuit_history: maximum rows to keep; oldest deleted when exceeded. Default 100000. Maximum is 200000. Set to 0
 to disable.
CVT_CIRCUIT_HISTORY_MAX_RECORDS = 100000

[ufm events]
# Enable/Disable sending cable validation issues to UFM as external events.
# Default is `false`. Set to `true` to enable sending external events.
ENABLE_UFM_EXTERNAL_EVENTS = false
#
# Send UFM external events only for newly detected issues (true),
# or for all issues on every report cycle (false).
# Default is true.
UFM_SEND_ON_NEW_ISSUE_ONLY = true
#
# Maximum events to send in a single batch (cap per send).
# Keeps individual UFM API requests small to avoid timeouts and excessive memory use.
# Set to 0 to disable batching and send all events in a single request (may cause issues with large event counts).
# Default (when unset): 100.
UFM_MAX_EVENTS_PER_SEND = 100
#
# Syndrome severity overrides. Valid values: INFO, WARNING, CRITICAL
# Uncomment and change only the syndromes you want to override.
# Default for all syndromes is WARNING.
# UFM_SEVERITY_LINK_DOWN = WARNING
# UFM_SEVERITY_PROTO_DOWN = WARNING
# UFM_SEVERITY_ADMIN_DOWN = WARNING
# UFM_SEVERITY_RX_DISABLE = WARNING
# UFM_SEVERITY_NEG_FAIL = WARNING
# UFM_SEVERITY_UNPROVISIONED = WARNING
# UFM_SEVERITY_WRONG_NEIGHBOR = WARNING
# UFM_SEVERITY_UNKNOWN_NEIGHBOR = WARNING
# UFM_SEVERITY_WRONG_PORT = WARNING
# UFM_SEVERITY_MISSING_CABLE = WARNING
# UFM_SEVERITY_EXTRA_CABLE = WARNING
# UFM_SEVERITY_FLAPPING_LINK = WARNING
# UFM_SEVERITY_ANOMALOUS_PORT = WARNING
# UFM_SEVERITY_MODULE_PN_MISMATCH = WARNING
```

More details on the configuration parameters can be found at CVT Configuration

7.1.2.1.2 Setting Environmental Variables with Docker Run

7.1.2.1.2.1 Specifying the Network Interface

If the host system is equipped with multiple network interfaces and the switches are connected to the host through an interface that differs from the default management interface, the user can designate this particular interface by utilizing a specific environment variable, namely AGENTS_IFC_NAME. To illustrate, assuming the hypothetical interface name is eno3:

```
docker run --name cables_bringup -itd --network=host --env AGENTS_IFC_NAME=eno3
```

7.1.2.1.2.2 Adding Hostnames

If the switches are not configured in the DNS server, you may add hostnames; the user may use the --add-host option when running the container. For example (assuming the switch name is switch-3245fa and its IP is 192.168.1.1):

```
docker run --name cables_bringup -itd --network=host --add-host=switch-3245fa:192.168.1.1 cables_bringup
```

7.1.2.1.2.3 Using Volumes

Volumes can be used for data persistence or easier file transfer to the cables_bringup container. The volume must be mapped to /cable_bringup_root in the container for data persistence. This volume can also be used for loading topology files. Example:

```
docker run --name cables_bringup -itd --network=host -v /opt/bringup_data:/cable_bringup_root cables_bringup
```

7.1.2.1.2.4 Overriding Apache Configuration

In the event that a host machine is running another Apache instance and utilizing the default ssh ports 443, an alternative port may be designated for the bringup server by the user, these ports should be available and free. To accomplish this, the APACHE_HTTPS_PORT environment variables can be employed. Consider the following example:

```
docker run --name cables_bringup -itd --network=host --env APACHE_HTTPS_PORT=9443 cables_bringup
```

7.1.2.1.2.5 Specify that hostname contains dot

If the hostnames of the network elements being validated have a dot in them, this information has to be set in CVT so that the hostname will not be wrongly truncated when removing the domain name based on dot. If hostnames contain dot, then the topology input file should be created without domain name in the node name field. To accomplish this, the **CV_DOT_IN_HOSTNAME** environment variables can be set to **True**. Consider the following example:

```
docker run --name cables_bringup -itd --network=host --env CV_DOT_IN_HOSTNAME=True cables_bringup
```

7.1.3 Deploying the Module as a UFM Enterprise Plugin

> Warning: Please note that Running Cable Validation as plugin is not supported on UFM Gen2.0.

Deploy the module as a UFM Enterprise plugin as follows:

1. `docker load -i /<image_path>/ufm-plugin-cablevalidation-<version>.tar.gz`
2. `./manage_ufm_plugins.sh add -p cablevalidation -t <version>`
3. `./manage_ufm_plugins.sh start -p cablevalidation`
4. `docker exec -it ufm-plugin-cablevalidation bash`

7.1.3.1 Copy Files to the Plugin

Users have two methods for copying files, such as topology files, to the Cable Validation plugin:

1. Copy the files to the plugin's data volume /opt/ufm/ufm_plugins_data/cablevalidation which is mapped to /data/ inside the plugin container.
2. Use docker cp to copy the needed files to the container.

7.1.3.2 Overriding the Apache Configuration

When using Cable Validation as a plugin, the default ports 443 are already in use by UFM Enterprise. Therefore, port 8633 will be used for HTTPS by default. Users can opt to use different ports for the bring-up server, provided that these ports are available and free.
The plugin config.cfg file can be modified to update APACHE_HTTPS_PORT variables for that purpose. To make this adjustment, follow these steps:

1. Execute /opt/ufm/scripts/manage_ufm_plugins.sh add -p cablevalidation to add the Cable Validation plugin.
2. Stop the plugin using /opt/ufm/scripts/manage_ufm_plugins.sh stop -p cablevalidation
3. Use vim /opt/ufm/files/conf/plugins/cablevalidation/config.cfg to modify the 'APACHE_HTTPS_PORT' variable.
4. Update and save the file.
5. Start the plugin again with /opt/ufm/scripts/manage_ufm_plugins.sh start -p cablevalidation.

With these changes, the new configuration will take effect, and Apache will run with the updated ports.

7.1.3.3 Troubleshooting

7.1.3.3.1 Collector hostname resolution and agent reachability

7.1.3.3.2 Symptom

Agents fail to fetch images or publish reports. At collector startup you may see:

```
Collector IP address that will be used by the agents: 127.0.0.1
```

or agents are configured to reach `https://127.0.0.1:443/...`, which is not reachable from switches and hosts on the management network.

7.1.3.3.3 Cause

When no explicit collector address is configured, CVT auto-detects the IP by resolving the collector's hostname (`socket.gethostname()`). If DNS is unavailable or misconfigured, resolution falls back to `/etc/hosts`. On many Linux systems the hostname is mapped to `127.0.1.1`, which resolves to loopback — so agents are told to use `127.0.0.1` instead of the real management-network address.

The collector container startup script comments out the `127.0.1.1` line in `/etc/hosts` as a partial workaround, but that alone does not help when DNS is absent and no correct `/etc/hosts` entry exists.

7.1.3.3.4 Fix 1 — Add the collector hostname to `/etc/hosts` (labs without DNS)

On the collector host (and inside the collector container if hostname resolution happens there), map the collector hostname to its management-network IP:

```
# /etc/hosts
192.168.10.50   cvt-collector.my.lab   cvt-collector
```

Replace `192.168.10.50` with the address agents must use to reach the collector, and use the hostname returned by `hostname` on the collector.

Verify:

```
hostname
getent hosts "$(hostname)"
```

The resolved address must be the management IP, not `127.0.0.1` or `127.0.1.1`.

For Docker, you can also pass the mapping at run time:

```
docker run --name cables_bringup -itd --network=host \
  --add-host=cvt-collector.my.lab:192.168.10.50 \
  cables_bringup
```

7.1.3.3.5 Fix 2 — Set the collector IP explicitly in `cvt_env.conf` (recommended)

Prefer setting the address in the collector environment file so agents do not depend on hostname resolution.

File location (first match wins):

Deployment	Path
Production / bringup	`${BRINGUP_ROOT_PATH}/config/cvt_env.conf`
Plugin mode	`/config/cvt_env.conf`
Container default (if no persisted copy)	`/etc/cablevalidation/cvt_default_env.conf`

Edit the `[network]` section:

```
[network]

# Option A — NAT, K8s, load balancer, or any case where agents reach
# the collector through a fixed external address:
AGENTS_COLLECTOR_NAT_IP = 192.168.10.50
AGENTS_COLLECTOR_NAT_PORT = 443          # optional; defaults to APACHE_HTTPS_PORT or 443

# Option B — no NAT; bind to a specific interface on the collector host:
DEFAULT_AGENTS_INTERFACE_NAME = eno3     # switch and host agents

# Option C — different interfaces per agent type (only if needed):
HOST_SPECIFIC_INTERFACE_NAME = eno4      # host agents only
NVLSWITCH_SPECIFIC_INTERFACE_NAME = eth0 # NVL switch agents only
```

Precedence (highest first):

1. Environment variable set on the container (overrides the file)
2. `AGENTS_COLLECTOR_NAT_IP` — if set, all agent types use this address
3. Interface-based settings (`DEFAULT_AGENTS_INTERFACE_NAME` , etc.)
4. Auto-detect via hostname resolution (fallback that can yield `127.0.0.1`)

IPv6 literals are supported in `AGENTS_COLLECTOR_NAT_IP` ; see IPv6 dual-stack support for NAT and interface examples.

For more information, refer to CVT Configuration.

7.1.3.3.6 Apply and verify

1. Save `cvt_env.conf` (or set the equivalent `-e` / `--env` variables on the container).
2. Restart the CVT service so the collector reloads configuration.
3. On startup, confirm the printed line shows the expected address:

```
Collector IP address that will be used by the agents: 192.168.10.50
```

4. From a switch or host on the management network, verify reachability:

```
curl -sk "https://192.168.10.50:443/version"
```

7.1.3.3.7 Related settings

Variable	Section	Purpose
`AGENTS_COLLECTOR_NAT_IP`	`[network]`	Fixed IP agents use to reach the collector (NAT/K8s/explicit)
`AGENTS_COLLECTOR_NAT_PORT`	`[network]`	External port when NAT/port-forwarding is in use

Variable	Section	Purpose
`DEFAULT_AGENTS_INTERFACE_NAME`	`[network]`	Interface whose IP is advertised to switch and host agents
`HOST_SPECIFIC_INTERFACE_NAME`	`[network]`	Override interface for host agents
`NVLSWITCH_SPECIFIC_INTERFACE_NAME`	`[network]`	Override interface for NVL switch agents
`APACHE_HTTPS_PORT`	env / Apache config	HTTPS port when `AGENTS_COLLECTOR_NAT_PORT` is unset

7.2 CVT as a Service

7.2.1 Overview

The Cable Validation Tool (CVT) can be deployed and managed as a service, providing automated cable validation capabilities with persistent operation, automatic startup, and web-based management. This document covers the service architecture, configuration, deployment, and management of CVT in service mode.

7.2.2 Architecture

The CVT service architecture consists of multiple components managed by `supervisord.`

```
CVT Container
┌──────────────────────────────────────────────────────────┐
│ supervisord
│ ┌──────────────────────────────────────────────┐ │
│ │ CVT Service │ │
│ │ ┌──────────────────────────────────────────┐│ │
│ │ │ BringupCLI Engine │ │ │
│ │ │ • Topology Management │ │ │
│ │ │ • Agent Deployment │ │ │
│ │ │ • Validation Orchestration │ │ │
│ │ └──────────────────────────────────────────┘│ │
│ │ ┌──────────────────────────────────────────┐│ │
│ │ │ Web Service │ │ │
│ │ │ • REST API │ │ │
│ │ │ • Web UI │ │ │
│ │ │ • Authentication │ │ │
│ │ └──────────────────────────────────────────┘│ │
│ └──────────────────────────────────────────────┘ │
│ ┌──────────────────────────────────────────────┐ │
```

```
│   │     Apache Web Server        │ │
│   │ • HTTPS/SSL Support          │ │
│   │ • Reverse Proxy              │ │
│   │ • Static Content Serving     │ │
│   └──────────────────────────────┘ │
│   ┌──────────────────────────────┐ │
│   │     CV Controller            │ │
│   │ • Service Management API     │ │
│   │ • System Administration      │ │
│   └──────────────────────────────┘ │
└────────────────────────────────────┘
```

7.2.3 Service Components

7.2.3.1 CVT Service (`cvt-service`)

- Purpose: Main CVT application running in daemon mode
- Command: `/usr/local/bin/cvt-service --daemon`
- Features:
  - Automatic topology loading
  - Validation orchestration
  - Agent management
  - Signal handling (SIGTERM, SIGINT)

7.2.3.2 Apache Web Server (`apache`)

- Purpose: HTTPS frontend and reverse proxy
- Command: `/usr/local/bin/apache2_wrapper.sh`
- Features:
  - SSL/TLS termination
  - Web UI serving
  - API reverse proxy
  - Authentication handling

7.2.3.3 CV Controller (`cv_controller`)

- Purpose: Service management and system administration
- Command: `/usr/local/bin/run_cv_controller.sh`
- Features:
  - Service start/stop operations
  - System status monitoring
  - Configuration management

7.2.3.4 Cron Service (`cron`)

- Purpose: Scheduled tasks and maintenance
- Command: `cron -f`
- Features:
  - Log rotation
  - Periodic cleanup
  - Scheduled reports

7.2.4 Installation and Deployment

7.2.4.1 Docker Deployment for CVT Service

The CVT service is deployed using a pre-built Docker image (bringup service image). This allows for a consistent and isolated runtime environment.

1. Load the CVT Collector Service Image.
   If you have a local Docker image file (`.tar.gz`), load it using:

```
docker load -i /path/to/cables_bringup_{version}.tar.gz
```

2. [Optional] Pull the Image from a Registry
   If the image is available in a Docker registry, you can pull it directly:

```
docker pull mellanox/cables_bringup:{version}
```

3. Run the CVT Service Container: Use the following command to start the CVT service:

```
docker run -itd \
  --name cables_bringup \
  --network=host \
  {override/add env variables} \
  -v /path/to/data:/cable_bringup_root \
  mellanox/cables_bringup:{version}
```

Options:

Option	Description
`-itd`	Run container in interactive mode, allocate a pseudo-TTY, and run in detached mode
`--name cables_bringup`	Assigns a name to the container for easier management.
`--network=host`	Shares the host's network stack for the container.
`{override/add env variables}`	Specify any environment variables needed by the service.
`-v /path/to/data:/cable_bringup_root`	Mounts a host directory into the container for persistent data.
`mellanox/cables_bringup:{version}`	The Docker image and version tag to run.

7.2.5 Configuration

7.2.5.1 Environment Variables

The CVT service behavior is controlled through environment variables and configuration files:

7.2.5.1.1 Configuration File (`/<cable_bringup_root>/config/cvt_env.conf`)

Run the following command after editting the file to restart cvt-service:

```
supervisorctl restart cvt-service
```

For more information, refer to CVT Configuration.

7.2.5.1.2 Environment Variable Configuration

Run the following command to configure environment variables:

```
docker run
```

7.2.5.1.3 Web Service Configuration

Web service configuration can be done only in the `docker run` command.

```
# Web service interface and ports
export CVT_WEB_INTERFACE=127.0.0.1
export CVT_WEB_SERVICE_PORT=8251
export APACHE_HTTPS_PORT=443

# Controller service port
export CV_CONTROLLER_PORT=8252
```

7.2.5.1.4 UFM Plugin Mode

Install the CVT UFM plugin using the UFM CLI or Web UI, like any other plugin.

7.2.5.1.5 NetQ Plugin Mode

TBD

7.2.5.1.6 Supervisord Configuration

The service is managed by `supervisord` with the following configuration:

Directive	Description
`command`	Command to start the CVT service in daemon mode.

Directive	Description
`autostart`	Automatically starts the service when Supervisor starts.
`autorestart`	Restarts the service automatically if it exits unexpectedly.
`startretries`	Number of retry attempts if the service fails to start.
`startsecs`	Number of seconds the service must stay running to be considered started.
`stderr_logfile`	Path to the log file for standard error output.
`stdout_logfile`	Path to the log file for standard output.
`stdout_logfile_maxbytes` / `stderr_logfile_maxbytes`	Maximum size of each log file before rotation.
`stdout_logfile_backups` / `stderr_logfile_backups`	Number of rotated log backups to keep.
`user`	The user under which the service runs.
`killasgroup`	Terminates all processes in the service's process group on stop.
`stopasgroup`	Sends stop signals to the entire process group.
`environment`	Sets environment variables for the service (`PYTHONUNBUFFERED=1` ensures unbuffered Python output).

7.2.6 Service Management

7.2.6.1 CVT Service Management with Supervisor

The CVT service is managed through Supervisor, which allows you to start, stop, restart, and monitor services on your appliance. Below are the common commands:

Service	Description	Command
Start CVT Service	Starts the CVT service if it is not already running.	`supervisorctl start cvt-service`
Stop CVT Service	Stops the CVT service gracefully.	`supervisorctl stop cvt-service`
Restart CVT Service	Stops and then starts the CVT service. Useful for applying configuration changes.	`supervisorctl restart cvt-service`
Check Service Status	Displays the current status of the CVT service (e.g., RUNNING, STOPPED, FATAL).	`supervisorctl status cvt-service`
View Service Logs	Displays real-time logs from the CVT service. Useful for troubleshooting.	`supervisorctl tail cvt-service`
Start All Services	Starts all services managed by Supervisor.	`supervisorctl start all`

Service	Description	Command
Reload Configuration	• **reread**: Reloads Supervisor configuration files to detect new or removed programs. • **update**: Applies the configuration changes, starting or stopping services as needed.	`supervisorctl reread && supervisorctl update`

7.2.6.2 CVT Service Commands

Command	Description
`supervisorctl start cvt-service`	Starts the CVT service if it is not already running.
`supervisorctl stop cvt-service`	Stops the CVT service gracefully.
`supervisorctl restart cvt-service`	Restarts the CVT service. Useful for applying configuration changes.
`supervisorctl status cvt-service`	Displays the current status of the CVT service (RUNNING, STOPPED, FATAL).
`supervisorctl tail cvt-service`	Displays real-time logs from the CVT service for troubleshooting.
`supervisorctl start all`	Starts all services managed by Supervisor.
`supervisorctl reread && supervisorctl update`	Reloads configuration

7.2.6.3 Using CVT Controller API

The CVT Controller provides programmatic service management:

Command	Description
`curl -X POST` http://localhost:8252/start	Starts the CVT service via the controller.
`curl -X POST` http://localhost:8252/stop	Stops the CVT service via the controller.
`curl -X GET` http://localhost:8252/status	Retrieves the CVT service status via the controller. **Note:** This feature is not supported yet.

7.2.6.4 Legacy Mode Operation

For troubleshooting, development, or manual operations, you can stop the CVT service and run the traditional interactive CLI:

7.2.6.4.1 Stop Service and Run Legacy CLI

1. Stop the CVT Service

```
supervisorctl stop cvt-service
```

2. Run the Legacy Bringup CLI:

```
bringupcli
```

7.2.6.4.2 Legacy CLI Features

When running in legacy mode, you get:

- Interactive Command Shell: Full command-line interface with tab completion
- Manual Control: Step-by-step topology loading and validation control
- Real-time Feedback: Immediate command output and status updates
- Debugging: Easier debugging with direct command execution

7.2.6.4.3 Common Legacy Mode Workflow

1. Stop the CVT service:

```
supervisorctl stop cvt-service
```

2. Start interactive CLI:

```
bringupcli
```

3. Perform manual operations:

```
Cable Bringup: load_topo /path/to/topology.topo
Cable Bringup: deploy_all_agents
Cable Bringup: start_validation
Cable Bringup: show_switches
Cable Bringup: exit
```

4. Restart service when done:

```
upervisorctl start cvt-service
```

7.2.6.4.4 When to Use Legacy Mode

- Initial Setup: First-time configuration and testing
- Troubleshooting: Debugging connectivity or configuration issues
- Manual Operations: One-time tasks that don't require automation
- Development: Testing new topologies or validation scenarios
- Training: Learning CVT commands and workflows

7.2.6.4.4.1 Switching Back to Service Mode

1. Exit the legacy CLI:

```
Cable Bringup: exit
```

2. Restart the CVT service:

```
supervisorctl start cvt-service
```

3. Verify that the service is running:

```
supervisorctl status cvt-service
```

> While in legacy mode, automatic features (e.g., topology loading and validation) are disabled. The web UI remains accessible. Restarting the service restores full automatic functionality.

Command Line Options

The CVT service supports several command line options:

Command	Description
`cvt-service --daemon`	Runs the CVT service as a background daemon.
`cvt-service --kill-other-sessions`	Terminates any other running CVT sessions before starting a new one.
`cvt-service --version`	Displays the current version of the CVT service.
`cvt-service`	Starts the CVT service in interactive mode, useful for debugging.

7.2.7 Automatic Features

7.2.7.1 Topology Loading

The CVT service can automatically load topology files on startup based on the `STARTUP_TOPOLOGY` configuration:

7.2.7.1.1 Load Last Topology

`STARTUP_TOPOLOGY=last`

- Loads the most recently used topology from history
- Supports all topology formats (.topo, .dot, .xlsx, .json)

7.2.7.1.2 Load Specific File

`STARTUP_TOPOLOGY=/path/to/topology.topo`

- Loads a specific topology file on startup
- Supports absolute and relative paths
- Auto-detects file format based on extension

7.2.7.1.3 Supported Formats

- `.topo` : Native topology format
- `.dot` : Graphviz DOT format
- `.xlsx` : Unified topology Excel format
- `.json` : JSON topology format

7.2.7.2 Automatic Validation

Enable automatic validation start after topology loading:

```
AUTO_START_VALIDATION=true
```

Behavior:

- Only starts if topology was successfully loaded
- Uses validation settings from history
- Logs all activities for monitoring

7.2.7.2.1 Signal Handling

The service handles system signals gracefully:

- SIGTERM: Graceful shutdown with cleanup
- SIGINT:
  - Daemon mode: Graceful shutdown
  - Interactive mode: Cancel current operation (double Ctrl-C to exit)

7.2.7.3 Monitoring and Logging

7.2.7.3.1 Log Files

All service logs are centrally managed:

```
/cable_bringup_root/log/
├── cvt-service.log          # Main service logs
├── cv_controller_service.log   # Controller logs
├── apache2/
│   ├── access.log           # Web access logs
│   └── error.log            # Apache error logs
└── supervisord.log          # Supervisor logs
```

7.2.7.3.2 Log Rotation

Logs are automatically rotated with the following settings:

- Max size: 10MB per log file
- Backups: 7 historical files

- Format: Timestamped entries with log levels

7.2.7.3.3 Health Monitoring

Monitor service health using:

Command	Description
`supervisorctl status`	Checks the status of all services managed by Supervisor.
`supervisorctl tail -f cvt-service`	Monitors CVT service logs in real-time.
`systemctl status apache2`	Displays the current status of the Apache web server.
`htop`	Interactive tool to monitor system processes and resource usage.
`iostat -x 1`	Displays detailed I/O statistics for devices every 1 second.

7.2.8 Troubleshooting

7.2.8.1 Common Issues

Issue	Troubleshooting

Service Won't Start	1. **Check Supervisord Status**:
	`supervisorctl status`
	`systemctl status supervisor`
	2. **Check Log Files**:
	`tail -f /cable_bringup_root/log/cvt-service.log`
	`tail -f /var/log/supervisor/supervisord.log`
	3. **Verify Configuration**:
	`supervisorctl reread`
	`supervisorctl update`

Port Conflicts	1. **Check Port Usage**:
	`netstat -tlnp | grep -E ':(443|8251|8252)'`
	`lsof -i :443`
	2. **Update Port Configuration**:
	`# Edit environment variables`
	`export CVT_WEB_SERVICE_PORT=8253`
	`export CV_CONTROLLER_PORT=8254`

Permission Issues	1. **Check File Permissions**:
	`ls -la /cable_bringup_root/`
	`ls -la /etc/cablevalidation/`
	2. **Fix Ownership**:
	`chown -R root:root /cable_bringup_root/`
	`chmod -R 755 /cable_bringup_root/`

Issue	Troubleshooting

Memory/Performance Issues	1. **Monitor Resource Usage**:
	`htop`
	`free -h`
	`df -h`
	2. **Adjust Worker Limits**:
	`# In cvt_env.conf`
	`CVT_MAX_WORKERS=15  # Reduce for limited memory`

Topology Loading Failures	1. **Check File Paths**:
	`ls -la /path/to/topology/file`
	2. **Validate File Format**:
	`file topology.topo`
	`head -n 10 topology.topo`
	3. **Check History**:
	`# View topology loading history in logs`
	`grep "Loading topology" /cable_bringup_root/log/cvt-service.log`

7.2.8.2 Debug Mode

For detailed debugging, run CVT in interactive mode:

Command	Description
`supervisorctl stop cvt-service`	Stops the CVT service managed by Supervisor.
`/usr/local/bin/cvt-service`	Runs the CVT service in interactive mode.
`PYTHONUNBUFFERED=1 /usr/local/bin/cvt-service`	Runs the CVT service interactively with debug logging enabled.

7.2.8.2.1 Log Analysis

Analyze logs for common patterns:

Command	Description
`grep -i error /cable_bringup_root/log/cvt-service.log`	Searches the CVT service log for error messages.
`grep -i "starting|daemon" /cable_bringup_root/log/cvt-service.log`	Monitors the startup sequence of the CVT service.
`grep -i "topology|load" /cable_bringup_root/log/cvt-service.log`	Checks log entries related to topology loading.
`grep -i "validation|start|stop" /cable_bringup_root/log/cvt-service.log`	Monitors validation activities and their start/stop events.

7.2.9 Best Practices

	Flow
Security	1. **Use HTTPS**: Always use HTTPS for production deployments 2. **Strong Authentication**: Configure strong passwords for web users 3. **SSH Keys**: Use SSH key-based authentication for agent deployment 4. **Firewall**: Restrict access to necessary ports only 5. **Regular Updates**: Keep CVT and system packages updated
Performance	1. **Resource Allocation**: Ensure adequate CPU and memory resources 2. **Worker Tuning**: Adjust `CVT_MAX_WORKERS` based on system capacity 3. **Log Management**: Implement log rotation and archival 4. **Monitoring**: Set up comprehensive monitoring and alerting
Reliability	1. **Backup Configuration**: Regularly backup configuration files and data 2. **Health Checks**: Implement automated health monitoring 3. **Graceful Shutdown**: Always use proper shutdown procedures 4. **Recovery Procedures**: Document and test disaster recovery procedures
Maintenance	1. **Regular Monitoring**: Check service status and logs regularly 2. **Capacity Planning**: Monitor resource usage and plan for growth 3. **Update Procedures**: Establish procedures for updates and patches 4. **Documentation**: Keep configuration and operational documentation current
Development and Testing	1. **Staging Environment**: Use a staging environment for testing changes 2. **Configuration Management**: Use version control for configuration files 3. **Automated Testing**: Implement automated testing for service functionality 4. **Rollback Procedures**: Have rollback procedures ready for failed deployments

7.3 Device Status

This page explains how CVT tracks the lifecycle of each device in your topology, how statuses affect CVT operations, and how to manage device statuses through the UI.

7.3.1 What Is Device Status?

Every device in a CVT topology is assigned a lifecycle status that reflects where the device stands in the journey from initial discovery through validation and into production. CVT uses this status to decide which operations are appropriate for each device -- for example, whether to ping it, deploy an agent, or include it in validation.

Statuses change in two ways:

- Automatically -- CVT promotes or demotes devices based on connectivity checks, pre-check results, agent deployment outcomes, and validation events.
- Manually -- Administrators set statuses through the UI when a device needs to be placed into maintenance, marked for production, excluded from management, and so on.

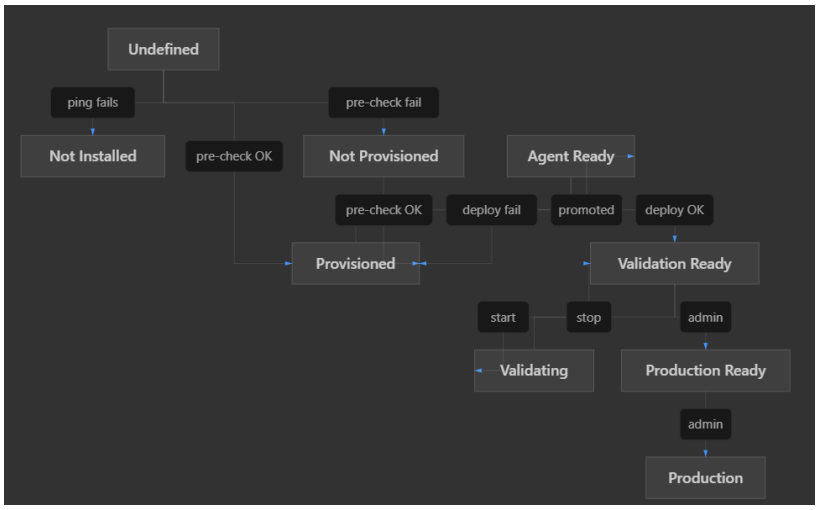
Only valid transitions are allowed. If a transition does not make sense for a device's current status, CVT rejects it and explains why.

7.3.1.1 Device Statuses at a Glance

Status	Description
Undefined	Initial state after topology load. CVT has not yet checked the device.
Not Installed	Device does not respond to ping. Assumed to be absent or unreachable.
Installed No Power	Device is physically present but not powered on. Behaves the same as Not Installed.
Not Provisioned	Device is reachable but failed one or more pre-checks (e.g., SSH credentials, disk space, OS version).
Provisioned	Device passes all pre-checks and is ready for an agent deployment decision.
Unsupported	Device model is not supported by CVT.
Agent Ready	Device is eligible for agent deployment. Only devices in this status appear in the deployment wizard.
Validation Ready	Agent is deployed and the device is ready to start validation.
Validating	Validation is actively running on this device.
Validation Only	Agent is externally managed (not by CVT). CVT can validate but cannot remove the agent.
Production Ready	Validation is complete and the device is ready to be promoted to production.
Production	Device is in production. The agent is idle and the device is hidden from the circuit view.
Drain / Troubleshooting	A production device that requires investigation. CVT re-deploys the agent and starts validation.
Maintenance	Device is under maintenance. All alarms are suppressed, validation is stopped, and pinging is disabled.
Unmanaged	Device is excluded from all CVT operations. Appears grayed out in the rack view.

7.3.1.2 Lifecycle Overview

The diagram below shows the primary path a device takes from discovery through production. This is the path that devices follow automatically when everything is working correctly.



At any point, an administrator can move a device into one of the operational statuses: Maintenance, Drain / Troubleshooting, Unmanaged, or Validation Only. The full set of allowed transitions is listed in the reference table below.

7.3.1.2.1 Allowed Transitions Reference

From	Allowed Targets
Undefined	Not Installed, Provisioned, Not Provisioned, Unmanaged
Not Installed	Installed No Power, Not Provisioned, Unmanaged
Installed No Power	Not Installed, Not Provisioned, Unmanaged
Not Provisioned	Provisioned, Unmanaged
Provisioned	Not Provisioned, Agent Ready, Unmanaged, Unsupported
Unsupported	Not Provisioned
Agent Ready	Validation Ready, Provisioned
Validation Ready	Validating, Production Ready, Drain / Troubleshooting, Maintenance, Provisioned, Agent Ready
Validating	Validation Ready, Production Ready, Drain / Troubleshooting, Maintenance, Provisioned, Agent Ready
Validation Only	Production Ready, Maintenance, Provisioned, Agent Ready
Production Ready	Production, Drain / Troubleshooting, Provisioned, Agent Ready
Production	Drain / Troubleshooting, Maintenance, Provisioned, Agent Ready
Drain / Troubleshooting	Validation Ready, Maintenance, Provisioned, Agent Ready
Maintenance	Provisioned, Agent Ready, Validation Ready
Unmanaged	Not Installed, Not Provisioned, Provisioned

7.3.1.3 Typical Device Journey

A device that is functioning correctly and has proper credentials will automatically progress through these statuses without any administrator intervention:

1. Undefined -- topology is loaded.
2. Provisioned -- CVT pings the device and runs pre-checks (SSH access, disk space, OS version). All checks pass.
3. Agent Ready -- CVT confirms the device is eligible for agent deployment.
4. Validation Ready -- the agent is deployed successfully.
5. Validating -- validation starts.

From there, the administrator reviews validation results and decides whether to promote the device to Production Ready and eventually to Production.

If something goes wrong at any step (ping failure, pre-check failure, deploy failure), CVT automatically moves the device to the appropriate status so you can see at a glance what needs attention.

7.3.1.4 How Status Affects CVT Operations

Each status determines which CVT operations apply to the device:

Status	Ping	Agent Deploy	Start Validation	Stop Validation	Deployment Wizard	Rack View	Circuit View
Undefined	Yes	--	--	--	--	Normal	Normal
Not Installed	--	--	--	--	--	Normal	Normal
Installed No Power	--	--	--	--	--	Normal	Normal
Not Provisioned	Yes	--	--	--	--	Normal	Normal
Provisioned	Yes	--	--	--	--	Normal	Normal
Unsupported	Yes	--	--	--	--	Normal	Normal
Agent Ready	Yes	Yes	--	--	**Yes**	Normal	Normal
Validation Ready	Yes	Yes	Yes	--	--	Normal	Normal
Validating	Yes	--	Yes	Yes	--	Normal	Normal
Validation Only	Yes	--	Yes	Yes	--	Normal	Normal
Production Ready	Yes	Yes	Yes	--	--	Normal	Normal
Production	Yes	--	--	--	--	Normal	**Hidden**
Drain / Troubleshooting	Yes	Yes	Yes	--	--	Normal	Normal
Maintenance	--	--	--	--	--	Normal	Normal
Unmanaged	--	--	--	--	--	**Grayed**	Normal

Key takeaways:

- Devices that are Not Installed, Installed No Power, Maintenance, or Unmanaged are not pinged at all.
- Only Agent Ready devices appear in the deployment wizard.
- Production devices are hidden from the circuit view to reduce noise.
- Unmanaged devices appear grayed out in the rack view.

7.3.1.5 Managing Device Status in the UI

7.3.1.5.1 Viewing Device Status

1. Open the CVT web UI and navigate to the Devices view.
2. The Lifecycle column displays a color-coded badge for each device:

Badge Color	Statuses
Green	Validation Ready, Validating, Validation Only, Production Ready, Production
Light Green	Provisioned, Agent Ready
Orange	Not Provisioned
Gray	Unmanaged, Not Installed, Installed No Power, Unsupported
Blue	Drain / Troubleshooting, Maintenance
Light Gray	Undefined

3. Click any device row to open the detail panel on the right side.

7.3.1.5.2 Changing the Status of a Single Device

1. Select a device in the Devices view by clicking its row.
2. Click the Set Status button in the toolbar (admin-only), or right-click the device and choose Set Device Status.
3. The modal shows the device name and its current status badge.
4. Choose a new status from the dropdown. Only transitions that are valid from the device's current status are listed.
5. Optionally enter a reason describing why you are making this change.
6. Click Apply.

The devices table refreshes automatically to show the updated status.

7.3.1.5.3 Changing the Status of Multiple Devices (Bulk Update)

1. Select multiple devices using Ctrl+click or Shift+click. The detail panel on the right is hidden during multi-select.
2. Click Set Status.
3. The bulk modal shows the list of selected devices and a dropdown listing all statuses. (CVT validates each device individually on the backend.)
4. Optionally enter a reason.
5. Click Apply.
6. A results summary appears showing:
    - Number of successful updates.

- Number of failures.
- Per-device error details for any transitions that were rejected.

7.3.1.5.4 Viewing Status History

1. Select a device in the Devices view.
2. In the detail panel, click the Status History tab.
3. The history table shows every transition for that device:
    - Timestamp -- when the change occurred.
    - From / To -- previous and new status (displayed as badges).
    - Reason -- why the transition happened (e.g., "pre-check passed" or a user-provided reason).
    - Changed By -- `auto` for system-driven transitions, or the administrator's username for manual changes.

History is sorted newest-first.

7.3.1.6 Agent Deployment and Device Status

7.3.1.6.1 Standard Deployment

When you open the Deploy Agents dialog, only devices in Agent Ready status appear in the device list. After a successful deployment:

- The device transitions from Agent Ready to Validation Ready.

If deployment fails:

- The device transitions from Agent Ready back to Provisioned so that pre-checks can re-evaluate it on the next connectivity cycle.

7.3.1.6.2 Force Deployment

The Deploy Agents dialog includes a Force deploy checkbox (admin-only). When enabled, agent deployment proceeds for devices regardless of their current status. This is useful when you need to deploy agents on devices that have not yet been promoted to Agent Ready.

Force deploy still respects one safety rule: it will not redeploy on a device that already has the correct agent version installed.

7.3.1.7 Topology Upload and the Managed Column

7.3.1.7.1 Unified PTP Files

In Unified PTP files, the Managed column controls whether CVT manages each device. Switches with `Managed=False` are automatically initialized to Unmanaged status on topology load. The "Allow CVT to manage agents on servers" checkbox is not shown for Unified PTP uploads because management is controlled per-node in the file itself.

7.3.1.7.2 DOT Files and Legacy ETH PTP

For DOT file uploads and Legacy ETH PTP files (which lack a per-node Managed column), the "Allow CVT to manage agents on servers" checkbox remains available.

7.3.1.8 Automatic Transitions

CVT handles most status transitions automatically. The following sections describe the key automated workflows.

7.3.1.8.1 Connectivity Checks

After a topology is loaded, CVT periodically pings devices and runs pre-checks:

Event	Transition
Ping fails on a new device	Undefined → Not Installed
Ping succeeds, pre-checks pass	Undefined → Provisioned → Agent Ready
Ping succeeds, pre-checks fail	Undefined → Not Provisioned
Previously unreachable device becomes reachable and passes pre-checks	Not Installed → Not Provisioned → Provisioned → Agent Ready
Pre-checks fail on a Provisioned device	Provisioned → Not Provisioned

If the agent is already installed when pre-checks pass, CVT continues the chain through to Validation Ready.

7.3.1.8.2 Agent Deployment

Event	Transition
Agent deployed successfully	Agent Ready → Validation Ready
Agent deployment fails	Agent Ready → Provisioned
Agent uninstalled	Validation Ready → Provisioned

7.3.1.8.3 Validation Lifecycle

Event	Transition
Validation starts	Validation Ready → Validating
Validation stops	Validating → Validation Ready

7.3.1.8.4 Crash Recovery

If the CVT collector stops unexpectedly while devices are in the Validating state, those devices are automatically reset to Validation Ready on the next startup so that validation can be restarted cleanly.

7.3.1.9 Migration from Excluded Devices

If you are upgrading from a CVT version that used the `excluded_devices.txt` file to exclude devices, the migration happens automatically on first startup:

1. Devices listed in the file are set to Unmanaged status.
2. The file is renamed to `excluded_devices.txt.migrated`.
3. On subsequent restarts, the migration is skipped.

No manual action is required.

7.3.1.10 Common Workflows

7.3.1.10.1 Bringing a New Device into Production

1. Load the topology (or add the device to an existing topology).
2. Wait for CVT to ping the device and run pre-checks. The device progresses automatically: Undefined → Provisioned → Agent Ready.
3. Deploy the agent (the device appears in the deployment wizard). On success: Agent Ready → Validation Ready.
4. Start validation. The device moves to Validating.
5. Review validation results. When satisfied, use Set Status to promote the device to Production Ready.
6. When ready to put the device into service, promote it to Production.

7.3.1.10.2 Excluding a Device from CVT

1. Select the device in the Devices view.
2. Click Set Status and choose Unmanaged.
3. The device is immediately excluded from pinging, deployment, and validation. It appears grayed out in the rack view.

To bring the device back, set it to Not Installed, Not Provisioned, or Provisioned (depending on its actual state). CVT will re-evaluate it during the next connectivity cycle.

7.3.1.10.3 Putting a Device into Maintenance

1. Select the device and set its status to Maintenance.
2. CVT stops pinging, stops validation, and suppresses alarms for this device.
3. When maintenance is complete, set the device to Provisioned, Agent Ready, or Validation Ready depending on whether you want CVT to re-validate it.

7.3.1.10.4 Troubleshooting a Production Device

1. Set the production device to Drain / Troubleshooting.
2. CVT re-deploys the agent and starts validation so you can investigate the issue with full telemetry.

3. Once resolved, move the device back to Validation Ready or directly to Production Ready if no further validation is needed.

7.3.1.11 Tips and Best Practices

- Let CVT drive the early stages. Devices progress automatically from Undefined through Agent Ready. Manual intervention is only needed if something fails or if you want to skip steps using force deploy.
- Use the Maintenance status before performing hardware work on a device. This prevents false alarms and unnecessary agent activity.
- Review status history when investigating issues. The history tab provides a complete audit trail of every status change, including timestamps and reasons.
- Use bulk updates when managing large deployments. Select multiple devices and transition them in a single operation rather than one at a time.
- Validation Only is intended for devices where the agent is managed by an external system. CVT can run validation on these devices but will not attempt to install or remove the agent.

7.4 Hierarchical Anomalies

Cable Validation Tool (CVT) performs hierarchical anomaly detection on each monitored port using data collected from the AMBER (Advanced Monitoring of BER) telemetry. For each port, a series of health checks are evaluated in a fixed priority order. The first checkpoint that detects an issue is reported as the port's anomaly, and subsequent checks are skipped for that port. This ensures the most fundamental problem is surfaced first.

All anomalies detected by these checks are reported under the syndrome "Anomalous-port (Signal, Temperature)" in the Reports and Circuits views. The specific remediation action shown alongside the syndrome indicates which check was triggered.

The checkpoint priority order depends on the cable type:

Optical cables: EEPROM read > BER + RX power > Module temperature > Power threshold > Firmware opcode > Link state > Bad module

Copper cables: EEPROM read > BER > Module temperature > Module voltage > Firmware opcode > Link state > Bad module

NVLink cables: EEPROM read > BER > Module temperature > Module voltage > Firmware opcode > Link state

The table below lists all possible anomaly messages, the checks that produce them, and their meaning.

Anomaly Category	Message	Meaning
EEPROM Issue	Module read issue in sample. Check if module can be read correctly manually via mlxlink/ethtool -m, if not attempt to reseat the module.	Module is `plugged_enable` but critical EEPROM fields are unreadable — vendor name, vendor OUI, part number, serial number contain invalid/garbage characters, or module temperature is `N/A`, or (for optical only) RX/TX power lane values are `N/A`.

Anomaly Category	Message	Meaning
BER + Optical Check	Link Bit Error Rate is high, issue is confirmed on the optical line. Clean the optics, make sure Rx power readings per lane are within 2dBm of each other	BER grade is Marginal or worse AND one lane's Raw BER is 10x higher than the next, with its corresponding rx_power >2dBm lower than the best lane.
	Link Bit Error Rate is high. Clean the optics	BER grade is Marginal or worse, but the rx power variance check didn't flag (and SNR media check didn't flag).
	Rx power abnormal variance detected and is affecting the lane BER. Clean the optics	BER is fine, but one lane's rx_power is anomalously low relative to others (suspected bad optical side).
	Optical signal received on optical side is showing high noise levels. Clean the optics	SNR media lane check flagged
BER only Check	Link Bit Error Rate is high. Reseat the connector	BER grade is Marginal or worse on a copper or NVLink cable. (Q3400_RA IB XDR switch uses a different mechanism to determine BER issues as listed in the next table.)
Module Temperature	Module temperature lower than range. Check rack and system cooling	`module_temperature` < `temperature_low_th`.
	Module temperature exceeds range. Check rack and system cooling	`module_temperature` > `temperature_high_th`.
Optical Power Levels	No light observed on Rx/Tx. Verify connectivity and light levels on the optics and fiber	All valid lanes have TX or RX power at -40 dBm (no light), and the linkdown counter changed since last sample.
	Module power out of range. Clean the optics, or replace transceiver.	At least one valid lane has TX or RX power outside the **[low_th, high_th]** range.
Module Voltage	Module Voltage lower than range. Check power supply	`module_voltage` < `voltage_low_th`.
	Module Voltage exceeds range. Check power supply	`module_voltage` > `voltage_high_th`.
Firmware Status Opcode	`<status_message>`. Clean optics	Opcode **15** — firmware reports a signal integrity issue (the first part is the firmware's `status_message` field, e.g. "Bad signal integrity").
	<status_message>. Verify cabling on both sides of the link	Opcode **57** — firmware reports a cabling/connectivity issue.
	<status_message>. Verify cooling on device, and reseat transceiver.	Opcode **1030** — firmware reports a thermal/cooling issue.

Anomaly Category	Message	Meaning
	<status_message>. Reseat transceiver, if the issue remains; test known working transceiver in the same port and test the current transceiver in known working switch.	Opcodes **59, 1026, 1027, 1031, 1033, 1048** — module communication issue.
	<status_message>. Reseat the transceiver.	Any other non-valid opcode (not **0**, **1**, or **1024**) without a specific mapped action.
Link State	Link negotiation issue. Verify speed and FEC configuration on both ports in the link	`phy_manager_state` = **"Polling"** and module is not unplugged. Port is stuck negotiating.
	Module communication issue. Reseat the transceiver	`phy_manager_state` = **"Rx_disable"** and module is not unplugged.
Module Operational Status	Bad module in port. Verify the module was seated correctly, replace if needed.	`module_oper_status` is neither `plugged_enable` nor **unplugged** (e.g. plugged_disabled, module error states).

7.4.1 XDR BER Anomaly Handling

For Q3400_RA IB XDR switch BER anomaly reporting follows a different path than other protocols. It uses a two-phase BER evaluation with a sliding window mechanism to reduce false positives and provide severity-graded anomalies. This switch has 4 planes and generates one amber per plane. The grade is calculated per port and per plane. And a bad grade for any of the planes creates an anomaly for the port.

The two phases are determined by `time_since_last_clear`:

- Pre-staging phase (first 4 hours after counters are cleared): Used for validating newly installed or reseated hardware. Individual sample grades are evaluated directly — a single bad sample is enough to trigger an anomaly.
- On-going phase (after 4 hours): Uses a BER sliding window that accumulates sample grades over time. Anomalies are only raised when multiple bad samples are observed within the window, and severity escalates from Warning to Error based on the count and pattern of bad samples.

All XDR BER anomalies appear under the syndrome "Anomalous-port (Signal, Temperature)" in the UI, with the phase and severity indicated by a prefix in the remediation action text (**Pre-staging:**, **Warning:**, or **Error:**).

Stage	Message	Meaning
Pre-staging phase (time_since_last_clear <= 4 hours)	Pre-staging: Warning: Link down counter changed since last check.	Grade is **Marginal** — BER metrics are within thresholds, but the link-down counter changed since the last advanced stats snapshot.
	Pre-staging: Link Bit Error Rate is high. Reseat the connector.	Grade is **Poor** — one or more BER thresholds exceeded during pre-staging (symbol_errors > 0, effective_ber > 1e-12, first_zero_hist > 9, or link_error_recovery_counter > 0).
On-going phase (time_since_last_clear > 4 hours)	Warning: Link Bit Error Rate is high. Reseat the connector.	2-3 marked (bad BER) sampling windows within the current window duration.
	Error: Link Bit Error Rate is high. Reseat the connector.	4+ marked sampling windows, OR 2 consecutive windows at Warning severity. Persists until a full Good window completes.

7.4.1.1 Anomaly Clearing Behavior

During the on-going phase, a BER window spans 4 hours. A Good window is one with 0-1 marked (bad BER) samples. Once a port reaches Warning or Error severity, the anomaly persists until a full Good window completes. Mid-window, the previous window's severity carries forward even if no new bad samples have been observed yet. When a Good window completes, the history is cleared and the port returns to normal.

7.5 Bringup CLI

7.5.1 Running Bringup CLI

Running the bringup CLI can be done in two ways:

1. Direct Execution in the Container:

```
docker exec -it cables_bringup bringupcli
```

2. Alternatively, it is possible to run exec bash in the container and run the appropriate CLI command based on the fabric type from anywhere within the container:

- - Start a bash session in the container:

```
docker exec -it cables_bringup bash
```

  - Execute the desired CLI command:

```
# stop cvt service
supervisorct stop cvt-service
# run bringupcli
bringupcli -k
```

7.5.2 Bringupcli Usage

bringupcli may have command line arguments, see usage below for more details:

```
root@r-ufm65:/# bringupcli -h
usage: bringupcli \[-h\] \[-V\] \[-k\] \[-d\]
```

Optional Arguments

Argument	Description
`-h, --help`	Show this help message and exit
`-v, --version`	Show program version number and exit
`-k, --kill-other-sessions`	Kill other CLI sessions if existent
`-d, --daemon`	Run as daemon

7.5.3 Bringupcli Commands

Bringup CLI commands are no longer required to operate CVT. All operations are available through the Web UI, which is the recommended way to use CVT.

7.5.3.1 Load Topology Commands

7.5.3.1.1 load_topo

Description: Loads topo file if the fabric is InfiniBand, and dot file if the fabric is Ethernet.

```
load_topo <filename> dns=<true|false> cluster=<cluster name >
```

Parameters:

- `filename` : absolute path for the topo/dot file
- `dns` (Optional): assumes that DNS is active, and you can access the switches by hostnames by default dns=true.
- `cluster` (Optional): cluster name, if cluster name is provided it will be set to the provided value, else it will be set to 'default'.

7.5.3.1.2 load_ptp

Description: Loads PTP topology file (Excel file).

```
 load_ptp <filename> format=<legacy_ib|legacy_eth|unified_topo> dns=<true|false> cluster=<cluster name>
sheets=<comma separated sheets> dc_layout=<file path> hca_mapping=<file path>
```

Parameters:

- **filename** : The absolute path for the P2P file.
- **format** : The format of ptp is of legacy or unified topology:
  - legacy_ib: allows to load legacy ib ptp file
  - legacy_eth: allows to load legacy eth ptp file
  - unified_topo: allows to load the unified ptp file which supports IB, eth, xdr and nvlink protocols
- **dns** (Optional): Assumes that DNS is active, and you can access the switches by hostnames by default **dns=true** .
- **cluster** (Optional): Cluster name, *if cluster name is provided it will be set to the provided value, else it will be set to 'default'.*
- **sheets** (Optional): comma separated sheets name. If provided, only the specified sheets from the Excel file are loaded. If not provided, all sheets in the file are loaded.
- **dc_layout** (Optional, Ethernet Fabric Only): A CSV file that describes the data center layout, for more information please find DC Floor Layout File.
- **hca_mapping** (Optional, InfiniBand and NVOS Fabrics Only): A CSV file that defines the relationship between port numbers and HCA names, for more information please find HCA Mapping File.

7.5.3.1.3 load_ip

Description: Loads switch IP addresses, can be used if DNS is inactive.

```
load_ip <filename> cluster=<cluster name >
```

Loads the IP/switch-name mapping, to allow reaching the switch via REST API to retrieve local topology, GUID, etc. The file format is pairs of IP addresses and hostname. This file will be used in association with a 'topo' file in case DNS is unavailable.
An IP file example:

```
A comment
10.0.30 switch1
10.0.0.31 switch2
```

Parameters:

- **filename** : The absolute path for the IP file.
- **cluster** (Optional): The cluster name, *if cluster name is provided it will be set to the provided value, else it will be set to 'default'.*

7.5.3.1.4 load

Description: Loads both IP addresses and topo files.

```
load <topo filename> <ip filename> cluster=<cluster name>
```

Loads the .topo and .ip files.

> ⚠ Note: if you have multiple files describing a topology, use the commands:
>
> **load_ip file.ip**
>
> **load_topo file1.topo file2.topo file3.topo**

Parameters:

- **topo filename** : The absolute path for the topology file directory
- **ip filename** : The absolute path for the IP file directory
- **cluster** (Optional): The cluster name, *if cluster name is provided it will be set to the provided value, else it will be set to 'default'.*

7.5.3.1.5 load_clusters

> ⚠ Note: will be deprecated in coming release

Clusters file should have the following format, where topo file should be in xlsx format and the IP file is optional.

```
load_clusters <filename> dns=true|false
```

```
cluster_name, topo_file, ip_file
CLUSTER1, cluster1_topo.xlsx, cluster1.ip
CLUSTER2, cluster2_topo.xlsx,
```

Parameters:

- **filename** : The absolute path for the cluster file
- **dns** (optional): Assumes that DNS is active, and you can access the switches by hostnames by default dns=true.

7.5.3.2 Validations Commands

7.5.3.2.1 show_clusters

Description: Show list of loaded clusters as loaded from the clusters file.

```
show_clusters
```

7.5.3.2.2 show_switches

Description: Show list of loaded switches as loaded from the topology file

```
show_switches cluster=<cluster_name>
```

Parameters:

- **cluster** (Optional): cluster name, If the cluster name is provided, show the switch in the given cluster only.

Output Example:

```
MQM8700 sw-hdr-proton01
-----------------------
MQM8700 sw-hdr-proton01 P3 --> swx-proton03 mlx5_0 P1
MQM8700 sw-hdr-proton01 P4 --> swx-proton04 mlx5_2 P1
MQM8700 ufm-sw-hdr01
--------------------
MQM8700 ufm-sw-hdr01 P1 --> ufm-sw-hdr02 P1
MQM8700 ufm-sw-hdr02
--------------------
MQM8700 ufm-sw-hdr02 P1 --> ufm-sw-hdr01 P1
```

7.5.3.2.3 check_switch_status

Description: Check switch connectivity status (Ping/JSON-API/Agent)

```
check_switch_status cluster=<cluster name >
```

Parameters:

- **cluster** (Optional): cluster name, If the cluster is provided, the check will be done for the switches in the provided cluster only.

Output Example:

```
Host IP ping JSONAPI Agent
---------------------------- ------------- ---- ---- -----
sw-hdr-proton01.mtr.labs.mlnx 209.44.74 True True True
ufm-sw-hdr01.mtr.labs.mlnx 10.209.36.113 True True True
ufm-sw-hdr02.mtr.labs.mlnx 10.209.36.122 True True True
```

7.5.3.2.4 start_validation

Description: Push topology to switches and get validation reports.

```
start_validation timeout=<n> cluster=<cluster_name>
```

Parameters:

- **cluster** (Optional): cluster name, If the cluster is provided, the validation will be started in the switches in the provided cluster only.

- **timeout** (Optional): timeout in which validation stops, n is in seconds (s), minutes (m), hours (h) or days (d). For example timeout=20m or timeout=2h.

If timeout is not provided, use the **stop_validation** command to stop it.

7.5.3.2.5 stop_validation

Description: Stops validation routine. Unsubscribe from getting switches updates.

```
stop_validation
```

7.5.3.3 Troubleshooting

	Description	Example
deploy_single_agent	Deploys agent on a specific node will be deprecated soon. use **deploy_agents** instead.	**deploy_single_agent <switch-ip/host-ip>**
deploy_all_agents	Deploys agents on loaded nodes that have no agents. will be deprecated soon. use **deploy_agents** instead.	**deploy_all_agents**
deploy_agents	Deploys agents on all or specific nodes.	**deploy_agents all** **deploy_agents node_ip1 node_ip2**
remove_all_agents	Removes agents from loaded nodes that have agents. will be deprecated soon. use **remove_agents** instead.	**remove_all_agents**
remove_single_agent	Removes an agent from a specific node. will be deprecated soon. use **remove_agents** instead.	**remove_single_agent <switch-ip/host-ip>**
remove_agents	Remove agents on from all or specific nodes.	**remove_agents all** **remove_agents node_ip1 node_ip2**

7.5.3.4 Set Credentials Commands

7.5.3.4.1 show_creds

Description:Display the credentials for all nodes

```
show_creds [format=json|report]
```

These credentials are used for communication with switches and hosts.
Parameters:

- **format** : the format for displayed data .

7.5.3.4.2 set_default_creds

Description: Sets the default switch/host credentials to override the built-in default credentials.

```
set_default_creds user=<user> pwd=<pwd> type=<switch|host> save=<true|false>
```

These credentials are used for communication with any switch that does not have specific credentials.
Parameters:

- **user** : user name.
- **pwd** : password.
- **type** (Optional): the default value is switch
- **save** (Optional): If save it set to true (default: true), credentials will be saved encrypted to a file

7.5.3.4.3 set_node_creds

Description: Sets the credentials for a specific switch/host, it can be used when the switch credentials are different than the defaults.

```
set_node_creds <switch> user=<user> pwd=<pwd> save=true|false
```

Parameters:

- **switch** : switch name
- **user** : user name.
- **pwd** : password.
- **save** (Optional): If save it set to true (default: true), credentials will be saved encrypted to a file

7.5.3.4.4 remove_node_creds

Description: remove the credentials for a specific switch/host.

```
remove_node_creds <switch>
```

Parameters:

- **switch** : switch name

7.5.3.4.5 set_credential_profile

Description: Sets the credentials for a credential profile name specified in unified topology ptp file.

```
set_credential_profile <credential_profile_name> user=<user> pwd=<pwd> [save=true|false]
```

Parameters:

- **credential_profile_name** : Credential Profile name.
- **user** : user name.
- **pwd** : password.
- **save** (Optional): If save it set to true (default: true), credentials will be saved encrypted to a file.

7.5.3.4.6 remove_credential_profile

Description: remove the credentials associated to a credential profile.

```
remove_credential_profile <credential_profile_name>
```

Parameters:

- **credential_profile_name** : Credential Profile name

7.5.3.5 Web User Commands

7.5.3.5.1 add_web_login

Description: Add new users to login to the web gui apart from the default 'admin' login.

```
add_web_login user=<user> pwd=<pwd> account_type=<account_type>
```

Parameters:

- **user** : username
- **pwd** : password
- **account_type** : Account_type can be cabler, admin, nvidia, or developer

7.5.3.5.2 delete_web_user

Description: Delete a web user account.

```
delete_web_user user=<user>
```

Parameters:

- **user** : username.

7.5.3.5.3 show_web_users

Description: Shows the web users added along with their account types.

```
show_web_users
```

Example output:

```
admin: admin
nUser: nvidia
dUser: developer
```

7.5.3.5.4 update_web_user

Description: Update the specified web user, allowing changes to the password, account type, or both.

```
update_web_user user=<user> pwd=<pwd> account_type=<account_type>
```

Parameters:

- `user` : username
- `pwd` : password
- `account_type` : Account_type can be cabler, admin, nvidia, or developer

7.5.3.6 Other Commands

7.5.3.6.1 show_switch_history

Description: Lists data files collected from switches in the last days

```
show_switch_history <switch names> past=<n> start_time=<n> end_time=<n> prev=<n>
```

Parameters:

- `switch_names` (Optional): a space delimited switch names, if no switches are provided, it will bring data of all switches
- `past` (Optional): Past argument can be used to specify the history interval, by default it is set to one week past=1w.
- `start_time` , `end_time` (Optional): time period to look for data, if no time is provided, data of the last week will be provided
- `prev` (Optional): retrieve data during the previous period from specified time to now. 'prev' formats: num[day|week]

7.5.3.6.2 amber_show_latest

Description: Shows latest collected amber data from switches

```
amber_show_latest
```

7.5.3.6.3 show_incoming_reports

Description: show or hide incoming reports

```
show_incoming_reports show=true/false
```

Parameters:

- `show` (Optional): the default value is true

7.5.3.6.4 show_statistics

Description: Shows a summary for all issues

```
show_statistics
2024-11-21 12:58:46.726351 - show statistics for default cluster:
Status Number of occurrences Nodes affected
------------------------------------ --------------------- --------------
No Transceiver 0 0
Link Down, No signal 0 0
Admin Down 0 0
ErrDisable - Flap 0 0
ErrDisable - Rx 0 0
Negotiation fail 0 0
Wrong-neighbor 0 0
Wrong-port 0 0
Unknown-neighbor 0 0
Extra-cable 0 0
Underperforming-link (BER) 0 0
Flapping-link 0 0
Anomalous-port (Signal, Temperature) 0 0
Unreachable-device 0 0
Correctly-wired 0
No report
```

7.5.3.6.5 show_switches

Description: Shows the current loaded switches

```
show_switches
```

Example output:

```
MQM8700 sw-hdr-proton01, rack: PXX, unit: 30
---------------------------------------------
MQM8700 sw-hdr-proton01 P3 --> swx-proton03 mlx5_0 P1
MQM8700 sw-hdr-proton01 P4 --> swx-proton04 mlx5_2 P1
```

7.5.3.6.6 add_certificate

Description: Updates the SSL certificate file used by Apache for secure connections

```
add_certificate <crt file> <key_file>
```

The provided file should be a valid SSL certificate file in crt format. The old certificate file will be backed up before replacing it with the new one.

7.5.3.6.7 version

Description: Shows application version.

7.5.3.6.8 exit

Description: Exits the application.

7.5.3.6.9 help

Description: Shows a list of commands. For help on a specific command, run help <command>

7.5.4 Tool Initializing Using Bringup CLI

To initialize the tool, perform the following:

1. Run bringupcli
2. Open bringup GUI
3. Load the fabric topology file using one of the Load Topology Commands or from the bringup GUI
4. Set the credentials for the switches.
5. Deploy the agent on all switches.

> ⚠ **wget** or **curl** should be installed on the switches and hosts as it is used to download the agent image from the collector

6. Start validation using GUI or using the Validation Command.

7.5.5 Running bringup GUI

1. Open the following URL in the browser: https://<bringup_machine_ip>/cables_validation
2. Enter credentials in the login page.
3. You may change the default self signed certificate located by default in the container at:

```
SSLCertificateFile ${BRINGUP_CONF_APACHE_PATH}/certs/cv-cert.crt
SSLCertificateKeyFile ${BRINGUP_CONF_APACHE_PATH}/private/cv-cert.key
```

7.6 CVT Configuration

7.6.1 CVT Environment Configuration (`cvt_env.conf`)

This document outlines the configuration options available in the **cvt_env.conf** file, which controls various aspects of the Cable Validation Tool (CVT) collector and agent behavior.
The **cvt_env.conf** file is located in the shared volume under **<cable_bringup_root>/config/cvt_env.conf**.

7.6.1.1 Configuring CVT

CVT can be configured through two primary methods:

1. Environment Variables: Setting system-level environment variables.
2. Configuration File (`cvt_env.conf`): Defining variables within the `cvt_env.conf` file.

7.6.1.1.1 Precedence

When a configuration option is defined in both the environment variables and the `cvt_env.conf` file, environment variables take precedence over the values specified in the configuration file.

For example, if `AGENT_COLLECT_INTERVAL` is set to `300` in your shell's environment variables and to `600` in `cvt_env.conf` , CVT will use `300` as the collection interval.

7.6.1.1.2 Changing Configuration

- Via Configuration File: To apply changes made to the `cvt_env.conf` file, you must edit the file and then restart the CVT service: `supervisorctl restart cvt-service`
- Via Environment Variables: To apply changes using environment variables, you need to set the environment variables in the Docker container run command when launching the CVT service.
    - Example: `docker run --name cables_bringup -itd --env VAR1=value1 --env VAR2=value2 --network=host mellanox/cables_bringup:<version>`

7.6.1.2 Network Configuration

Variable	Description	Default
`AGENTS_COLLECTOR_NAT_IP`	Collector External (NAT) IP address. Define this if there is a NAT between the collector and the agents. This IP is used by agents to fetch images and send data/reports.	Empty (no NAT configured)
`AGENTS_COLLECTOR_NAT_PORT`	NAT port is used for port-forwarding from K8s or other intermediate networks to the collector. If the port is empty, CVT will use the https port of the collector (APACHE_HTTPS_PORT) or default 443	Empty (default is APACHE_HTTPS_PORT or 443)
`DEFAULT_AGENTS_INTERFACE_NAME`	Interface name of the collector over which all agents (switch and host) communicate. The IP of this interface is used by agents to fetch images and send data/reports.	Empty (IP of default interface is used)
`HOST_SPECIFIC_INTERFACE_NAME`	Use if a different interface is desired for host agents, separate from `DEFAULT_AGENTS_INTERFACE_NAME` . The IP of this interface is used by host agents to communicate with the collector.	Empty (uses same interface as `DEFAULT_AGENTS_INTERFACE_NAME`)
`PING_COUNT`	Count for ICMP reachability for collector-side connectivity checks (Linux ping: -c count, -W seconds per probe).	1 count

Variable	Description	Default
PING_TIMEOUT_SEC	Timeout for ICMP reachability for collector-side connectivity checks (Linux ping: -c count, -W seconds per probe).	3 sec

7.6.1.3 Agent Configuration

Variable	Description	Value / Unit	Default	Constraints
CV_DOT_IN_HOSTNAME	Set to true if the switch hostname contains a dot (other than the domain part).	Boolean	Empty (implies false)	—
FULL_REPORT_PUBLISH_INTERVAL_MINUTES	Time after which a full report is forced to be published.	Minutes	720 (12 hours)	Interval less than 10 minutes is not supported
AMBER_PUBLISH_EACH_ITERATION	Set to true to publish amber data on each agent iteration, regardless of changes.	Boolean	false	—
AGENT_COLLECT_INTERVAL	Agent data collection interval. Controls how often the agent collects and processes port/link data.	Seconds	600 (10 minutes)	—
NVLINK_VALIDATION	Enable/Disable nvlink addition and validation for gb200/300 nodes.	Boolean	true	—
CUSTOM_AMBER_FIELDS	Allows users to dynamically specify additional raw amber fields or processed agent attributes to be captured and reported without requiring any modifications to the core source code. Example: CUSTOM_AMBER_FIELDS = transceiver_reinsert_cnt, transceiver_swap_cnt, Advanced_Status_Opcode	Comma separated strings	Empty	

Variable	Description	Value / Unit	Default	Constraints
`CHECK_PORTS_INTERVAL`	Interval to check ports in seconds (default 10 seconds). This controls how often the agent checks the ports for changes.	Seconds	10 sec	
`MAX_AUTH_FAILURE_INTERVAL`	Maximum interval between authentication failures in seconds (default 10 minutes) interval starts from 10 seconds and doubles every failure, till it reaches the maximum interval	Seconds	600 (10 minutes)	
`GB_RACK_COMPUTE_PRODUCT_NAME_REGEX`	Regex matched against dmidecode Product Name to detect GPU compute platforms (GB200, GB300, Dell GB200 trays, etc.). When matched, GPU support is enabled during agent install and runtime (nvidia-smi, docker GPU flags, NVLink ports, cable cartridge collection).	Regex	(GB[23]00|Dell Server 9712a)	

7.6.1.4 Collector Configuration

Variable	Description	Value / Unit	Default	Constraints / Notes
`MAX_INACTIVE_INTERVAL`	Maximum time to wait for an agent to become inactive.	Minutes	15	—

Variable	Description	Value / Unit	Default	Constraints / Notes
`CHECK_NEW_SWITCHES_INTERVAL`	Interval to check for new switches.	Minutes	15	—
`START_VALIDATION_TIMEOUT`	Time to wait for an agent to become active after starting validation.	Minutes	5	—
`WAIT_TIME_INACTIVE_AGENTS`	Time to wait for an agent to become inactive.	Minutes	1	—
`CVT_MAX_WORKERS`	Maximum number of workers to run in parallel for general operations (validation, connectivity, DNS).	—	30	—
`CVT_DEPLOYMENT_MAX_WORKERS`	Maximum number of workers for agent deployment (limited due to image transfers).	—	30	—
`CVT_QUICK_TIMEOUT`	Quick timeout for unreachable devices, reducing wait time for failed connections.	Seconds	3	—
`AGENT_COMM_TIMEOUT`	Agent communication timeout for individual HTTP requests to agents.	Seconds	30	—

Variable	Description	Value / Unit	Default	Constraints / Notes
`CVT_BATCHING_THRESHOLD`	Batching threshold; batching is only used for deployments larger than this value to reduce overhead.	—	5000	—
`CVT_BATCH_SIZE`	Batch size when batching is used (devices per batch).	—	1000	—
`CVT_DNS_RES_OPTIONS`	DNS resolver options for fast timeouts to avoid long waits on unresolvable hostnames. Configures system resolver behavior when `load_topo` performs parallel DNS resolution.	Format: `timeout:X attempts:Y single-request`	`timeout:1 attempts:1 single-request`	`timeout:X` : Seconds to wait per DNS query; `attempts:Y` : Number of retry attempts; `single-request` : Send A and AAAA queries separately to improve performance

7.6.1.5 SSH Configuration

More details on this can be found at Agent Deployment and SSH Configuration

Variable	Description	Value / Unit	Default
`CV_SSH_KEY_FILE`	SSH private key file path for passwordless authentication to **HOST** devices only. SSH keys are **not** used for switch devices. Must be accessible inside the collector container. Leave empty to use standard SSH key discovery.	Path	Empty (uses standard SSH key discovery)
`SSH_CONN_TIMEOUT`	SSH connection timeout for both SSH command execution and SFTP file transfers. Increase for slow networks, decrease for faster failure detection.	Seconds	20
`SSH_LOOK_FOR_KEYS`	Enable automatic SSH key discovery from SSH agent and default locations. Applies only to HOST devices when no password is provided.	Boolean	true

7.6.1.6 Apache Configuration

More details on this can be found at Apache Configuration for CVT Collector

Variable	Description	Value / Unit	Default
`CVT_APACHE_PROFILE`	Apache performance profile based on server resources and cluster scale. This controls Apache's concurrency, connection handling, and optimization settings.	small, medium, large, xlarge, auto	small
`CVT_APACHE_LOG_LEVEL`	Log level for Apache error logs. Higher levels reduce log volume.	debug, info, notice, warn, error, crit, alert, emerg	warn
`CVT_APACHE_ACCESS_LOG`	Enable or disable Apache access logs (request logging). Set to 'false' to disable access logs and reduce disk I/O on high-traffic deployments.	Boolean	true

7.6.1.7 Application Configuration

Variable	Description	Options / Value	Default
`STARTUP_TOPOLOGY`	Topology loading at startup. Specifies which topology file to load.	`none` : Do not load any file `last` : Load the last loaded topology from history `<path>` : Load a specific topology file (`.topo` , `.dot` , `.xlsx` , `.json`)	none
`AUTO_START_VALIDATION`	Automatically start validation if a topology file is loaded.	Boolean	false
`STARTUP_CREDENTIALS_FILE`	credentials file to load at startup	none:no credentials will be loaded. <path>: load the credentials file (json, ini, cfg)	

7.6.1.8 Data Management

Variable	Description	Value/Unit	Default
`ENABLE_STATS_POLLING`	Set to true to enable polling for stats from the CVT collector.	Boolean	false

Variable	Description	Value/Unit	Default
`CVT_DATA_ARCHIVE_SCHEDULE`	Cron schedule for data directory archiving	minute hour day month weekday	5 0 * * * (daily at 00:05 UTC)
`CVT_MAX_PLAIN_DATA_DIRS`	Maximum number of plain (uncompressed) data directories to keep. Older directories are compressed to .tgz archives	Integer	7
`CVT_MAX_DATA_DIRS`	Maximum total data entries (plain + archived) to keep. Oldest archives are deleted when this limit is exceeded	Integer	30

7.6.1.9 Database Management

Variable	Description	Value/Unit	Default
`CVT_CIRCUIT_HISTORY_MAX_AGE_DAYS`	circuit_history table: delete rows older than this many days. Set to 0 to disable age-based pruning.	Days	14
`CVT_CIRCUIT_HISTORY_MAX_RECORDS`	circuit_history table: maximum rows to keep; oldest deleted when exceeded. Maximum is 200000. Set to 0 to disable.	Records	100000

7.7 Update Certificate

To update a certificate, run the following command:

```
add_certificate <crt file> <key_file>: update the ssl certificate
```

7.8 Apache Configuration for CVT Collector

7.8.1 Apache Configuration for CVT Collector

This document describes the configurable Apache settings for the CVT Collector, designed to optimize performance for different cluster scales.

7.8.1.1 Overview

The CVT Collector uses Apache as its web server. Apache performance can be tuned via profiles that adjust concurrency, connection handling, and logging settings based on your deployment scale.

7.8.1.2 Configuration Options

All Apache settings are configured in `cvt_env.conf` under the `[apache]` section:

```
[apache]
```

```
CVT_APACHE_PROFILE = small
CVT_APACHE_LOG_LEVEL = warn
CVT_APACHE_ACCESS_LOG = true
```

7.8.1.2.1 CVT_APACHE_PROFILE

Selects the Apache performance profile. Each profile includes MPM (Multi-Processing Module) settings and connection optimization settings.

Profile	Cores	RAM	Agents	MaxRequestWorkers
`small`	2-16	8-64GB	< 1,000	150
`medium`	32-128	128-512GB	1,000-10,000	2,048
`large`	200-300	1TB+	10,000-20,000	12,800
`xlarge`	400+	1.5TB+	20,000+	25,600
`auto`	-	-	-	Auto-detect based on CPU cores

Default: `small`

7.8.1.2.2 CVT_APACHE_LOG_LEVEL

Controls the verbosity of Apache error logs. Higher levels reduce log volume.

Level	Description
`debug`	Debug messages (very verbose)
`info`	Informational messages
`notice`	Normal but significant conditions
`warn`	Warning conditions (default)
`error`	Error conditions
`crit`	Critical conditions
`alert`	Action must be taken immediately
`emerg`	System is unusable

Default: `warn`

Recommendation for large clusters: Use `error` or `crit` to reduce log I/O.

7.8.1.2.3 CVT_APACHE_ACCESS_LOG

Enable or disable Apache access logs (HTTP request logging).

Value	Description
`true`	Access logs enabled (default)
`false`	Access logs disabled

Default: `true`

Recommendation for large clusters: Set to `false` to reduce disk I/O.

7.8.1.3 Profile Details

7.8.1.3.1 Small Profile (Default)

For development and small production deployments.

```
# MPM Settings
StartServers          2
MinSpareThreads      25
MaxSpareThreads      75
ThreadLimit          64
ThreadsPerChild      25
MaxRequestWorkers   150
# Connection Settings
KeepAlive On
KeepAliveTimeout 5
MaxKeepAliveRequests 100
Timeout 180
```

7.8.1.3.2 Medium Profile

For mid-size production clusters.

```
# MPM Settings
StartServers                 4
MinSpareThreads             75
MaxSpareThreads            250
ThreadLimit                 64
ThreadsPerChild             64
ServerLimit                 32
MaxRequestWorkers         2048
AsyncRequestWorkerFactor     2


# Connection Settings
KeepAlive On
KeepAliveTimeout 3
MaxKeepAliveRequests 500
Timeout 150
EnableMMAP Off
```

7.8.1.3.3 Large Profile

For large-scale production deployments.

```
# MPM Settings
StartServers                 8
MinSpareThreads            200
MaxSpareThreads            600
ThreadLimit                128
ThreadsPerChild            128
ServerLimit                100
MaxRequestWorkers        12800
AsyncRequestWorkerFactor     2


# Connection Settings
KeepAlive On
KeepAliveTimeout 2
MaxKeepAliveRequests 1000
```

```
Timeout 120
EnableMMAP Off
```

7.8.1.3.4 XLarge Profile

For very large-scale hyperscale deployments.

```
# MPM Settings
StartServers                    8
MinSpareThreads               250
MaxSpareThreads               750
ThreadLimit                   128
ThreadsPerChild               128
ServerLimit                   200
MaxRequestWorkers           25600
AsyncRequestWorkerFactor        2

# Connection Settings
KeepAlive On
KeepAliveTimeout 2
MaxKeepAliveRequests 1000
Timeout 120
EnableMMAP Off
```

7.8.1.4 Usage

7.8.1.4.1 Via cvt_env.conf (Recommended)

Edit the configuration file (persisted across restarts):

```
[apache]
CVT_APACHE_PROFILE = large
CVT_APACHE_LOG_LEVEL = error
CVT_APACHE_ACCESS_LOG = false
```

The configuration file is located at:

- Container: `/cable_bringup_root/config/cvt_env.conf`
- Default: `/etc/cablevalidation/cvt_env.conf`

7.8.1.4.2 Via Environment Variables

Set environment variables when starting the container:

```
docker run \
   -e CVT_APACHE_PROFILE=large \
   -e CVT_APACHE_LOG_LEVEL=error \
   -e CVT_APACHE_ACCESS_LOG=false \
   ...
```

Environment variables take precedence over cvt_env.conf settings.

7.8.1.4.3 Auto-Detection

Use **`auto`** to automatically select a profile based on available CPU cores: **`CVT_APACHE_PROFILE = auto`**

Auto-detection thresholds:

- 400+ cores → **`xlarge`**
- 200-399 cores → **`large`**

- 32-199 cores → **medium**
- < 32 cores → **small**

7.8.1.5 Configuration Files

The Apache configuration is applied at container startup via **apache2_wrapper.sh** :

Setting Type	Destination File
MPM settings	`/etc/apache2/mods-available/mpm_event.conf`
Performance settings	`/etc/apache2/conf-available/cvt_performance.conf`
Log level	`/etc/apache2/apache2.conf`
Access logs	`/etc/apache2/conf-available/*.conf` `/etc/apache2/sites-available/*.conf`

7.8.1.6 Customizing Profiles

Profiles are defined in `/etc/apache2/apache_profiles.conf` (copied from `conf/collector/apache_profiles.conf` at build time).

To add a custom profile:

1. Edit the profiles file
2. Add a new section with MPM and Performance settings:

```
[custom]
# custom - Description
# === MPM ===
StartServers 6
MinSpareThreads 100
...
# === Performance ===
KeepAlive On
KeepAliveTimeout 4
...
```

3. Set CVT_APACHE_PROFILE = custom in `cvt_env.conf`

7.8.1.7 Troubleshooting

7.8.1.7.1 Apache fails to start with "already running"

This occurs when a stale PID file exists from a previous unclean shutdown. The wrapper script automatically cleans stale PID files, but if the issue persists:

```
rm -f /cable_bringup_root/config/run/apache2/apache2.pid
supervisorctl start apache
```

7.8.1.7.2 Check current configuration

```
# Verify profile was applied
cat /etc/apache2/mods-available/mpm_event.conf
cat /etc/apache2/conf-available/cvt_performance.conf
```

```
# Check Apache config syntax
apache2ctl configtest


# View current settings
apache2ctl -V
```

7.8.1.7.2.1 View Apache logs

```
# Error log
tail -f /cable_bringup_root/log/apache2/error.log

# Access log (if enabled)
tail -f /cable_bringup_root/log/apache2/access.log
```

7.8.1.8 Best Practices for Large Clusters

1. Use appropriate profile: Match your hardware resources
2. Disable access logs: Reduces disk I/O significantly
3. Increase log level: Use `error` or `crit` instead of `warn`
4. Monitor resources: Watch CPU and memory usage after profile changes
5. Test before production: Validate configuration in staging environment

Example high-performance configuration:

```
[apache]
CVT_APACHE_PROFILE = large
CVT_APACHE_LOG_LEVEL =crit
CVT_APACHE_ACCESS_LOG = false
```

7.9 Agent Deployment and SSH Configuration

7.9.1 SSH Configuration and Usage in Agent Deployment/ Uninstall

The Cable Validation Tool (CVT) uses SSH for deploying and managing agents on Linux-based devices (hosts and switches). This document provides verification and QA teams with comprehensive guidance on SSH configuration, testing procedures, troubleshooting, and validation criteria for agent deployment and uninstall operations.

7.9.1.1 SSH Architecture

7.9.1.1.1 System Components

The CVT system uses multiple SSH components to handle different types of device connections:

1. SSH Connection Management
    - Base SSH client for establishing secure connections
    - Linux-specific SSH client for command execution on hosts and Linux switches
    - SFTP client for secure file transfers during deployment
    - Specialized client for MLNX-OS switch communication
2. Agent Deployment System
    - Linux agent deployment handler for hosts and Linux switches

- MLNX-OS agent deployment handler for Mellanox switches

7.9.1.1.2 Device Support Matrix

Device Type	OS Type	SSH Usage	Authentication
Host	Linux	SSH + SFTP	Password or SSH Keys
Switch	Linux (Cumulus, NVOS)	SSH + SFTP	Password (required)
Switch	MLNX-OS	JSON API (not SSH)	Password only

Important Notes:

- SSH is NOT used for MLNX-OS switches - they use JSON API over HTTP/HTTPS
- For switches (including Linux switches), password authentication is **required** as the agent uses these credentials to communicate with the switch for port information retrieval
- Supported Linux switch operating systems: Cumulus Linux, NVOS (for NVLink and XDR switches)

7.9.1.2 SSH Configuration

7.9.1.2.1 Environment Variables

Configure SSH behavior using these environment variables in **/etc/cablevalidation/cvt_env.conf**:

```
[ssh]
# SSH private key file path for HOST devices only
# NOTE: SSH keys are NOT used for switch devices (switches require password authentication)
# Switch passwords are mandatory as agents use them to communicate with switches for port information
# Path must be accessible inside the collector container
CV_SSH_KEY_FILE=

# SSH connection timeout in seconds (default: 20)
# Applied to both SSH commands and SFTP transfers
SSH_CONN_TIMEOUT=20

# Enable automatic SSH key discovery (default: true)
# Only applies to HOST devices when no password is provided
# Searches: SSH agent, ~/.ssh/id_rsa, ~/.ssh/id_dsa, ~/.ssh/id_ecdsa, ~/.ssh/id_ed25519
SSH_LOOK_FOR_KEYS=true
```

7.9.1.2.2 Key Configuration Details

1. CV_SSH_KEY_FILE
    - Only used for HOST devices (switches always require passwords)
    - Must be a container-accessible path
    - Leave empty for automatic key discovery
    - Switch devices cannot use SSH keys due to agent communication requirements
2. SSH_CONN_TIMEOUT
    - Connection timeout in seconds
    - Applies to both SSH commands and SFTP transfers
    - Increase for slow networks, decrease for faster failure detection
3. SSH_LOOK_FOR_KEYS
    - Only affects HOST devices (not applicable to switches)
    - When enabled, searches standard SSH key locations

- Only used when no password is provided for hosts

7.9.1.3 Authentication Methods

7.9.1.3.1 1. Password Authentication

- Supported: All Linux devices (hosts and switches)
- Configuration: Set credentials using CVT credential management
- Usage: MANDATORY for all switches (required for agent communication with switch for port information)
- Usage: Optional for hosts (can use SSH keys instead)

7.9.1.3.2 2. SSH Key Authentication

- Supported: HOST devices only (NOT supported for switches)
- Configuration: Set `CV_SSH_KEY_FILE` or enable `SSH_LOOK_FOR_KEYS`
- Usage: Available for hosts only
- Switch Limitation: Switches cannot use SSH keys because deployed agents need password credentials to communicate with the switch OS for retrieving port information

7.9.1.3.3 Authentication Priority (for hosts only)

1. SSH key authentication (if key available and no password set)
2. Password authentication (if password provided)
3. Automatic key discovery (if `SSH_LOOK_FOR_KEYS=true` and no password)

7.9.1.3.4 Switch Authentication Requirements

- All switch types require password authentication
- SSH keys are not supported for switches
- Passwords are used by agents for ongoing switch communication
- Supported switch OS types: Cumulus Linux, NVOS (NVLink/XDR switches), SONiC

7.9.1.4 Agent Deployment Process

7.9.1.4.1 Linux Devices Deployment Flow

1. Preparation Phase
    - System generates deployment script from template
    - Script customized with environment-specific values (image URLs, checksums, configuration)
    - Temporary deployment file created for transfer
2. File Transfer Phase (SFTP)
    - Secure connection established to target device
    - Deployment script uploaded to `/tmp` directory on target device
    - Connection closed after successful transfer
3. Execution Phase (SSH)
    - SSH connection established for command execution

    - Deployment script executed with elevated privileges
    - Cleanup commands remove temporary files
    - Connection closed after completion
4. Validation Phase
    - Deployment results logged and validated
    - Temporary files removed from both systems
    - Success/failure status reported

7.9.1.4.2 Deployment Script Features

The `install_agent.sh` script handles:

- Architecture detection (x86_64, aarch64)
- Docker prerequisite checks
- Image download with checksum verification
- Container deployment with appropriate parameters
- GPU support detection (for specific hardware)
- LLDP socket mounting (for ethernet monitoring)
- Comprehensive logging to `/var/log/cvt_deployment.log`

7.9.1.5 Agent Uninstall Process

7.9.1.5.1 Linux Devices Uninstall Flow

1. Preparation
    - Generate uninstall script from template (`uninstall_agent.sh`)
    - Create temporary uninstall file
2. File Transfer and Execution
    - Same SFTP upload process as deployment
    - Execute uninstall script with sudo privileges
3. Uninstall Operations
    - Container shutdown and removal
    - Docker image cleanup
    - System resource cleanup

7.9.1.6 Credential Management

7.9.1.6.1 Setting Credentials

The CVT system supports multiple levels of credential configuration:

Default Credentials

- Configure default username/password for all switches (password required)
- Configure default username/password for all hosts (password can be empty if using SSH keys)
- Applied when no specific credentials are found

Node-Specific Credentials

- Set unique credentials for individual devices
- Override default credentials for specific IP addresses
- For hosts: password can be empty when using SSH key authentication
- For switches: password is always required
- Highest priority in credential resolution

Credential Profiles

- Group devices with common credentials
- Assign profile names to device groups
- Manage credentials for multiple devices centrally
- Same password rules apply: switches require passwords, hosts can use empty passwords with SSH keys

7.9.1.6.2 Credential Priority

1. Node-specific credentials
2. Credential profile credentials (if assigned)
3. Default credentials for device type

7.9.1.7 Troubleshooting

7.9.1.7.1 Common SSH Issues

1. Authentication Failures

   SSH Authentication failure: please check device credentials

   - Verify credentials in CVT credential management
   - Check if SSH keys are properly configured for hosts
   - Ensure SSH service is running on target device
2. Connection Timeouts

   Failed to execute commands on node: <IP>: Connection timeout

   - Increase **SSH_CONN_TIMEOUT** value
   - Check network connectivity to target device
   - Verify firewall rules allow SSH (port 22)
3. Permission Denied

   Failed to execute commands on node: <IP>: Permission denied

   - Verify sudo access for the user account
   - Check if password is required for sudo
   - Ensure user has Docker access permissions
4. File Transfer Failures

   Failed to upload deployment file

   - Check SFTP connectivity
   - Verify write permissions to **/tmp** directory
   - Ensure sufficient disk space on target device

7.9.1.7.2 SSH Key Issues

1. Key File Not Found

- Verify `CV_SSH_KEY_FILE` path is accessible in container
- Check file permissions (should be 600 or 400)
- Ensure key file is mounted into container if using Docker volumes

2. Key Format Issues
   - Ensure key is in OpenSSH format (not PuTTY or other formats)
   - Verify key format compatibility with paramiko SSH library
   - Check for proper key file structure and encoding
3. Key Permission Problems
   - Verify SSH key file permissions are restrictive (600 or 400)
   - Ensure correct ownership of key files
   - Check that key files are readable by the CVT process

7.9.1.7.3 Deployment Script Issues

1. Docker Not Available

   `docker is not installed, it is required for running the agent`
   - Install Docker on target device
   - Ensure Docker service is running
   - Add user to docker group if needed
2. Image Download Failures

   `Failed to fetch the image from server`
   - Check network connectivity to image server
   - Verify image URL is accessible
   - Check firewall rules for HTTP/HTTPS traffic
3. Checksum Verification Failures

   `Checksum verification failed!`
   - Image may be corrupted during download
   - Network issues during transfer
   - Script will automatically retry download

7.9.1.7.4 Debugging Steps

1. Enable Debug Logging
   - Check deployment logs: `/var/log/cvt_deployment.log` on target device
   - Review CVT collector logs for SSH connection details
2. Manual SSH Testing
   - Test SSH connectivity with specified timeout values
   - Verify SFTP connectivity for file transfer operations
   - Test with specific SSH keys when configured
   - Validate authentication methods work as expected
3. Network Connectivity Testing
   - Verify basic network connectivity to target devices
   - Test SSH port accessibility (default port 22)
   - Check for firewall or network restrictions
   - Validate network latency and timeout settings

7.9.1.8 Best Practices

7.9.1.8.1 Security

1. SSH Key Management
    - Use dedicated SSH keys for CVT operations
    - Rotate keys regularly
    - Restrict key access with proper file permissions
    - Consider using SSH agent forwarding in containers
2. Credential Security
    - Use strong passwords
    - Implement credential rotation policies
    - Use credential profiles for device groups
    - Store credentials securely (CVT encrypts stored credentials)
3. Network Security
    - Use SSH key authentication when possible
    - Implement network segmentation
    - Configure firewall rules appropriately
    - Consider using SSH jump hosts for isolated networks

7.9.1.8.2 Performance

1. Connection Management
    - Adjust `SSH_CONN_TIMEOUT` based on network conditions
    - Use parallel deployment for multiple devices
    - Monitor deployment worker limits
2. Resource Management
    - Ensure sufficient bandwidth for image transfers
    - Monitor disk space on target devices
    - Clean up temporary files after deployment

7.9.1.8.3 Operational

1. Monitoring
    - Monitor deployment success rates
    - Track authentication failures
    - Review deployment logs regularly
2. Documentation
    - Maintain inventory of SSH keys and their usage
    - Document credential profiles and their assignments
    - Keep network topology documentation updated

7.9.1.9 Container Considerations

When running CVT in containers:

1. SSH Key Access
    - Mount SSH keys into container using volume mapping

    - Configure `CV_SSH_KEY_FILE` to point to container-accessible path
    - Verify key file permissions and ownership within container
2. Network Access
    - Ensure container can reach target devices
    - Configure network settings for direct device access
    - Verify container networking doesn't block SSH connections
3. SSH Agent
    - Forward SSH agent for key-based authentication
    - Configure agent socket mounting for container access
    - Verify agent accessibility within container environment

7.9.1.10 Troubleshooting Agent Deployment Issues on mlnx_os switches

While deploying the agents, if you encounter below error messages from the nodes you can try these steps:

- Error: `"% Failed to load image:[/var/opt/tms/images/cables_agent_latest.tar.gz].operation cancelled. check the log for details"`
  This can indicate that there is not enough space in the node to copy the agent image. Make some space and try again.
- Error: `"% Rule numbers must be consecutive: cannot have 40 without 39 (chain INPUT)"`
  This error suggests that the switch is not accepting the IP filter rule needed to permit communication between the collector and the agent on port 8251.
  The switch needs to support ip filter rules and have atleast 40 default rules.
  CVT checks if IP filter is enabled. Only if enabled and `Packet filtering for IPv4 Table filter` is set to enabled, the new rule will be added.
  If IP filter is not enabled, then CVT does not add any new rule because all ports are allowed by default.
  Check if IP filter is enabled: `show ip filter`. If you wish to enable it: `ip filter enable`
  If IP filter is enabled and does not have the default rules, add the default rules: `ip filter reset-to-default-rules`
  The new rule being added is `ip filter chain input rule insert 40 target accept dup-delete dest-port 8251 protocol tcp`

> ⚠ Note: EDR switches are not supported at the moment as they support only older software versions and don't have ip filter rules enabled and defined by default.

7.10 High Availability (HA) Mode Support for CVT Plugin

7.10.1 Overview

The CVT (Cable Validation Tool) plugin supports High Availability (HA) mode to ensure continuous operation and service resilience in production environments. In an HA deployment, CVT operates in an active-passive configuration where the collector runs on only one node at a time. When a failover event occurs, the CVT collector on the active node (node1) is stopped, and the CVT collector on the standby node (node2) automatically starts up. This architecture ensures that cable validation monitoring continues with minimal interruption, which is critical for maintaining network health and detecting issues in real-time.

7.10.2 HA Architecture

The HA implementation for CVT follows a shared-storage model where both nodes have access to common configuration and data files. This shared storage ensures that when the standby node takes over, it has immediate access to the same topology data, validation history, and configuration that was being used by the primary node. The shared resources include:

- Configuration files - All CVT configuration settings, including `cvt_env.conf`
- Topology data - Current and historical topology files
- Database and persistent storage - All collected metrics and historical information

Note on Validation State: The runtime validation state (agent status, validation results, and operational data) is maintained in memory only and is not persisted to shared storage. When a failover occurs, the standby node will reload the topology from shared storage and restart validation. The validation state will be rebuilt as agents are redeployed and begin reporting data. This means there will be a brief period during failover where the validation state is being re-established, but the topology configuration and historical data remain intact.

When a failover occurs—whether due to planned maintenance, system failures, or network issues—the standby node can quickly resume operations by loading the same topology and restarting validation, minimizing the gap in validation coverage.

7.10.3 Failover Configuration

To enable seamless automatic failover in HA mode, the CVT plugin requires specific configuration settings that allow the standby collector to automatically resume validation operations immediately upon startup. Without proper configuration, manual intervention would be required after each failover to reload the topology and restart validation, which would defeat the purpose of having an HA setup. The automated recovery mechanism ensures that network monitoring remains consistent and continuous across failover events.

7.10.4 Required Configuration Settings

For proper HA failover support, users must configure two critical parameters in the `cvt_env.conf` file under the `[application]` section:

Setting	Description
STARTUP_TOPOLOGY=last	This setting instructs the collector to automatically load the last successfully loaded topology from the history when starting up. In a failover scenario, this ensures that the standby collector immediately restores the exact topology state that was active on the primary collector before the failover. Since both nodes share the same data files, the "last" topology reference points to the same topology file that was in use on the primary node. Without this setting, the collector would start with no topology loaded (the default `none` value), requiring manual topology loading after every failover and causing a gap in validation coverage.
AUTO_START_VALIDATION=true	This setting enables automatic validation startup once a topology is successfully loaded. When combined with `STARTUP_TOPOLOGY=last`, this creates a fully automated recovery process: upon startup, the standby collector loads the last topology from shared storage and immediately begins validation operations without requiring any user intervention. This is essential for maintaining continuous validation during failover events, as it eliminates the manual steps that would otherwise be needed to resume monitoring. The validation process will automatically deploy agents to network devices and resume cable validation exactly as it was running on the primary node.

7.10.5 Complete HA Configuration Example

To configure CVT for HA mode with automatic failover support, update the `[application]` section in `cvt_env.conf`:

```
[application]
# Topology loading at startup - specify what to load:
#   none  - do not load any topology file (default)
#   last  - load the last loaded topology from history
#   <path> - load a specific topology file
(supports .topo, .dot, .xlsx, .json)
# For HA mode: set to 'last' to enable automatic topology restore on
failover
STARTUP_TOPOLOGY=last

# Automatically start validation if a topology file is loaded
# For HA mode: set to 'true' to enable automatic validation resume on
failover
AUTO_START_VALIDATION=true
```

7.10.6 Failover Behavior

With the above configuration, the failover sequence operates as follows:

1. Primary node (node1) failure - The CVT collector on node1 stops due to hardware failure, network issue, or planned maintenance
2. HA cluster failover - The cluster management system (e.g., Pacemaker, Kubernetes, or other HA solution) detects the failure and initiates failover to node2
3. Standby node (node2) startup - The CVT collector starts on node2
4. Automatic topology restore - CVT reads the `STARTUP_TOPOLOGY=last` setting and automatically loads the last topology from the shared storage
5. Automatic validation start - CVT reads the `AUTO_START_VALIDATION=true` setting and immediately begins validation operations
6. Validation state rebuild - Agents are redeployed to network devices and begin reporting data. The runtime validation state (agent status, current validation results) is rebuilt in memory as agents come online and start collecting cable data
7. Service resumed - Cable validation monitoring continues with minimal interruption

The entire process is fully automated, requiring no manual intervention to restore service after a failover event. While the topology configuration and historical data are immediately available from shared storage, the runtime validation state will be progressively rebuilt as agents reconnect and report their status.

7.10.7 Important Considerations

- Shared Storage: Ensure that the shared storage is reliable and accessible from both nodes with low latency
- Network Configuration: Both nodes should have similar network configurations to ensure agents can communicate with the collector regardless of which node is active
- Cluster Management: Use a proper HA cluster management solution to handle failover detection and node transitions
- Testing: Regularly test failover scenarios to ensure the configuration is working as expected
- Monitoring: Implement monitoring to detect failover events and verify that validation resumes successfully on the standby node

7.10.8 Summary

Together, the `STARTUP_TOPOLOGY=last` and `AUTO_START_VALIDATION=true` configuration settings create a robust failover mechanism where the standby collector can automatically take over validation operations in an active-passive HA deployment. This configuration ensures minimal disruption to network monitoring and allows HA deployments to provide the high reliability that production environments demand, without requiring manual intervention during failover events.

7.11 Cluster Sizing Guide

7.11.1 Overview

This sizing guide provides hardware and network recommendations for Cable Validation deployments based on cluster size. Recommendations are based on performance analysis of enterprise deployments and optimal resource utilization patterns.

Important Note: Cable Validation Tool (CVT) handles both **switches and hosts** in modern deployments. The legacy naming in the codebase (e.g., "SwitchAgentMgr", "switch_ip") reflects historical origins when CVT only handled switches, but now applies to all managed devices (switches, hosts, HCAs, etc.).

7.11.2 Sizing Methodology

Key Factors:

- Device Overload Threshold: Individual devices (switches/hosts) can handle ~5-10 concurrent REST API calls
- Network Bandwidth: 10G MGMT interface provides ~800-900 MB/s practical throughput
- CPU Utilization: Target 15-25 load average for optimal performance
- Memory Requirements: ~50-100 MB per 1000 devices for topology and batch processing
- Batch Processing: Optimal batch sizes scale with worker count
- Mixed Workloads: Switches and hosts may have different response characteristics

7.11.3 Quick Configuration Reference

7.11.3.1 🎯 Simple 3-Variable Configuration

CVT performance can be optimized with just three environment variables:

```
# 1. Agent Deployment (~200MB image + local container operations) export
CVT_DEPLOYMENT_MAX_WORKERS=60

# 2. Everything Else (validation, connectivity, DNS, etc.)  export CVT_MAX_WORKERS=150

# 3. Batching Control (when to split large deployments) export CVT_BATCHING_THRESHOLD=10000
```

Note: Agent deployment includes multiple phases: image fetch (~200MB), save to disk, load image, and container creation. Higher worker counts are possible because the process isn't purely bandwidth-limited.

See the Simple Tuning Guide for detailed configuration guidance.

7.11.4 Cluster Sizing Table

Cluster Size	Recommended CPUs	Recommended Memory	Recommended MAX_WORKERS	DEPLOYMENT_MAX_WORKERS	MGMT Bandwidth	Expected Time	Notes
Small Clusters (1-1,000 devices)							
100 devices	4-8 cores	4-8 GB	30-50	20	1G	30-60 seconds	Single server, basic setup
500 devices	8-16 cores	8-16 GB	50-75	20-40	1G	1-2 minutes	Development /test environment

Cluster Size	Recommended CPUs	Recommended Memory	Recommended MAX_WORKERS	DEPLOYMENT_MAX_WORKERS	MGMT Bandwidth	Expected Time	Notes
1,000 devices	16-32 cores	16-32 GB	50-100	20-40	1G	2-3 minutes	Small production deployment
Medium Clusters (1,000-10,000 devices)							
2,500 devices	32-64 cores	32-64 GB	75-100	40	10G	2-3 minutes	Regional deployment
5,000 devices	64-128 cores	64-128 GB	100-150	40-60	10G	3-5 minutes	Large regional deployment
7,500 devices	96-192 cores	96-192 GB	125-175	60	10G	4-6 minutes	Multi-site deployment
10,000 devices	128-256 cores	128-256 GB	150-200	60	10G	5-8 minutes	Enterprise deployment
Large Clusters (10,000-25,000 devices)							
15,000 devices	192-384 cores	192-384 GB	175-225	60-80	25G+	6-10 minutes	Large enterprise
20,000 devices	256-512 cores	256-512 GB	200-250	80	25G+	8-12 minutes	Hyperscale deployment
25,000 devices	320-640 cores	320-640 GB	225-275	80-100	40G+	10-15 minutes	Hyperscale deployment
Hyperscale Clusters (25,000+ devices)							
30,000 devices	384-768 cores	384-768 GB	250-300	100-120	40G+	12-18 minutes	Hyperscale datacenter
35,000 devices	448-896 cores	448-896 GB	275-325	120-140	40G+	15-20 minutes	Hyperscale datacenter
40,000 devices	512-1024 cores	512 GB-1TB	300-350	140-160	40G+	18-25 minutes	Massive hyperscale

7.11.5 Detailed Recommendations by Scale

7.11.5.1 Small Clusters (1-1,000 devices)

Characteristics:

- Single server deployment
- Basic network infrastructure
- Development/test environments
- Device Mix: Primarily switches, some hosts/HCAs

Sizing Logic:

- CPU: 1 core per 25-50 devices
- Memory: 10-20 MB per device for topology data
- Workers: Conservative scaling to avoid device overload
- Network: 1G sufficient for small clusters

7.11.5.1.1 Medium Clusters (1,000-10,000 devices)

Characteristics:

- Production deployments
- 10G management networks
- Regional or multi-site deployments
- Device Mix: Mixed switches and hosts, HCAs in compute clusters

Sizing Logic:

- CPU: 1 core per 40-80 devices (better efficiency at scale)
- Memory: 8-15 MB per device (shared topology data)
- Workers: Balanced scaling considering device capacity
- Network: 10G required for concurrent processing
- Host Considerations: Hosts may respond differently than switches

7.11.5.2 Large Clusters (10,000-25,000 devices)

Characteristics:

- Enterprise-scale deployments
- High-performance requirements
- 25G+ management networks
- Device Mix: Large numbers of compute hosts + infrastructure switches

Sizing Logic:

- CPU: 1 core per 60-100 devices (enterprise efficiency)
- Memory: 5-12 MB per device (optimized topology handling)
- Workers: Approaching device overload thresholds
- Network: 25G+ to handle concurrent load
- Mixed Response: Account for different device response characteristics

7.11.5.3 Hyperscale Clusters (25,000+ devices)

Characteristics:

- Massive datacenter deployments
- Enterprise-grade hardware (like customer's 448-core server)
- 40G+ management networks
- Device Mix: Thousands of compute hosts + infrastructure switches

Sizing Logic:

- CPU: 1 core per 80-120 devices (maximum efficiency)
- Memory: 3-10 MB per device (highly optimized)
- Workers: At or near device overload limits
- Network: 40G+ essential for performance
- Device Diversity: Must handle switches, hosts, HCAs, storage devices

7.11.6 Network Bandwidth Analysis

7.11.6.1 Bandwidth Requirements by Cluster Size

Cluster Size	Concurrent Workers	Peak Bandwidth Required	Network Recommendation	Device Types
1,000	50 workers	~50-100 MB/s	1G (sufficient)	Switches + some hosts
5,000	125 workers	~200-400 MB/s	1G (tight) / 10G (recommended)	Mixed switches/ hosts
10,000	175 workers	~400-700 MB/s	10G (required)	Balanced switches/ hosts
25,000	250 workers	~600-900 MB/s	10G (tight) / 25G (recommended)	Majority hosts + switches
40,000	350 workers	~800-1200 MB/s	25G (minimum) / 40G (optimal)	Large compute + storage

Bandwidth Calculation Logic:

- Per Worker: ~2-4 MB/s during active validation startup
- Peak Usage: During initial topology push to all devices
- Sustained Usage: Much lower during normal validation operation
- Burst Patterns: High bandwidth during startup, lower during monitoring

Key Insights:

- Bandwidth is BURSTY: High during startup, low during validation
- 10G Limit: Starts getting tight around 10,000 devices
- 25G Sweet Spot: Good performance for 25,000-40,000 devices
- 40G Future-Proof: Optimal for large hyperscale deployments

7.11.7 Memory Usage Patterns

7.11.7.1 Memory Breakdown by Component

Component	Memory per 1000 Devices	Notes
Topology Data	20-40 MB	Device definitions, links, mixed switches/hosts
Batch Processing	15-30 MB	Temporary data during processing
Connection Pools	5-10 MB	HTTP session management
Results Storage	10-20 MB	Validation results and reports
Device Metadata	5-15 MB	Host-specific data, HCA mappings
Total	**55-115 MB**	**Per 1000 devices (switches + hosts)**

7.11.8 Performance Optimization Guidelines

7.11.8.1 Quick Performance Tuning

For detailed tuning instructions and troubleshooting, see the Simple Tuning Guide which provides:

- Easy-to-follow configuration decisions
- Monitoring guidance and success criteria
- Troubleshooting common issues
- System tuning for large deployments

7.11.8.2 CPU Optimization

- Target Load: 15-25 average load during processing
- NUMA Awareness: Use dual-socket servers for 20,000+ devices
- Worker Scaling: Adjust `CVT_MAX_WORKERS` based on CPU cores and observed load
- Device Mix: Account for different CPU requirements of switches vs hosts
- Monitoring: If load stays low, increase workers; if timeout errors increase, reduce workers

7.11.8.2.1 Memory Optimization

- Batching: Use `CVT_BATCHING_THRESHOLD` to control memory usage on large deployments
- Connection Pooling: Automatically scales with worker count
- Garbage Collection: Monitor for large deployments (20,000+ devices)
- Device Metadata: Additional memory for host-specific data (HCA mappings, etc.)

7.11.8.2.2 Network Optimization

- Bandwidth Planning: Set `CVT_DEPLOYMENT_MAX_WORKERS` based on management network capacity
- Connection Reuse: Essential for large deployments (handled automatically)

- Bandwidth Monitoring: Watch for saturation at scale
- Device Response Variance: Hosts may respond differently than switches
- Burst Patterns: High bandwidth during startup, lower during validation operation

7.11.9 Device Type Considerations

7.11.9.1 Switches vs Hosts Performance Characteristics

Device Type	Typical Response Time	Concurrent Call Limit	Special Considerations
Network Switches	1-3 seconds	5-10 concurrent	REST API on switch OS
Compute Hosts	2-5 seconds	3-8 concurrent	Agent on host OS, may be busier
Storage Devices	1-4 seconds	5-12 concurrent	Usually dedicated management
HCA Devices	1-2 seconds	8-15 concurrent	Lightweight agent

7.11.10 Example Configurations

7.11.10.1 Small Deployment (<1,000 devices)

```
# Network: 1G management interface
# Server: 8-32 cores
export CVT_DEPLOYMENT_MAX_WORKERS=20
export CVT_MAX_WORKERS=50
# CVT_BATCHING_THRESHOLD=10000 (default, no need to change)
```

7.11.10.2 Medium Deployment (1,000-10,000 devices)

```
# Network: 10G management interface
# Server: 64-128 cores
export CVT_DEPLOYMENT_MAX_WORKERS=40
export CVT_MAX_WORKERS=100
# CVT_BATCHING_THRESHOLD=10000 (default, no need to change)
```

7.11.10.3 Large Deployment (10,000-30,000 devices)

```
# Network: 25G+ management interface
# Server: 192-384 cores
export CVT_DEPLOYMENT_MAX_WORKERS=60
export CVT_MAX_WORKERS=150
export CVT_BATCHING_THRESHOLD=5000
```

7.11.10.4 Hyperscale Deployment (30,000+ devices)

```
# Network: 40G+ management interface
# Server: 448+ cores
export CVT_DEPLOYMENT_MAX_WORKERS=80
export CVT_MAX_WORKERS=200
export CVT_BATCHING_THRESHOLD=3000
```

7.11.11 Configuration Guidelines

CVT_DEPLOYMENT_MAX_WORKERS (Agent Deployment):

- Based on network bandwidth and local container operations (~200MB per device image)
- Deployment includes: image fetch, save to disk, load image, container creation
- 1G network: 20 workers (~4GB concurrent + local ops)
- 10G network: 60 workers (~12GB concurrent + local ops)
- 25G+ network: 100-160 workers (higher bandwidth + parallel local operations)

CVT_MAX_WORKERS (Validation Operations):

- Based on server CPU cores and device capacity
- 8-32 cores: 30-50 workers
- 32-128 cores: 75-150 workers
- 128+ cores: 150-300 workers
- Watch for device timeout errors and reduce if needed

CVT_BATCHING_THRESHOLD (Batch Processing):

- <5,000 devices: 10000 (default, single batch)
- 5,000-20,000 devices: 5000 (light batching)
- 20,000+ devices: 3000 (aggressive batching)

7.11.12 Scaling Considerations

7.11.12.1 Vertical Scaling Limits

- Single Server: Effective up to ~40,000 devices (switches + hosts)
- CPU Bound: Beyond 40,000 devices, consider distributed processing
- Memory Bound: Rarely an issue with modern servers (hosts require slightly more memory)
- Network Bound: Primary constraint for large deployments
- Device Mix: Higher host percentage may require more resources

7.11.12.2 Horizontal Scaling Options

- Multiple Collectors: Split clusters across multiple servers
- Geographic Distribution: Regional collectors for global deployments
- Load Balancing: Distribute devices across multiple validation instances

7.11.13 Device Mix Impact on Sizing

7.11.13.1 Typical Cluster Compositions

Cluster Type	Switches %	Hosts %	Notes	Sizing Impact
Infrastructure-Heavy	80%	20%	Network-focused deployment	Lower memory, higher network load
Compute-Heavy	30%	70%	HPC/AI clusters	Higher memory, variable response times
Balanced	50%	50%	Mixed enterprise deployment	Standard sizing applies
Storage-Heavy	40%	60%	Storage clusters with many storage hosts	Higher memory, faster responses

7.11.13.2 Sizing Adjustments by Device Mix

Infrastructure-Heavy Clusters (80% switches):

- CPU: Use lower end of range
- Memory: Use lower end of range
- Workers: Can be more aggressive
- Network: Higher bandwidth needs per device

Compute-Heavy Clusters (70% hosts):

- CPU: Use higher end of range
- Memory: Use higher end of range (HCA mappings, host metadata)
- Workers: More conservative (hosts may be busier)
- Network: Variable load patterns

Balanced Clusters (50/50 mix):

- CPU: Use middle of range
- Memory: Use middle of range
- Workers: Standard recommendations apply
- Network: Standard bandwidth planning

7.11.13.3 Monitoring and Alerting

7.11.13.3.1 Key Metrics to Monitor

1. Server Load: Target 15-25 during processing
2. Memory Usage: Should stay well below allocated
3. Network Utilization: Watch for bandwidth saturation (especially during agent deployment)
4. Device Response Times: Primary performance indicator
5. Error Rates: Timeout and connection errors
6. Validation Completion Time: Compare against expected times in sizing table

7.11.13.3.2 Success Indicators

Good Signs (can increase `CVT_MAX_WORKERS`):

- ✅ Server load increases during validation (better CPU utilization)
- ✅ Validation completes faster than baseline
- ✅ No significant increase in timeout errors
- ✅ Network bandwidth stays below 80%

Warning Signs (reduce `CVT_MAX_WORKERS`):

- ⚠️ Many "Timeout while trying to start validation" errors
- ⚠️ "Connection refused" errors from devices
- ⚠️ Server load stays low (underutilization)
- ⚠️ Network bandwidth hits 90%+

7.11.13.3.3 Scaling Triggers

- Scale Up Workers: Load < 10, low error rates, fast completion
- Scale Down Workers: High error rates, device timeouts, network saturation
- Increase Deployment Workers: Network utilization < 50% during agent deployment
- Decrease Deployment Workers: Network bandwidth > 80% during agent deployment
- Adjust Batching: Memory usage > 80% (reduce `CVT_BATCHING_THRESHOLD`)

7.11.13.3.4 Performance Expectations by Cluster Size

Cluster Size	Expected Validation Time	Expected Load Average	Target Worker Count
1,000 devices	1-3 minutes	8-12	50-100
5,000 devices	3-5 minutes	12-18	100-150
10,000 devices	5-8 minutes	15-22	150-200
25,000 devices	10-15 minutes	18-25	225-275
40,000 devices	18-25 minutes	20-28	300-350

7.11.13.4 Document Relationship

This Cluster Sizing Guide provides comprehensive hardware and infrastructure planning:

- CPU, memory, and network capacity planning
- Expected performance at different scales
- Device type considerations (switches, hosts, HCAs)
- Detailed sizing methodology

For day-to-day performance tuning, refer to the Simple Tuning Guide:

- Simple 3-variable configuration
- Quick-start configurations by deployment size
- Troubleshooting and monitoring guidance

- Practical tuning adjustments

7.11.14 Simple Tuning Guide

7.11.14.1 Simple CVT Performance Tuning Guide

7.11.14.1.1 Easy Configuration - Just 3 Variables

Based on extensive testing and real-world deployments, CVT performance can be optimized with just these simple settings:

7.11.14.1.1.1 Core Configuration:

```
# 1. Agent Deployment (~200MB image + local container operations)
export CVT_DEPLOYMENT_MAX_WORKERS=60

# 2. Everything Else (validation, connectivity, DNS, etc.)
export CVT_MAX_WORKERS=150

# 3. Batching Control (when to split large deployments)
export CVT_BATCHING_THRESHOLD=10000
```

7.11.14.1.2 How to Decide Values

7.11.14.1.2.1 CVT_DEPLOYMENT_MAX_WORKERS (Agent Deployment)

Question: How much bandwidth can your management network handle?

Note: Deployment includes multiple phases: image fetch (~200MB), save to disk, load image, and container creation. The process is not purely bandwidth-limited, so we can use higher worker counts.

Network Speed	Recommended Value	Reasoning
1G	20	20 × 200MB = ~4GB concurrent + local ops
10G	60	60 × 200MB = ~12GB concurrent + local ops
25G+	100-160	Higher bandwidth + parallel local operations

7.11.14.1.2.2 CVT_MAX_WORKERS (Everything Else)

Question: How many CPU cores does your server have?

Server CPU Cores	Recommended Value	Reasoning
8-32 cores	30-50	Conservative scaling
32-128 cores	75-150	Balanced scaling
128+ cores	150-300	Aggressive scaling

But watch for switch overload! If you see many timeout errors, reduce this value.

7.11.14.1.2.3 CVT_BATCHING_THRESHOLD (When to Batch)

Question: How many switches do you have?

Switch Count	Recommended Value	What Happens
<5,000	10000 (default)	Single batch (faster)
5,000-20,000	5000	Light batching
20,000+	3000	More aggressive batching

7.11.14.1.3 Quick Start by Deployment Size:

7.11.14.1.3.1 Small Deployment (<1,000 switches)

```
export CVT_DEPLOYMENT_MAX_WORKERS=20
export CVT_MAX_WORKERS=50
# No need to change batching threshold
```

7.11.14.1.3.2 Medium Deployment (1,000-10,000 switches)

```
export CVT_DEPLOYMENT_MAX_WORKERS=40
export CVT_MAX_WORKERS=100
# No need to change batching threshold
```

7.11.14.1.3.3 Large Deployment (10,000-30,000 switches)

```
export CVT_DEPLOYMENT_MAX_WORKERS=60
export CVT_MAX_WORKERS=150
export CVT_BATCHING_THRESHOLD=5000
```

7.11.14.1.3.4 Hyperscale Deployment (30,000+ switches)

```
export CVT_DEPLOYMENT_MAX_WORKERS=80
export CVT_MAX_WORKERS=200
export CVT_BATCHING_THRESHOLD=3000
```

7.11.14.1.4 How to Know If Your Settings Are Good

7.11.14.1.4.1 Good Signs:

- ✅ Server load increases during validation (better CPU utilization)
- ✅ Validation completes faster than before
- ✅ No significant increase in timeout errors
- ✅ Network bandwidth stays below 80%

7.11.14.1.4.2 Warning Signs:

- ⚠️ Many "Timeout while trying to start validation" errors
- ⚠️ "Connection refused" errors from switches

- ⚠️ Server load stays low (underutilization)
- ⚠️ Network bandwidth hits 90%+

7.11.14.1.4.3 Adjustment Strategy:

1. Too many timeouts: Reduce `CVT_MAX_WORKERS` by 25-50
2. Server underutilized: Increase `CVT_MAX_WORKERS` by 25-50
3. Deployment too slow: Increase `CVT_DEPLOYMENT_MAX_WORKERS` (if network allows)
4. Memory issues: Reduce `CVT_BATCHING_THRESHOLD`

7.11.14.1.5 Expected Performance

7.11.14.1.5.1 Your Customer's 32,149 Switches:

Current Configuration:

```
export CVT_DEPLOYMENT_MAX_WORKERS=60    # Optimal for 10G network
with 200MB image
export CVT_MAX_WORKERS=150              # Good for 448-core server
export CVT_BATCHING_THRESHOLD=10000     # Will use batching (32K >
10K)
```

Expected Results:

- Current: 7m 13s
- Optimized: 2-3 minutes (60-75% improvement)
- Server Load: Should increase from 5-7 to 15-20

7.11.14.1.6 Monitoring and Troubleshooting

7.11.14.1.6.1 Critical Monitoring Points:

Watch for Device Overload:

1. Timeout Errors: Increase in "Timeout while trying to start validation" messages
2. Connection Refused: "Connection error" messages from devices
3. Response Times: Slower device response times
4. Failure Rate: Higher percentage of failed device connections

Server Utilization Monitoring:

1. Load Average: Should increase from baseline to target ranges
2. CPU Usage: Better utilization of available cores
3. Memory: Watch for any memory pressure during large deployments
4. Network: Monitor bandwidth utilization on management interface

7.11.14.1.6.2 Success Criteria by Worker Count

Worker Count	Expected Load Average	Expected Time Improvement	Risk Level
50-75	8-12	30-50% faster	Low

Worker Count	Expected Load Average	Expected Time Improvement	Risk Level
100-150	12-18	50-70% faster	Medium
200+	18-25	70%+ faster	High

7.11.14.1.6.3 Red Flags (Scale Back If You See)

- ⚠️ Significant increase in timeout errors
- ⚠️ "Connection refused" errors from devices
- ⚠️ Device response times getting slower
- ⚠️ Higher failure rates than baseline

7.11.14.1.6.4 Green Lights (Scale Up If You See)

- ✅ Stable or improved device response times
- ✅ No increase in connection errors
- ✅ Server load well below target range
- ✅ Good success rate maintained

7.11.14.1.7 System Tuning for Large Deployments

7.11.14.1.7.1 File Descriptor Limits

```
# Increase file descriptor limits for high concurrency
ulimit -n 65536

# Make permanent by adding to /etc/security/limits.conf:
echo "* soft nofile 65536" >> /etc/security/limits.conf
echo "* hard nofile 65536" >> /etc/security/limits.conf
```

7.11.14.1.7.2 Network Optimization

```
# Optimize network settings for high concurrency
echo 65536 > /proc/sys/net/core/somaxconn
echo 1 > /proc/sys/net/ipv4/tcp_tw_reuse
echo 1 > /proc/sys/net/ipv4/tcp_tw_recycle
```

7.11.14.1.7.3 Memory Settings

```
# For very large deployments (20,000+ devices)
echo 1 > /proc/sys/vm/overcommit_memory
echo 80 > /proc/sys/vm/overcommit_ratio
```

7.11.14.1.8 Agent Deployment Time Estimates

7.11.14.1.8.1 Deployment Duration by Cluster Size:

Devices	Workers	Network	Estimated Time
1,000	20	1G	1-2 hours

Devices	Workers	Network	Estimated Time
4,000	60	10G	2-4 hours
10,000	80	25G	4-8 hours
25,000+	100-160	40G+	10-20 hours

Note: Agent deployment includes image fetch (~200MB per device), local save, image load, and container creation. With the reduced image size and parallel local operations, deployment is significantly faster than with larger images.

7.11.14.1.9 Bottom Line

Most customers only need to set 2 variables:

1. `CVT_DEPLOYMENT_MAX_WORKERS` (based on network bandwidth)
2. `CVT_MAX_WORKERS` (based on server CPU and device tolerance)

The third variable (`CVT_BATCHING_THRESHOLD`) usually works fine at default!

7.12 CVT as UFM Plugin

CVT can be installed as a UFM plugin. it provides the same functionalities as standalone CVT. CVT UI is embedded into UFM UI, no CVT user management since it's managed by UFM.

There are specific features related to integration with UFM.

1. Send CVT issues as Alarms to UFM
2. Load Topology directly from UFM.

7.12.1 CVT Issues Integration with UFM Alarms

7.12.1.1 Overview

When the Cable Validation Tool (CVT) runs as a UFM Plugin, cable-validation issues are pushed to UFM's Events log, so operators can view and act on them directly from the standard UFM Events UI alongside other system events.

7.12.1.1.1 Issues reported to UFM Events

- Wrong neighbor / Unknown neighbor / Wrong port
- Missing cable / Extra cable
- Link down / Protocol down / Admin down / Rx disable
- Negotiation failure / Flapping link / Anomalous port
- Unprovisioned
- Module P/N mismatch

7.12.1.1.2 Not reported to UFM Events

- Unreachable switch – device-level, reported only inside CVT.

- CC issue – detected by the collector, not forwarded.
- RX/TX power mismatch (power loss) – detected by the collector, not forwarded.

> ***Availability: this feature is supported only when CVT is installed as a UFM Plugin. Standalone CVT deployments do not send events to UFM.***

7.12.1.2 Prerequisites

#	Requirement
1	UFM Enterprise is installed and running.
2	CVT Plugin is installed on the UFM host and running.
3	You can edit `cvt_env.conf` and restart the CVT plugin.

7.12.1.3 3. Configuration file: `cvt_env.conf`

All settings for this feature live in the CVT configuration file `cvt_env.conf`, under the **[ufm events]** section.

Deployment	Path
UFM Plugin (inside container)	`/config/cvt_env.conf`
ufm machine	**/opt/ufm/files/conf/plugins/cablevalidation/cvt_env.conf**

7.12.1.4 Enabling the feature

Edit `cvt_env.conf` and set:
[ufm events]
# Master switch to send CVT issues to UFM as external events
ENABLE_UFM_EXTERNAL_EVENTS = true

Then restart the CVT plugin.

If **ENABLE_UFM_EXTERNAL_EVENTS** is **false** or missing, CVT works normally but sends no events to UFM.

7.12.1.5 [ufm events] variables

Variable	Default	Description
ENABLE_UFM_EXTERNAL_EVENTS	**false**	Master switch. **true** = send CVT issues to UFM Events.
UFM_SEND_ON_NEW_ISSUE_ONLY	**true**	**true** = send events only for newly detected issues. **false** = send events for all current issues on every report cycle.

Variable	Default	Description
`UFM_MAX_EVENTS_PER_SEND`	`100`	Maximum events per batch sent to UFM. `0` disables batching (sends all in one request — not recommended for large event counts).
`UFM_SEVERITY_<SYNDROME>`	`WARNING`	Override severity per syndrome. Allowed values: `INFO`, `WARNING`, `CRITICAL`. Invalid values are ignored with a warning.

7.12.1.5.1 Severity override keys (supported syndromes)

`LINK_DOWN`, `PROTO_DOWN`, `ADMIN_DOWN`, `RX_DISABLE`, `NEG_FAIL`, `UNPROVISIONED`, `WRONG_NEIGHBOR`, `UNKNOWN_NEIGHBOR`, `WRONG_PORT`, `MISSING_CABLE`, `EXTRA_CABLE`, `FLAPPING_LINK`, `ANOMALOUS_PORT`, `MODULE_PN_MISMATCH`.

7.12.1.6 Viewing CVT events in UFM

1. Log in to the UFM Web UI.
2. Open the Events page.
3. Filter by Source = `cable_validation(plugin)`.
4. Each CVT event contains:
    - Severity — `INFO` / `WARNING` / `CRITICAL`.
    - Event ID — `551` (INFO), `553` (WARNING), `554` (CRITICAL).
    - Source — always `cable_validation(plugin)`.
    - Source Type — the syndrome (e.g. `Wrong-neighbor`, `Extra-cable`, `Link-down`).
    - Object — the affected port.
    - Description — `"<syndrome>: <port>. <corrective action>"`.

7.12.2 UFM Topology Support

7.12.2.1 UFM Topology Support

7.12.2.1.1 Feature Summary

The UFM Topology feature allows CVT to read the UFM periodic topology file and use it in two ways:

1. Load directly -- load the UFM topology for cable validation without requiring a manually crafted PTP file.
2. Generate a stub PTP -- produce a Unified PTP Excel file from the UFM topology that users can enrich offline (add rack layout, IPs, server profiles) and re-upload.

This feature is available only when CVT runs as a UFM plugin (`CV_UFM_PLUGIN_MODE=true`). The topology source is the IBDM periodic topo file that UFM writes to disk at `/opt/ufm/files/periodicTopo/master.topo` .

7.12.2.1.2 Architecture

```
/opt/ufm/files/periodicTopo/master.topo
            |
            v
    TopoFileParser (topo_ib_file_parser.py)
            |
            v
  SwitchData / HostData / LinkData
            |
    +-------+-------+
    |               |
    v               v
Load for        UfmPtpGenerator
validation      (ufm_ptp_generator.py)
    |               |
    v               v
Active          unified_ptp.xlsx
Topology        (4 sheets)
```

7.12.2.1.2.1 Key Components

Component	Location	Role
`TopoFileParser`	`common/topolib/topo_ib_file_parser.py`	Parses `.topo` IBDM files
`UfmPtpGenerator`	`common/topolib/ufm_ptp_generator.py`	Converts parsed topology to PTP xlsx
`TopologyLoadService`	`collector/topology_load_service.py`	Orchestrates `load_ufm_topo()` and `generate_ptp_from_ufm()`
`RequestHandler`	`collector/request_handler.py`	REST endpoint definitions
`Topology`	`common/topolib/topo.py`	Topology model and `validate_switch_data()`

7.12.2.1.3 User Workflows

#	Action	Endpoint	Result
1	Load UFM topology directly	`POST /topology/load/ufm`	Topology loaded, ready for validation
2	Generate stub PTP from UFM	`POST /topology/generate/ptp`	PTP xlsx saved to `uploads/topology/ptp/`
3	Download generated PTP	`GET /files_manager/ptp/files/{name}`	User downloads xlsx
4	Edit PTP offline	--	User adds rack, layout, server profiles
5	Re-upload edited PTP	`POST /topology/load/ptp`	Enriched topology loaded for validation

7.12.2.1.4 UI Walkthrough

7.12.2.1.4.1 UFM Tab in Load Topology Wizard

- The UFM tab appears in the Load Topology wizard only when CVT is running in UFM plugin mode (`CV_UFM_PLUGIN_MODE=true`).
- The tab provides two actions:
  - Load UFM Topology -- loads the UFM periodic topology directly for validation.
  - Generate PTP from UFM -- generates a stub PTP Excel file from the UFM topology, which the user can then download, edit offline, and re-upload.
- Optional enrichment inputs can be provided:
  - IP File -- maps hostnames to IP addresses (populates the IP column in the Nodes sheet).
  - HCA Mapping -- overrides the default `mlx5_X` port derivation for hosts with non-standard port naming.
  - Server Profile -- replaces the auto-generated Server Profile sheet entirely.
  - DNS toggle -- use hostnames for agent communication (default: true).
- `hca_mapping` and `server_profile` are mutually exclusive; providing both results in a 400 error.

7.12.2.1.4.2 Tab Visibility

- When `CV_UFM_PLUGIN_MODE=false` (standalone deployment), the UFM tab is hidden from the UI.
- The UI checks `window["is_ufm"]` to control tab visibility.

7.12.2.1.5 REST API Quick Reference

7.12.2.1.5.1 POST /topology/load/ufm

Load the UFM periodic topo file directly for validation.

Request body (all fields optional):

```
{ "ip_file": "uploads/topology/ip/fabric.ip", "hca_mapping": "uploads/topology/hca_mapping/mapping.csv", "server_profile": "uploads/topology/server_profile/profile.csv", "dns": true }
```

Parameter	Type	Description
`ip_file`	string	IP mapping file for switch/host IP resolution
`hca_mapping`	string	HCA mapping CSV for port name mapping
`server_profile`	string	Server profile CSV (mutually exclusive with `hca_mapping`)
`dns`	bool	Use hostnames for communication (default: true)

Response: `202 Accepted`

Poll `GET /topology/load/result` for completion status.

7.12.2.1.5.2 POST /topology/generate/ptp

Generate a stub PTP xlsx from the UFM periodic topo file.

Request body (all fields optional):

`{ "output": "my_fabric_ptp.xlsx", "ip_file": "uploads/topology/ip/fabric.ip", "hca_mapping": "uploads/topology/hca_mapping/mapping.csv", "server_profile": "uploads/topology/server_profile/profile.csv" }`

Parameter	Type	Description
`output`	string	Output filename (default: timestamped `ufm_ptp_YYYYMMDD_HHMMSS.xlsx`)
`ip_file`	string	IP mapping file to populate the IP column
`hca_mapping`	string	HCA mapping CSV for port name overrides
`server_profile`	string	Server profile CSV (mutually exclusive with `hca_mapping`)

Response: `202 Accepted`

Poll `GET /topology/load/result` for completion. On success:

`{ "status": "completed", "file": "ufm_ptp_20260311_143000.xlsx", "switches": 42, "hosts": 120, "links": 500, "warnings": [] }`

Download via `GET /files_manager/ptp/files/ufm_ptp_20260311_143000.xlsx`.

7.12.2.1.5.3 Validation Rules

- `hca_mapping` and `server_profile` are mutually exclusive -- 400 if both provided.
- Both endpoints require `CV_UFM_PLUGIN_MODE=true` -- 400 otherwise.
- Both operations are asynchronous -- clients must poll for results.

7.12.2.1.6 Generated PTP File Format

The output follows the Unified PTP Excel format with four sheets:

7.12.2.1.6.1 Nodes Sheet

Column	Auto-Populated	Notes
FabricId	No	Empty stub for user to fill
Rack	No	Empty stub for user to fill
Unit	No	Empty stub for user to fill
TrayIndex	No	Empty stub for user to fill
NodeName	Yes	Hostname extracted from topo
NodeType	Yes	`switch` or `host`
NodeOS	Yes	`mlnx_os` for switches, `linux` for hosts

Column	Auto-Populated	Notes
NodeModel	Yes	Switch model (e.g., `MQM8700`); empty for hosts
ServerProfile	Yes	`ufm_default` for hosts; empty for switches
Managed	Yes	`yes` for all nodes
CredentialProfile	No	Empty
IP	Optional	Populated if `ip_file` is provided

7.12.2.1.6.2 Links Sheet

Column	Auto-Populated	Notes
Protocol	Yes	`ib` (from topo parser)
Shuffle-ID	No	Empty
A-Node	Yes	Source node hostname
A-Port	Yes	Switch port (e.g., `P12`) or host port (e.g., `mlx5_0`)
A-Module-PN	No	Empty
Z-Node	Yes	Destination node hostname
Z-Port	Yes	Same format as A-Port
Z-Module-PN	No	Empty

7.12.2.1.6.3 DC Floor Layout Sheet

Headers only (empty stub for user to fill).

7.12.2.1.6.4 Server Profile Sheet

Auto-generated default mapping based on observed HCA indices:

ServerProfile	CustomNicName	NicOSName	PhysicalPort	RDMAName
ufm_default	mlx5_0	mlx5_0	1	mlx5_0
ufm_default	mlx5_1	mlx5_1	2	mlx5_1
...	...	...	...	...
ufm_default	mlx5_N	mlx5_N	N+1	mlx5_N

Range determined from the maximum `mlx5_X` index observed across all host links. Overridden entirely if `server_profile` CSV is provided.

7.12.2.1.7 Data Transformation Rules

Input	Transformation	Output
Switch line: `MQM8700 sw-hdr-01`	NodeName = 2nd token, NodeModel = 1st token	Node: `sw-hdr-01`, model `MQM8700`
Host line: `HCA_mlx5_2 compute-01`	Strip `HCA_*` prefix	Node: `compute-01`, type `host`
Host port: `mlx5_0/P1`	Strip `/P\d+` suffix	Port: `mlx5_0`
SYS node: `SYS0c42a... S0c42a...`	Excluded (unmanaged)	Not in PTP output
Down ports	Not present in UFM topo file	Silently accepted

7.12.2.1.8 Configuration

Setting	Default	Where	Description
`CV_UFM_PLUGIN_MODE`	`false`	Environment / Dockerfile	Enables UFM plugin features
`CVT_UFM_TOPO_FILE`	`/opt/ufm/files/periodicTopo/master.topo`	`cvt_env.conf`	Path to UFM periodic topo file

7.12.2.1.8.1 Docker Volume Mount

The CVT container requires read-only access to UFM filesystem:

`/opt/ufm/` files:/opt/ufm/files:ro

Configured in `conf/plugin/cablevalidation_shared_volumes.conf`.

7.12.2.1.9 Prerequisites

1. UFM plugin mode enabled: CVT must be deployed as a UFM plugin with `CV_UFM_PLUGIN_MODE=true`.
2. UFM periodic topology file exists: UFM must have generated the topo file at the configured path.
3. Volume mount configured: The CVT container must have read access to `/opt/ufm/files`.

8 Cables Agent

The Cables agent is implemented as a docker container that executes on the switch to gather data on neighboring switches and link quality. The agent operates a web service capable of providing information on ports and links through user queries. Moreover, the agent transmits validation reports to the bringup server.

- Open Agent Port
- Cable Agent Status
- Collecting Amber File

8.1 Open Agent Port

The agent establishes communication with the CVT collector using port 8251. To ensure proper functionality, this port must be open on the switch's firewall. The process of enabling this port differs based on the type of fabric being used:

1. InfiniBand Fabric:
   In the InfiniBand fabric, the tool automatically opens port 8251 without requiring any manual intervention from the user. No additional configuration is necessary.
2. Ethernet Fabric:
   For ethernet fabric, port 8251 must be opened manually or using ZTP tools. This can be accomplished by executing the following command directly on the switch:

```
nv set acl acl-default-whitelist rule 200 match ip tcp dest-port 8251
nv set acl acl-default-whitelist type ipv4
nv config apply
nv config save
```

3. NVOS Fabric:
   For NVOS fabric (XDR switches), port 8251 must be opened manually. This can be accomplished by executing the following commands directly on the switch:

```
nv set acl AAA type ipv4
nv set acl AAA rule 1 match ip tcp dest-port 8251
nv set acl AAA rule 1 action permit
nv set interface eth0-1 acl AAA inbound control-plane
nv config apply
nv config save
```

8.2 Cable Agent Status

8.2.1 REST APIs to check agent status

1. Check the CVT version running on the agent using **/version** endpoint of the agent.
   Example:

```
# curl -k https://<hostname/IP>:8251/version
{"agent_version": "1.8.1-4"}
```

2. Check the agent status using **/status** endpoint of the agent. It reports the current validation state and to which collector it is sending reports among other details. Example:

```
# curl -k https://<hostname/IP>:8251/status
{"state": "validating", "state_since": 1771547598, "auto_transition_via_idle": true, "collector_ip":
"<ip>", "last_publish_time": 1771610839}
```

> ⚠ Note: The above REST APIs would fail even if the agent is installed and running if the port 8251 is blocked.

8.2.2 Check if Agent Container is running

8.2.2.1 On InfiniBand

Check if cable agent is running on the switch:

1. Run:

```
ssh admin@<switch-ip-or-name>
```

2. Enable
3. Show docker images
4. Exit

If cables agent is running on the switch, the following output is prompted.

```
---------------------------------------------------------------------------
Image                                   Version      Created            Size
---------------------------------------------------------------------------
cables_agent                            latest       13 hours ago       788MB
```

8.2.2.2 On Ethernet, NVOS, and Hosts

Check if cable agent is running on the switch:

1. Run:

```
ssh admin@<switch-ip-or-name>
```

2. docker ps

If cables agent is running on the switch, the following output is prompted.

```
---------------------------------------------------------------------------
Image                                   Version      Created            Size
---------------------------------------------------------------------------
cables_agent                            latest       13 hours ago       788MB
```

8.3 Collecting Amber File

8.3.1 Ethernet Switches, InfiniBand Switches (NDR, HDR) and Hosts

Ethernet and InfiniBand(NDR/HDR) switches have one MST device for which Amber file is collected.

Hosts have one MST device per NIC, hence Amber file is collected per mst device.

Following command can be used to collect amber file:

1. Find the MST device by running the command:

```
# sudo mst status -v
```

2. Run mlxlink command to collect the amber:

```
# mlxlink -d <mst_device_name> --amber_collect <path_to_save_the_amber_file>
```

Example:

```
mlxlink -d /dev/mst/mt54000_pciconf0 -amber_collect /tmp/amber.csv
```

8.3.2 XDR Switches

On XDR switches there are four devices - one per ASIC.

1. To get the amber for all the ASICs, create a planarized device. To do so, run the following command:

```
# sudo mst start --planarized_device_pci
```

Expected output:

```
admin@swx-sw-bm01:~$ sudo mst status -v
MST modules:
------------
    MST PCI module loaded
    MST PCI configuration module loaded
PCI devices:
------------
DEVICE_TYPE             MST                           PCI       RDMA            NET                                     NUMA
Quantum3(rev:0)         /dev/mst/mt54004_pciconf3     09:00.0                                                   -1

Quantum3(rev:0)         /dev/mst/mt54004_pci_cr3      09:00.0                                                   -1

Quantum3(rev:0)         /dev/mst/mt54004_pciconf2     05:00.0                                                   -1

Quantum3(rev:0)         /dev/mst/mt54004_pci_cr2      05:00.0                                                   -1

Quantum3(rev:0)         /dev/mst/mt54004_pciconf1     04:00.0                                                   -1

Quantum3(rev:0)         /dev/mst/mt54004_pci_cr1      04:00.0                                                   -1

Quantum3(rev:0)         /dev/mst/mt54004_pciconf0     03:00.0                                                   -1

Quantum3(rev:0)         /dev/mst/mt54004_pci_cr0      03:00.0                                                   -1

Planarized devices:
-------------------
/dev/mst/planarized_device_pci

I2C devices:
-------------------
MST                         Serial
/dev/mst/dev-i2c-9          N/A
```

2. Run mlxlink command for the planarized device and it will generate 4 files automatically - one per asic:

```
sudo mlxlink -d /dev/mst/planarized_device_pci --amber_collect <path_to_save_the_amber_file>
```

Example:

```
sudo mlxlink -d /dev/mst/planarized_device_pci --amber_collect /tmp/amber.csv
```

This command creates four amber files – and labels the file name as: **amber_ASIC_0.csv, amber_ASIC_1.csv, amber_ASIC_2.csv, amber_ASIC_3.csv**

Each amber file will have information for a particular split port.

8.3.3 NV Switches

On NVLink Switches there are two mst devices - one per ASIC.

1. Find the MST devices by running the command:

```
# sudo mst status -v
```

Example:

```
MST modules:
------------
    MST PCI module loaded
    MST PCI configuration module loaded
PCI devices:
------------
DEVICE_TYPE             MST                         PCI       RDMA            NET                                     NUMA
Quantum3(rev:0)         /dev/mst/mt54004_pciconf1   03:00.0                                                   -1

Quantum3(rev:0)         /dev/mst/mt54004_pci_cr1    03:00.0                                                   -1

Quantum3(rev:0)         /dev/mst/mt54004_pciconf0   02:00.0                                                   -1

Quantum3(rev:0)         /dev/mst/mt54004_pci_cr0    02:00.0                                                   -1

I2C devices:
-------------------
MST                                     Serial
/dev/mst/dev-i2c-9                      N/A
/dev/mst/dev-i2c-8                      N/A
/dev/mst/dev-i2c-70                     N/A
```

2. Run mlxlink command to collect the amber:

```
# mlxlink -d <mst_device_name> --amber_collect <path_to_save_the_amber_file>
```

Example:

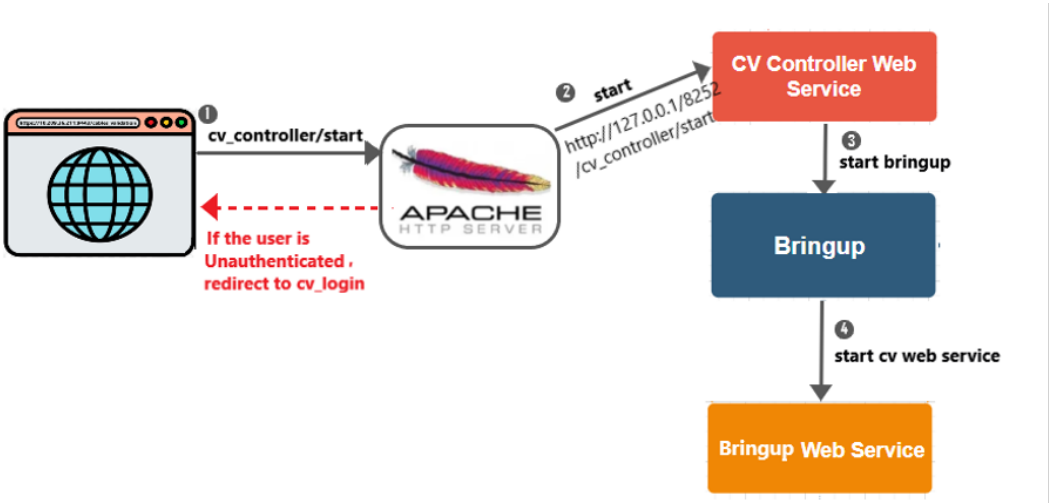
```
mlxlink -d /dev/mst/mt54004_pci_cr0 -amber_collect /tmp/amber_asic1.csv
mlxlink -d /dev/mst/mt54004_pci_cr1 -amber_collect /tmp/amber_asic2.csv
```

9 Cable Validation Controller

The CV Controller is a web service that is automatically started when the CVT container is launched via supervisord. It provides REST APIs to initiate and terminate the bring-up process, enhancing usability and eliminating the need for manual steps.

The CV Controller uses Apache to handle incoming REST API requests. Apache routes any request that includes the `cv_controller` prefix to the CV Controller service.

Authentication is handled through session-based authentication, meaning users must be logged in to access the service.



The CV controller service output will be saved in a new log file called cv_controller_collector.log under /cable_bringup_root/log directory.

9.1 Overriding CV Controller Port

The CV controller uses port 8252 by default. Users can modify this port by updating the CV_CONTROLLER_PORT variable in the config.cfg file.

To make this adjustment, follow these steps:

As Plugin

1. Execute /opt/ufm/scripts/manage_ufm_plugins.sh add -p cablevalidation to add the Cable Validation plugin.
2. Stop the plugin using /opt/ufm/scripts/manage_ufm_plugins.sh stop -p cablevalidation
3. Use vim /opt/ufm/files/conf/plugins/cablevalidation/config.cfg to modify the ' CV_CONTROLLER_PORT ' variable.
4. Update and save the file.
5. Start the plugin again with /opt/ufm/scripts/manage_ufm_plugins.sh start -p cablevalidation.

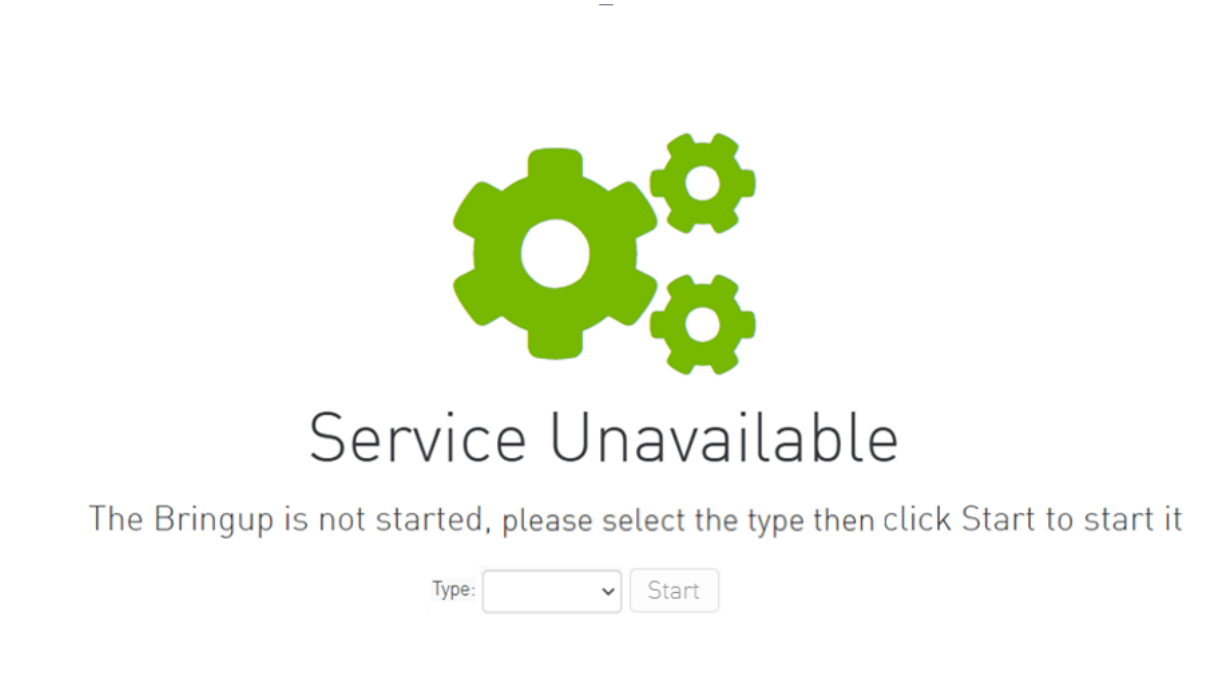
As Standalone

1. Pass the new port as Env by executing **docker run --name cables_bringup -itd --network=host --env CV_CONTROLLER_PORT=<new-port> cables_bringup**

With these changes, the new configuration will take effect, and Apache will run with the updated ports.

9.2 CV Controller GUI

When a user accesses the web page of the Cable Validation Tool, the following screen will appear if the bring-up service has not yet started:



The page displays a button to start the bring-up process, along with a drop-down menu to select the bring-up type (e.g., InfiniBand, XDR, Ethernet).

After starting the bringup service, Users can also stop the bring-up service from the **Service** tab on the **System Admin** page.

System Admin Validation Status: started

Auto Refresh Refresh Interval (sec) 30 Last Refresh 2025-04-06 22:32:36

Summary Report | Agent Status Report | Services | Resource Utilization | Users Management

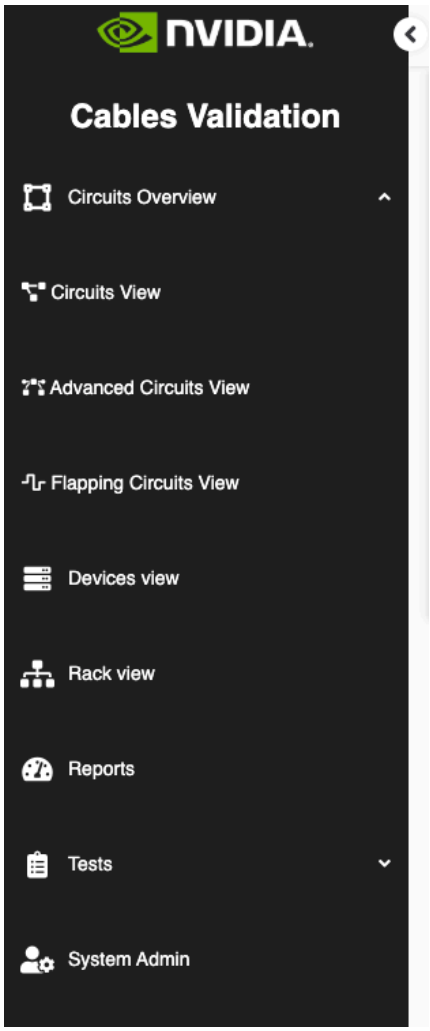
Loaded Topology

Topology File:/cable_bringup_root/data/uploads/topology/ptp/eth-ptp2.xlsx
DC Floor Layout:

Load Topology | Stop Validation | Stop Bringup

10 Bringup GUI

The Cable Validation Tool views facilitate rapid issue identification and correlation, enabling efficient remediation.
The bring-up GUI includes the following pages:"



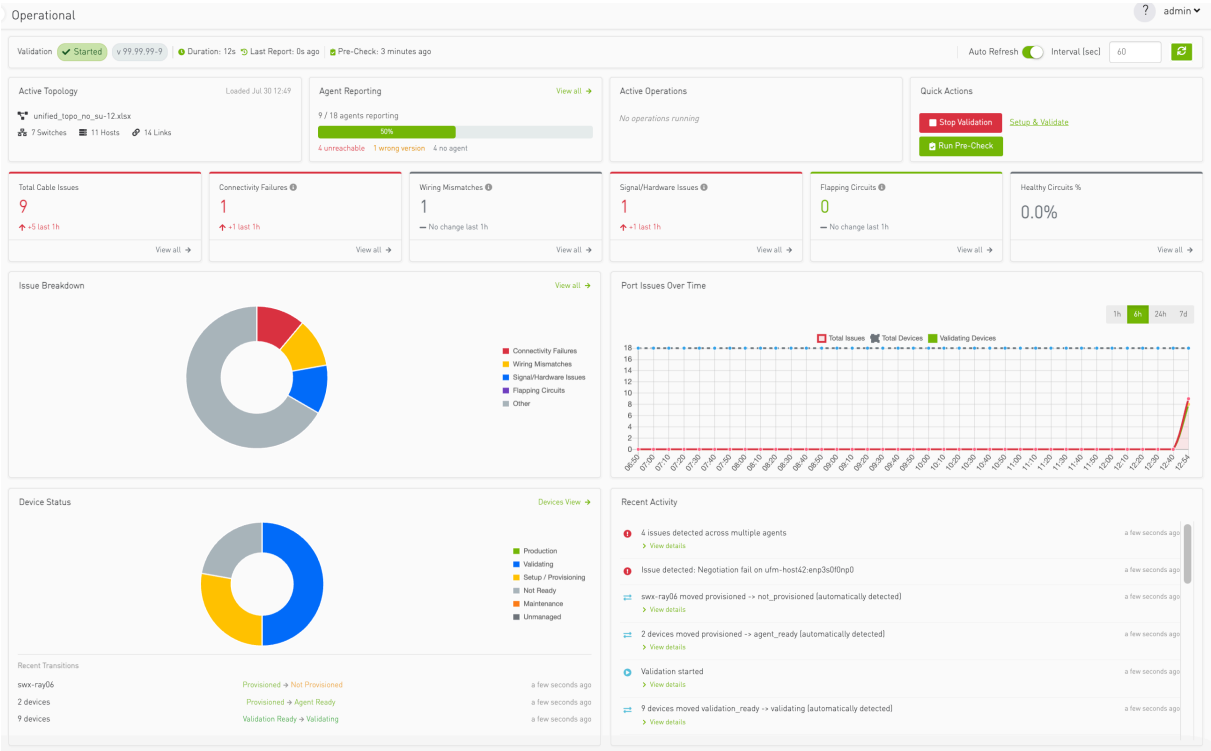
- Circuits View: Combines link data reported by both ends of a circuit (correlation).
- Flapping Circuits View: Flapping circuits with 24h flap history.
- Devices View: View all devices in the loaded topology.
- Rack View: Issues displayed by physical location.
- Reports: Summary (total), list of cable issues.
- Golden BER Tests: Golden amBER test (clear counters, validate compliance).
- Amber Collection Test: Collect Amber files.
- Advanced Flapping Monitoring: Monitors circuit flapping events along with BER (Raw/Eff), temperature, and Tx/Rx.
- System Admin: CVT tool monitoring and debugging (report delays, load topology, manage device access credentials, start/stop validation, etc.).

10.1 Dashboards

The Cables Validation Tool provides two dashboards, each designed for a different operational need. Select a dashboard below to learn more about its features and how to use it.

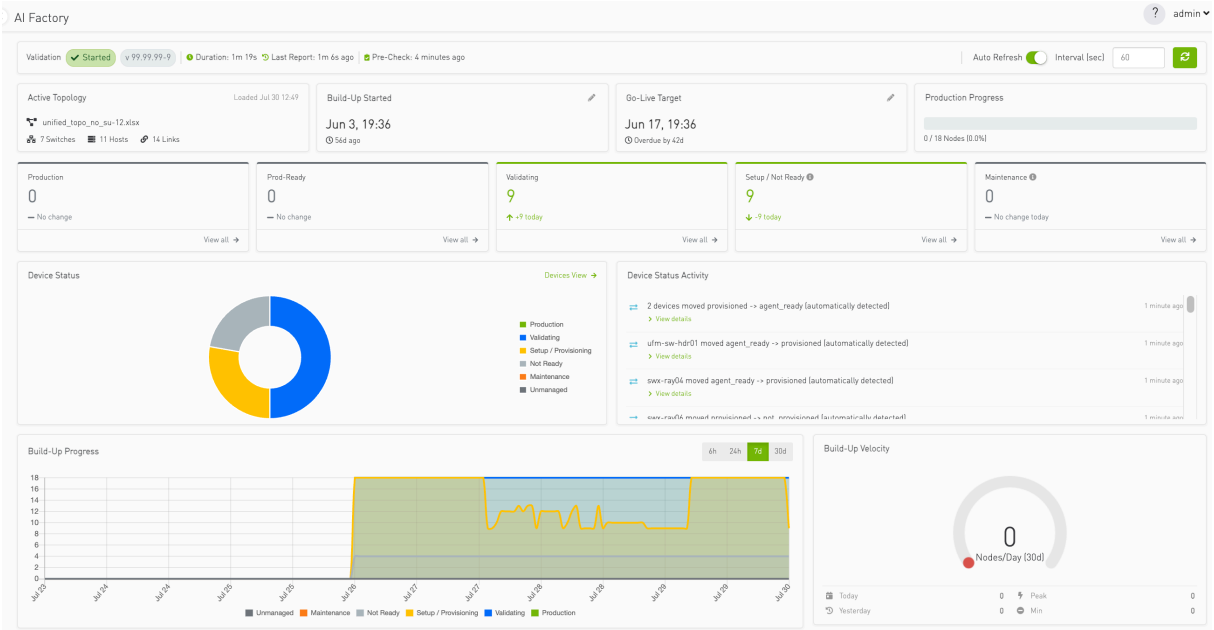
10.1.1 Operational Dashboard

Provides a real-time overview of cable-validation operations and infrastructure health. It displays topology and agent status, active operations, cable-issue KPIs, issue trends, device status, and recent activity. For usage instructions, see the Operational Dashboard guide.



10.1.2 AI Factory Dashboard

Tracks AI Factory build-up progress and readiness for production. It displays build-up milestones, go-live targets, device readiness states, production progress, status activity, and build-up velocity. For usage instructions, see the AI Factory Dashboard guide.



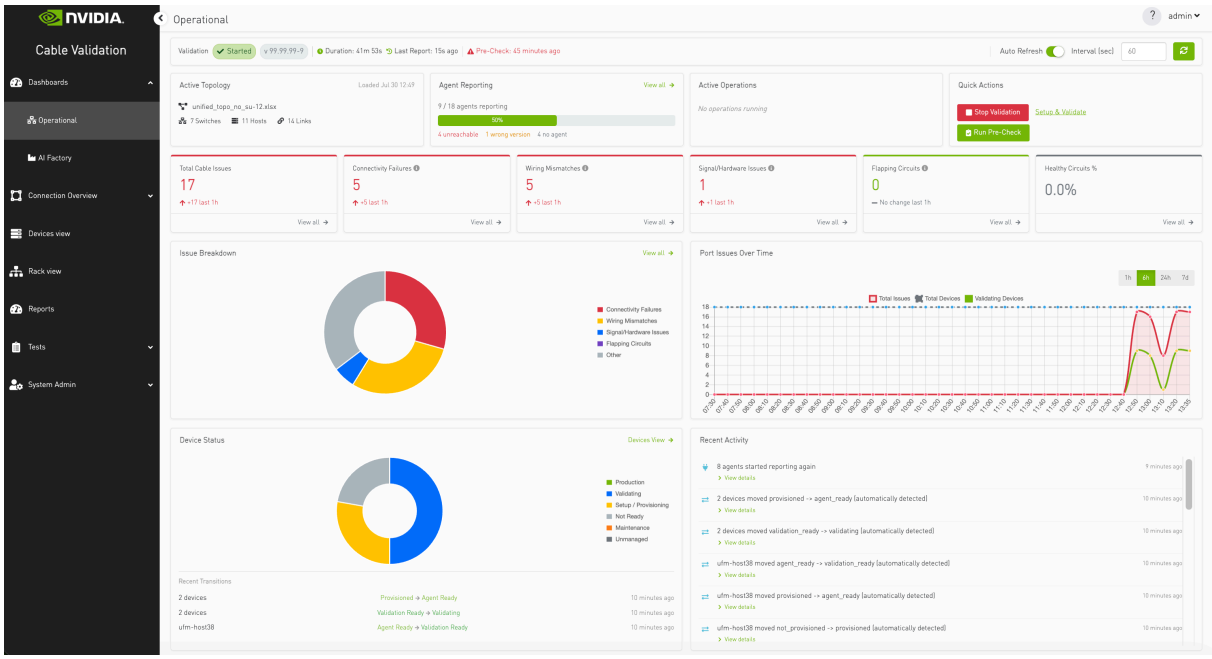
10.1.3 Operational Dashboard

10.1.3.1 Operational Dashboard

10.1.3.1.1 Overview

The Operational Dashboard provides a real-time view of cable-validation activity and fabric health. Use it to confirm that the correct topology is selected, check validation and agent-reporting status, identify cable issues, review device states, and open the relevant detailed views when investigation is needed.

The dashboard is available to all users. Some operational controls and details are visible only to administrators.



10.1.3.1.2 Opening the dashboard

1. In the left navigation, expand **Dashboards**.
2. Select **Operational**.

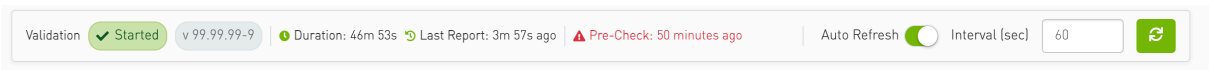
10.1.3.1.3 Recommended workflow

For a quick operational check, read the dashboard from top to bottom:

1. Confirm the **validation state**, CVT version, report freshness, and pre-check freshness.
2. Verify the **active topology**.
3. Check **Agent Reporting** for missing, late, stale, unreachable, or incompatible agents.
4. Review the **KPI cards** for issue counts and circuit health.
5. Use the charts to understand the issue distribution and whether conditions are changing over time.
6. Review **Device Status** and **Recent Activity** for changes that may require attention.
7. Select **View all**, **Devices View**, or **View details** to continue the investigation in a detailed view.

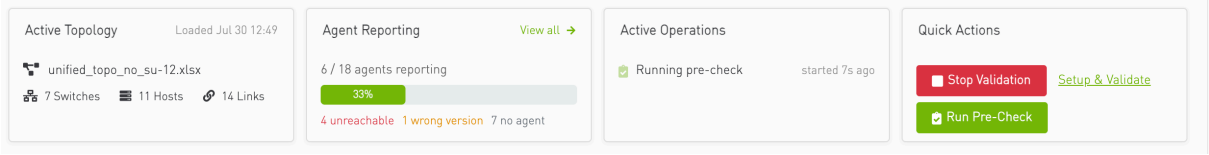
10.1.3.1.4 Validation toolbar

The toolbar at the top of the dashboard shows the current system context and controls how data is refreshed.

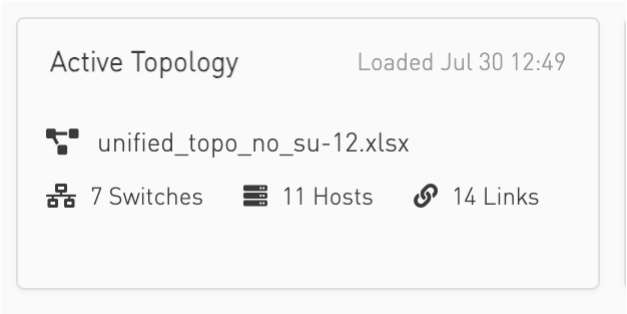


Item	Description and usage
Validation	Shows the current validation state, such as **Started**, **Starting**, **Stopped**, **Stopping**, or **Not Started**.
Version	Shows the installed CVT version.
Duration	When available, shows how long validation has been running.
Last Report	When available, shows how much time has passed since the latest agent-reporting cycle completed. Use this value to confirm that the dashboard data is current.
Pre-Check	Shows when Pre-Check was last run. **Never** or a warning icon indicates that results are missing or older than 30 minutes; consider running Pre-Check again before relying on readiness data.
Auto Refresh	Refreshes dashboard data automatically. Auto refresh starts enabled with a 60-second interval. You can toggle it, change the interval (minimum 60 seconds), or refresh manually.

10.1.3.1.5 Operational summary



10.1.3.1.5.1 Active Topology

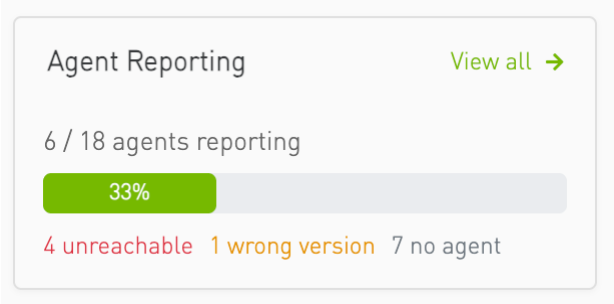


The Active Topology card identifies the topology represented by the dashboard. It shows:

- The loaded topology file name and load time
- The number of switches, hosts, and links when topology statistics are available

If no topology is loaded, the card displays No topology loaded. Go to Setup & Validate to load a topology before starting validation.

10.1.3.1.5.2 Agent Reporting

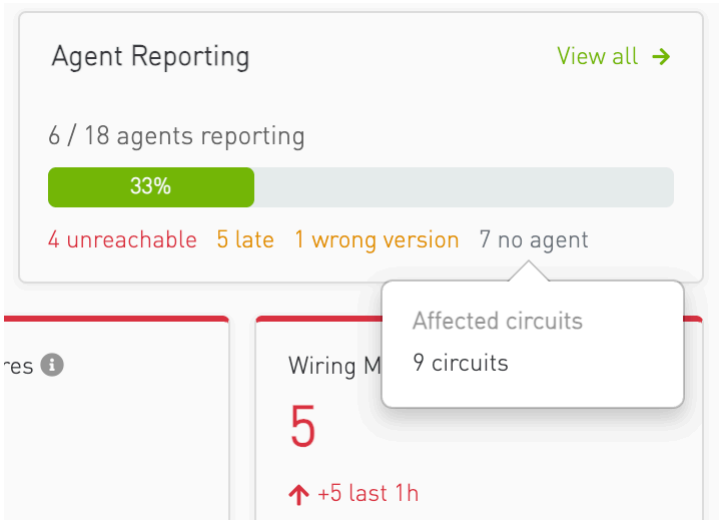


The **Agent Reporting** card shows how many devices are currently reporting validation data compared with the total number of devices. The progress bar provides a quick indication of reporting coverage.

The card can also identify:

- **Unreachable** — agents that cannot be reached
- **Stale** — agents whose reports are no longer current
- **Late** — agents whose latest report is later than expected but not yet stale
- **Wrong version** — agents running an incompatible or unexpected version
- **No agent** — devices without a reporting agent

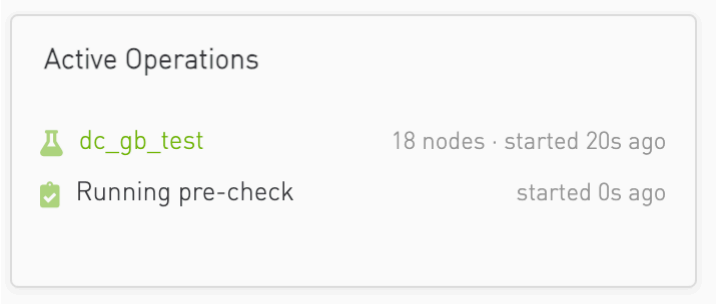
Hover over the **unreachable**, **stale**, or **no agent** counts to see how many circuits are affected.



Administrators can select **View all** to open the detailed agent status reports.

When validation is not running or no agents are reporting, the card displays **No reporting agents**.

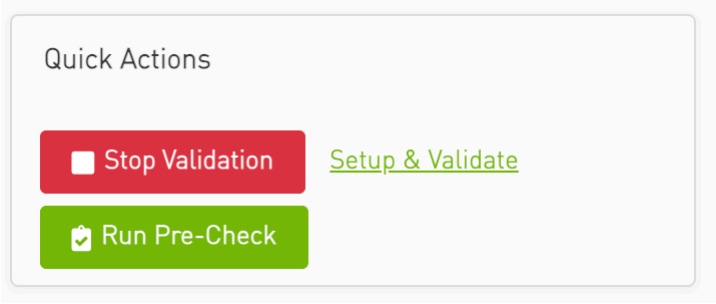
10.1.3.1.5.3 Active Operations



For administrators, Active Operations lists work currently in progress, including validation start or stop operations, topology loading, Pre-Check, agent deployment or removal, and running tests.

The card shows up to two operations directly. If more operations are active, hover over +N more to view the remaining items. A running test name can be selected to open its details.

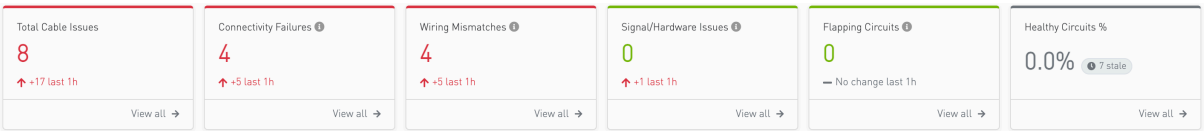
10.1.3.1.5.4 Quick Actions



The Quick Actions card is available to administrators:

Action	When to use it
Start Validation	Starts validation after a topology has been loaded. A progress window shows the operation status.
Stop Validation	Stops the currently running validation. A progress window shows the operation status.
Run Pre-Check	Runs readiness checks for the loaded topology. Use it when the toolbar shows that Pre-Check has never run or its results may be stale.
Setup & Validate	Opens the setup workflow to load or change the topology, manage agents, and prepare validation.

10.1.3.1.6 KPI cards



The KPI row summarizes the most important cable-health measurements.

KPI	What it means
Total Cable Issues	Total issue count represented by the dashboard issue groups. **Extra-cable** findings are not included in these KPI totals.
Connectivity Failures	Links that are down with no signal or have missing/unplugged media.
Wiring Mismatches	Connections to the wrong neighbor or port, or connections whose neighbor cannot be identified.
Signal/Hardware Issues	Signal, temperature, optical power, module part-number, negotiation, or receive-disable problems.
Flapping Circuits	Links that are flapping or error-disabled because of flapping.
Healthy Circuits %	Percentage of total circuits currently reported as healthy. The target is 100%.

For the issue-count cards, the arrow and value show how the count changed during the displayed period:

- An increase is shown as worsening.
- A decrease is shown as improving.
- No change indicates a stable count.

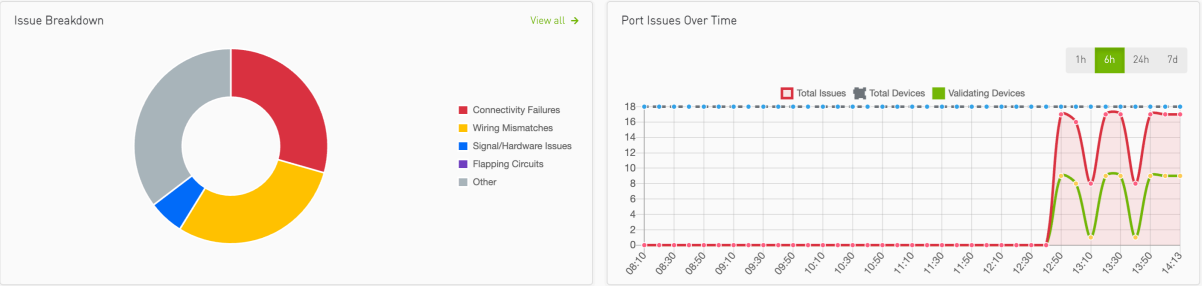
Hover over the information icon on a grouped KPI to see the included issue types.

Select View all to open the related Cable Issues, Flapping Circuits, or Circuits view.

A stale badge beside Healthy Circuits % means that one or both endpoints of some circuits have stale data. Hover over the badge to see the affected count. When this warning is present, the percentage may not reflect the current fabric state.

If validation is not running, KPI values are displayed as a dash because current validation results are unavailable.

10.1.3.1.7 Issue charts



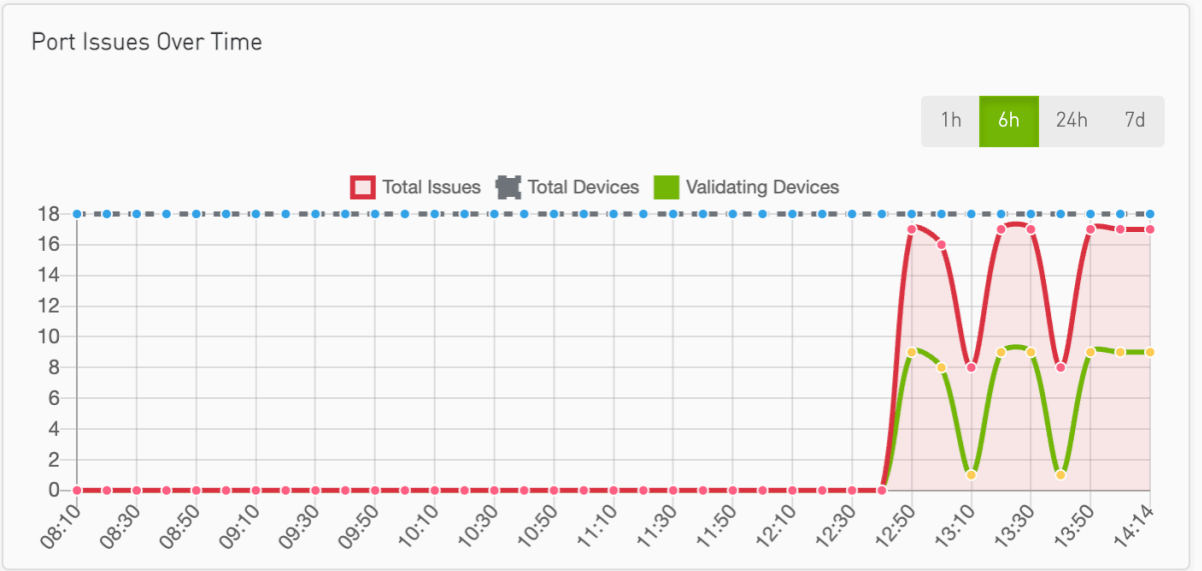
10.1.3.1.7.1 Issue Breakdown

The Issue Breakdown doughnut chart groups current issues into:

- Connectivity Failures
- Wiring Mismatches
- Signal/Hardware Issues
- Flapping Circuits
- Other configuration or administrative states

Use this chart to identify which type of issue contributes most to the current problem set. Select View all to open the Cable Issues view. If validation is running and no grouped issues are detected, the card displays No issues detected.

10.1.3.1.7.2 Port Issues Over Time



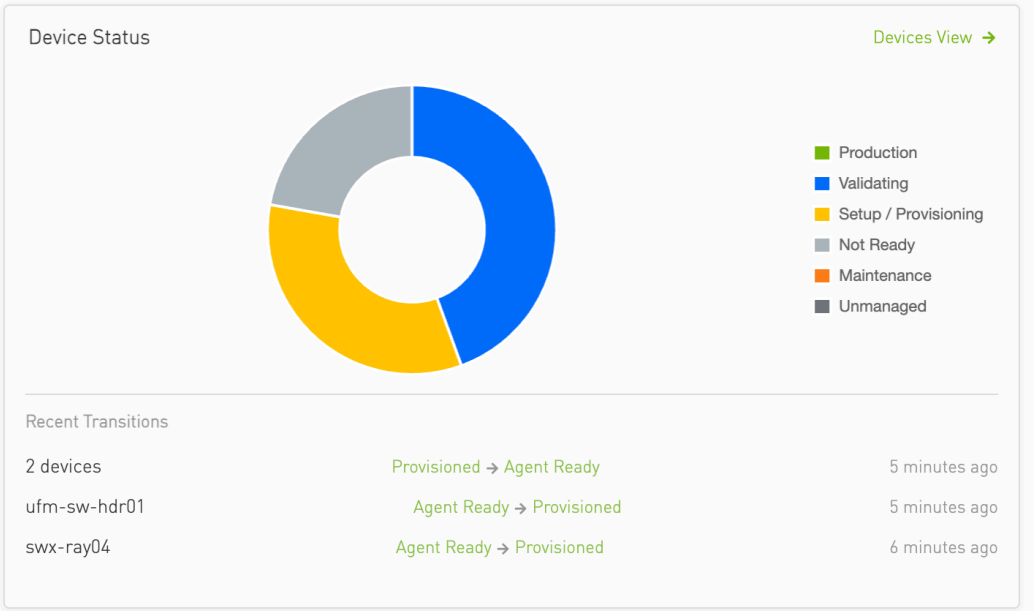
The Port Issues Over Time chart helps determine whether conditions are improving or worsening. It contains three series:

- Total Issues
- Total Devices
- Validating Devices

The default history range is 6h. Select 1h, 6h, 24h, or 7d above the chart to change the displayed period. Hover over a data point to see its exact values.

If validation is not running, the chart displays Validation is not running. If no samples are available for the selected period, it displays No history data available.

10.1.3.1.8 Device Status



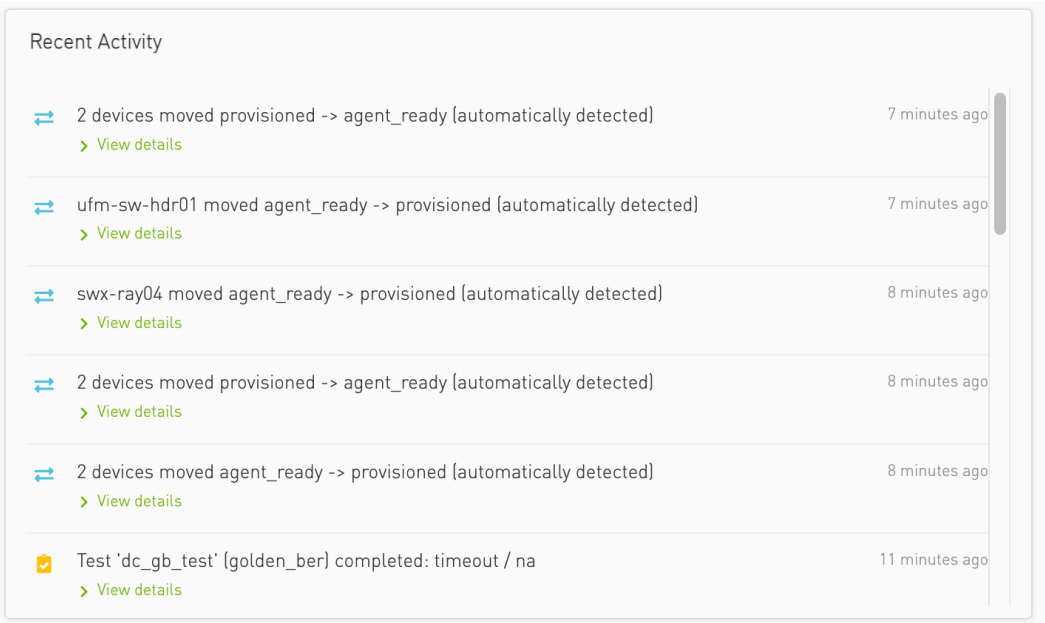
The Device Status card shows how devices are distributed across operational lifecycle groups:

Group	Included states
Production	Production and production-ready devices
Validating	Devices currently validating
Setup / Provisioning	Not provisioned, provisioned, agent-ready, and validation-ready devices
Not Ready	Not installed, installed without power, unsupported, undefined, and validation-only devices
Maintenance	Maintenance and drain/troubleshooting devices
Unmanaged	Devices outside the managed lifecycle

When recent device-state changes exist, the card shows up to three Recent Transitions, including the device name, previous state, new state, and time of the change.

Select **Devices View** to inspect individual devices. If no device data is available, the card displays **No Devices** and prompts you to load a topology.

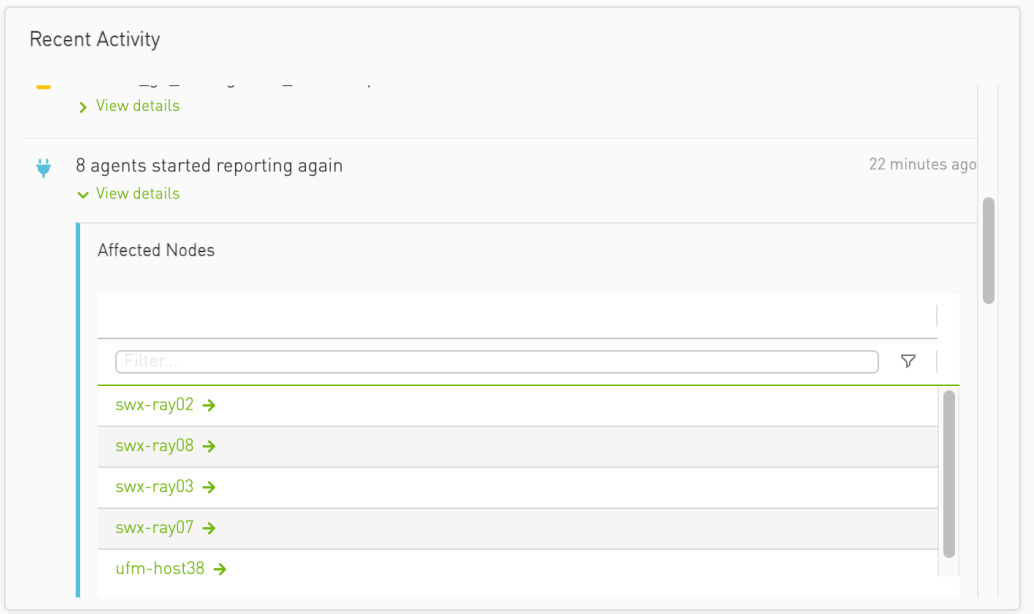
10.1.3.1.9 Recent Activity



The **Recent Activity** card shows the latest events, such as:

- Issues detected or resolved
- Device-state transitions
- Agents going online or offline
- Validation starting or stopping
- Topology loads
- Tests starting or completing
- Agent deployments or removals
- Pre-Check starting or completing

Each event includes a severity indicator, message, and relative time. When **View details** is available, select it to expand the affected nodes, issues, or other event information. The dashboard displays up to 20 recent events.



10.1.3.1.10 Understanding common dashboard states

State	What you will see	Recommended action
No topology loaded	No topology or device data; validation cannot be started from the dashboard.	Open **Setup & Validate** and load a topology.
Validation not running	KPI values show dashes and issue charts report that validation is not running.	If appropriate and you are an administrator, select **Start Validation**.
Pre-Check missing or stale	**Never** or a warning icon appears beside Pre-Check.	Run **Pre-Check** before relying on readiness information.
Agent reporting problems	Unreachable, stale, late, wrong-version, or no-agent counts appear.	Review affected circuits and open the agent status reports if you have administrator access.
Healthy Circuits % has a stale badge	The health percentage is based partly on stale endpoint data.	Investigate stale agents before treating the percentage as current.
No recent activity	The activity feed is empty for the selected scope.	Confirm the scope and refresh the dashboard; an empty feed can also mean no recent events occurred.

10.1.3.1.11 Related page

Return to Dashboards to choose a different dashboard.

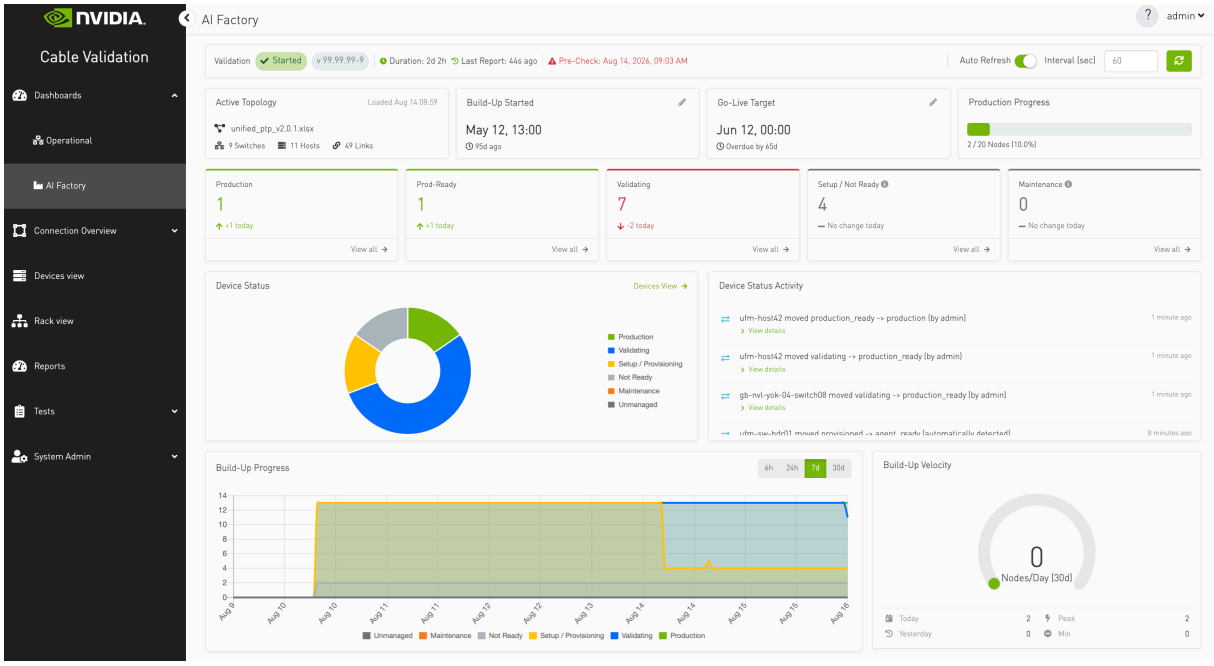
10.1.4 AI Factory Dashboard

10.1.4.1 AI Factory Dashboard

10.1.4.1.1 Overview

The AI Factory Dashboard helps administrators monitor an AI-fabric build-up from topology load through production readiness. Use it to track the build-up schedule, see how many nodes have reached each lifecycle stage, identify devices that are not progressing, and understand the rate at which the fleet is moving into production.

The AI Factory Dashboard is available to administrators under Dashboards → AI Factory.



10.1.4.1.2 Opening the dashboard

1. In the left navigation, expand Dashboards.
2. Select AI Factory.
3. Use the Scope selector in the upper-right corner to choose the workspace, deployment, or view you want to monitor.

The selected scope is applied to summary-based data such as topology context, KPI counts, device status, and device-status activity. Build-up progress and velocity are loaded from the configured build-up dataset.

10.1.4.1.3 Recommended workflow

For a quick build-up review, read the dashboard from top to bottom:

1. Confirm the validation state, CVT version, latest report, and Pre-Check freshness.
2. Verify the active topology and selected scope.
3. Check the Build-Up Started and Go-Live Target dates.
4. Review Production Progress and the status KPI cards to see where nodes are in the lifecycle.
5. Use Device Status Activity to identify recent changes or devices moving backward.
6. Review Build-Up Progress to understand the fleet's movement over time.
7. Check Build-Up Velocity to compare today's production rate with recent performance.
8. Select View all, Devices View, or View details when individual nodes require investigation.

10.1.4.1.4 Validation toolbar

The toolbar at the top of the dashboard shows the current validation context and controls data refresh.

Item	Description and Usage
Validation	Shows the current validation state, such as **Started**, **Starting**, **Stopped**, **Stopping**, or **Not Started**.
Version	Shows the installed CVT version.
Duration	When available, shows how long validation has been running.
Last Report	When available, shows how much time has passed since the latest agent-reporting cycle completed. Use it to confirm that the displayed status is current.
Pre-Check	Shows when Pre-Check was last run. **Never** or a warning icon indicates that results are missing or older than 30 minutes. Run Pre-Check again before relying on readiness information.
Scope	Selects the workspace, deployment, or view represented by summary-based dashboard sections.
Auto Refresh	Refreshes the dashboard automatically. Auto refresh starts enabled with a 60-second interval. You can toggle it, change the interval (minimum 10 seconds), or refresh manually.

10.1.4.1.5 Build-up Summary

10.1.4.1.5.1 Active Topology

The Active Topology card identifies the topology represented by the dashboard. It shows:

- The loaded topology file name and load time
- The active workspace and deployment when a specific scope is selected
- The number of switches, hosts, and links when topology statistics are available

Hover over the information icon beside the scope name to see additional scope details, including the workspace, deployment, and device count.

If no topology is loaded, the card displays No topology loaded. Load a topology before using the dashboard to evaluate build-up progress.

10.1.4.1.5.2 Build-Up Started

The **Build-Up Started** card records when the build-up campaign began. The card displays the start date and time together with the elapsed time. If no start time is set, it displays a dash and **Not started**.

CVT normally records the build-up start when the first topology is loaded. To correct or override it:

1. Select the pencil icon.
2. Choose the required local date and time.
3. Select the check icon to save, or the **X** icon to cancel.

The start time cannot be more than one year in the past.

10.1.4.1.5.3 Go-Live Target

The **Go-Live Target** card shows the planned date and time for reaching production. It also displays:

- The remaining time when the target is in the future
- **Overdue by** and the elapsed duration when the target has passed
- **Not set** when no target exists

To change the target, select the pencil icon, choose a date and time, and save. The target cannot be earlier than the build-up start and cannot be more than one year after it.

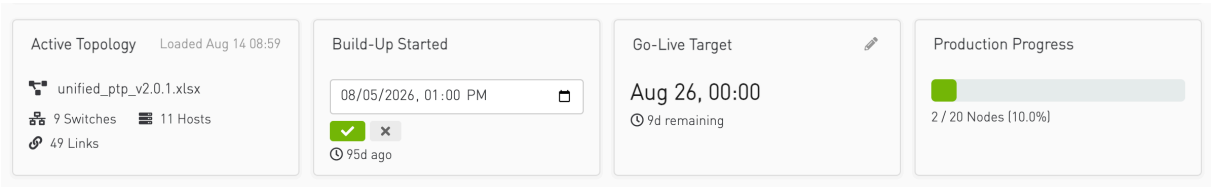
10.1.4.1.5.4 Production Progress

The **Production Progress** card shows the number and percentage of nodes that are either **Production** or **Prod-Ready**.

The calculation is:

- Production progress = Production nodes + Prod-Ready nodes
- Total nodes = Switches + Hosts in the active topology

For example, **89 / 148 Nodes (60.1%)** means that 89 of the 148 topology nodes have reached the production group.



10.1.4.1.6 Status KPI cards

The KPI row shows the current device count in five build-up categories.

KPI	What it means
Production	Nodes currently in the Production state.

KPI	What it means
Prod-Ready	Nodes that have completed validation and are ready to enter production.
Validating	Nodes currently undergoing validation.
Setup / Not Ready	All nodes not currently counted as Production, Prod-Ready, Validating, or Maintenance. Hover over the information icon to see this definition.
Maintenance	Nodes in Maintenance or Drain / Troubleshooting.

Each card includes a View all link to the Devices View.

The arrow and number on a card show how its count changed during the displayed period, normally today:

- An increase in Production, Prod-Ready, or Validating generally indicates forward progress.
- A decrease in Setup / Not Ready generally indicates that nodes are moving forward in the lifecycle.
- An increase in Maintenance indicates that more nodes have moved off the production path and may require attention.
- No change indicates that the count remained stable during the period.

Use the trend together with the current value. For example, a high Production count with no change may be healthy, while a large Setup / Not Ready count with no change can indicate stalled progress.

10.1.4.1.7 Device Status

The Device Status doughnut chart shows how nodes are distributed across six lifecycle groups:

Group	Included states
Production	Production and production-ready nodes
Validating	Nodes currently validating
Setup / Provisioning	Not provisioned, provisioned, agent-ready, and validation-ready nodes
Not Ready	Not installed, installed without power, unsupported, undefined, and validation-only nodes
Maintenance	Maintenance and drain/troubleshooting nodes
Unmanaged	Nodes outside the managed lifecycle

Use the chart to locate the largest non-production group. Select Devices View to inspect individual nodes. If no device data is available, the card displays No Devices and prompts you to load a topology.

Recent transitions are shown in the adjacent Device Status Activity card rather than inside this chart.

10.1.4.1.8 Device Status Activity

The Device Status Activity card shows up to 30 recent device-transition events for the selected scope. Use it to identify nodes that moved between lifecycle states, including transitions into Production, Maintenance, or a not-ready state.

Each event includes a message and relative time. Select View details when available to expand information about affected nodes or the status change. If there are no recent transitions, the feed displays No recent activity.

10.1.4.1.9 Build-Up Progress

The Build-Up Progress chart shows how device counts changed over time across these stacked series:

- Unmanaged
- Maintenance
- Not Ready
- Setup / Provisioning
- Validating
- Production

The default history range is 7d. Select 6h, 24h, 7d, or 30d to change the displayed period. Hover over the chart to see the count for every lifecycle group at a specific time.

Use this chart to answer questions such as:

- Is the Production area growing consistently?
- Are nodes moving from Setup / Provisioning into Validating?
- Is the Not Ready or Maintenance population increasing?
- Did progress stop during a particular period?

If no historical samples are available for the selected period, the chart displays No build-up data available.

10.1.4.1.10 Build-Up Velocity

The Build-Up Velocity gauge shows the average rate at which nodes reach the Production group, measured in nodes per day. When available, the gauge label also shows the number of days included in the calculation.

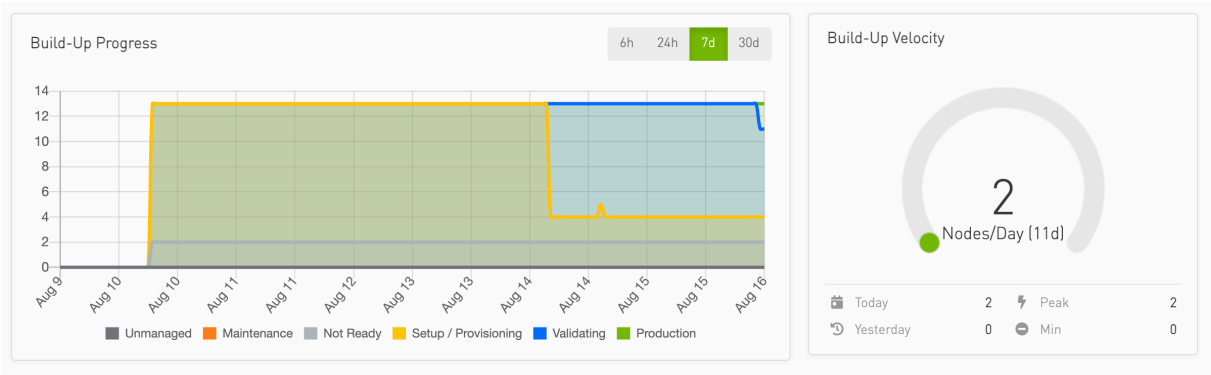
The supporting statistics are:

Statistic	Meaning
Today	Nodes that reached the Production group today
Yesterday	Nodes that reached the Production group yesterday
Peak	Highest daily production increase during the measured window
Min	Lowest daily production increase during the measured window

The gauge color provides a quick comparison:

- Green — today's rate is positive and at least as high as yesterday's
- Amber — today's rate is positive but below yesterday's
- Red — no nodes have reached Production today

Use the color as a current-rate indicator, not as a complete health assessment. Consider the build-up start time, remaining non-production nodes, and the progress chart before deciding whether the campaign is on track.



10.1.4.1.11 Understanding Common Dashboard States

State	What you will see	Recommended action
No topology loaded	No topology or device data; Production Progress shows no nodes.	Load the intended topology before evaluating build-up progress.
Build-up not started	Build-Up Started shows a dash and **Not started**.	Verify the topology, then set or correct the start time if needed.
Go-live target not set	Go-Live Target shows a dash and **Not set**.	Set a target so remaining or overdue time can be monitored.
Go-live target overdue	The card displays **Overdue by** with a duration.	Review stalled nodes, recent transitions, progress history, and velocity.
Large Setup / Not Ready count	Many nodes have not reached Validating, Prod-Ready, or Production.	Open Devices View and investigate node readiness and provisioning.
Maintenance count increasing	The Maintenance KPI shows an upward, unfavorable trend.	Review Device Status Activity and inspect affected nodes.
No build-up history	Build-Up Progress displays **No build-up data available**.	Confirm that status data has been collected and try another time range.
No recent activity	Device Status Activity is empty.	Confirm the selected scope and refresh; there may also have been no recent transitions.
Velocity is red	Today's production increase is zero.	Check whether work has paused or whether nodes are stalled in another lifecycle group.

10.1.4.1.12 Related Pages

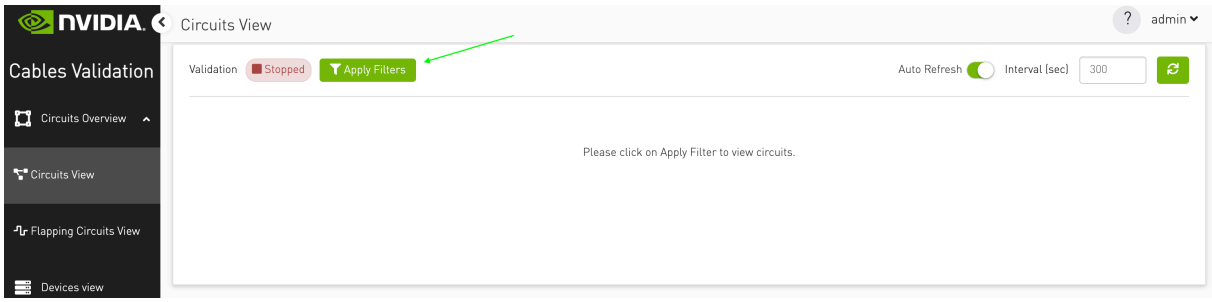
- Return to Dashboards to choose another dashboard.
- See the Operational Dashboard for cable-validation health, issue trends, agent reporting, and recent operational activity.

10.2 Circuits Overview

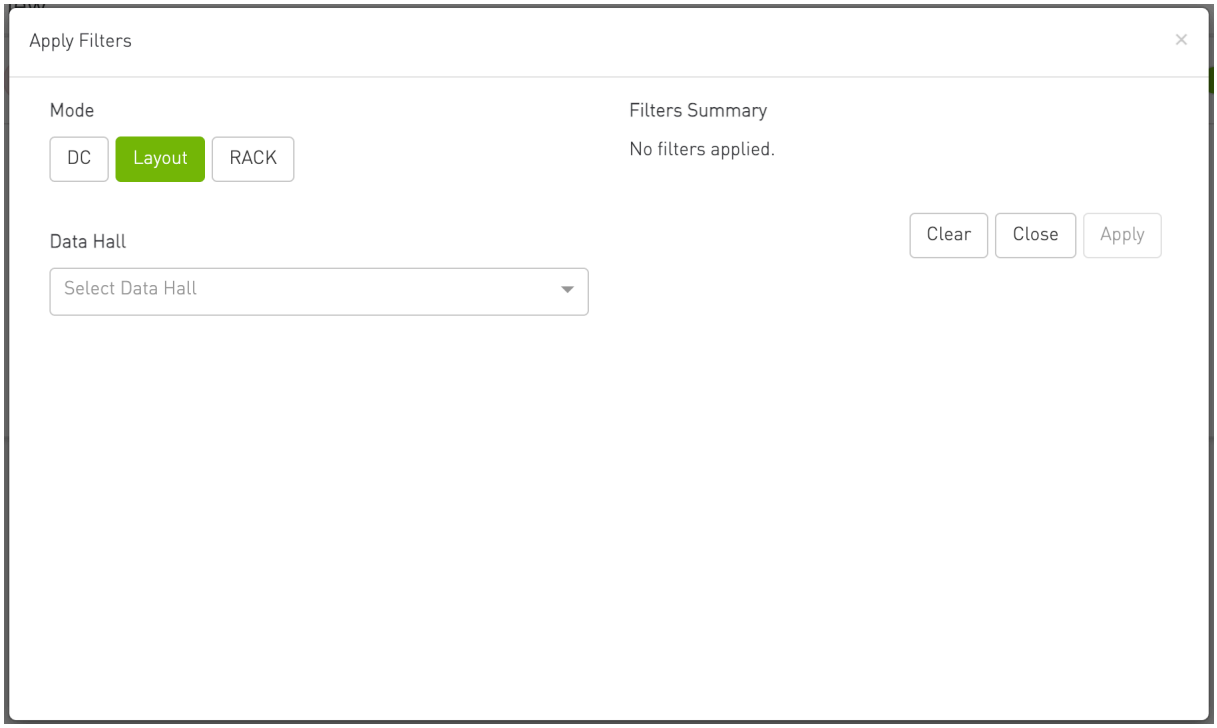
10.2.1 Circuits View

The **Circuits view** in the CVT provides a comprehensive overview of all circuits, including their properties, validation status, and a complete history of issues and remediation actions. This ensures robust issue tracking, operational transparency, and system reliability, and it also allows you to explore the connection details between two endpoints, labeled **A** and **Z**.

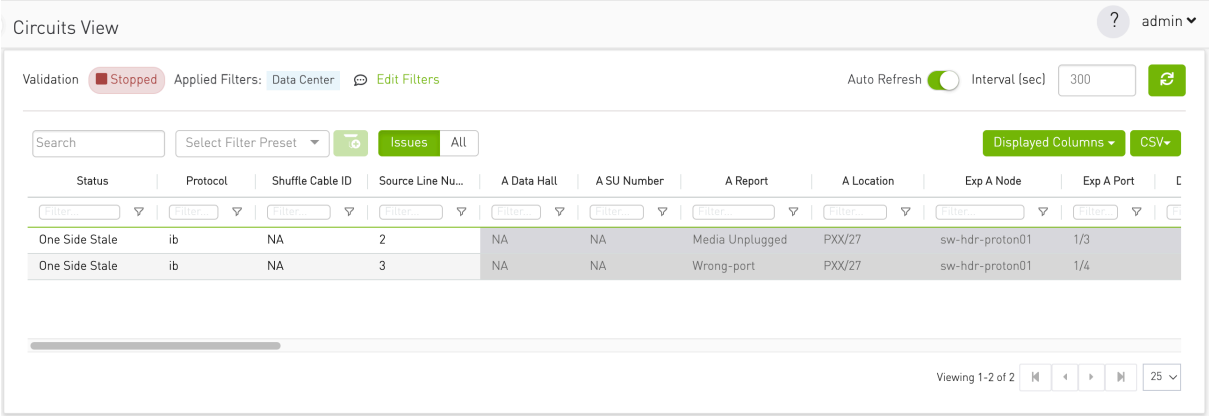
When you open the page, you'll see a message asking you to apply a filter to view the circuits.



Click "**Apply Filters**" to open the filter window, choose your preferred filter, and apply it. For more guidance, see the **Resource Filter** page.



Once your filter is applied, all matching circuits will appear in a table.



Each circuit includes the following information:

1. Status

The status reflects the overall health of the circuit and can have one of the following values:

- **Fail**: Indicates that there are issues with one or both endpoints.
- **Pass**: Indicates that there are no issues with either endpoint.
- **Incomplete**: Indicates no issues with the endpoints, but the port status is not yet ready.
- **One Side Stale**: One of the endpoints is stale.
- **Both Sides Stale**: Both endpoints are stale.

2. **Protocol**: circuit protocol type ib, ethernet or nvlink.

3. Shuffle Cable ID

4. Source Line Number

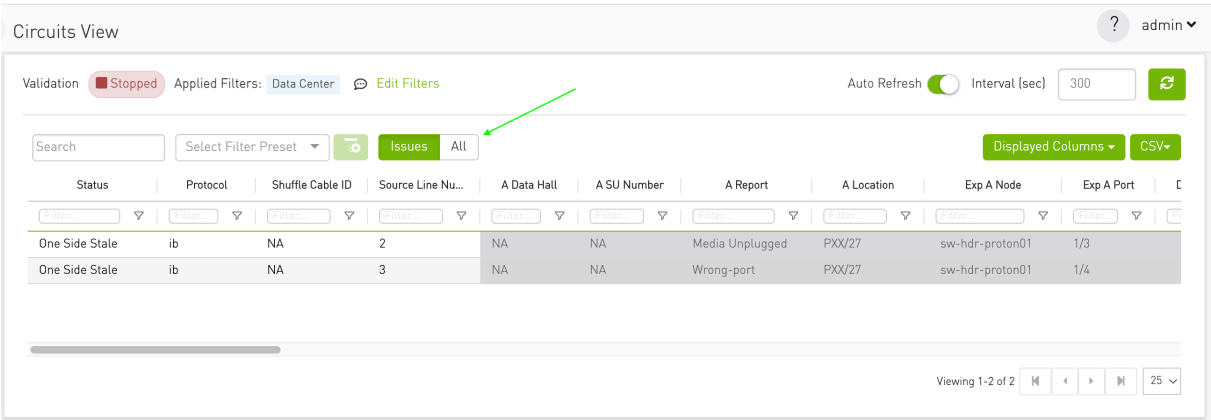
5. Endpoint Information

Detailed information about each endpoint includes:

- **Report**: the reported syndrome
- **Location**: Specifies the endpoint's location, Rack, and Unit.
- **Node and Port Details**: information about the nodes and the associated ports.
- **Transceiver Information**: Includes details such as:
    - Transceiver part number, hidden by default.
    - Transceiver serial number, hidden by default.
    - Firmware version, hidden by default.
    - Reinsert count, hidden by default.
    - Swap count, hidden by default.
    - Connection status: Displays **false** if one of the endpoints has a "No Transceiver" issue.
- **Port Status**: Indicates the current state of the port:
    - **Down**: If any of the following conditions are present:
        - "No Transceiver"
        - "Link Down, No Signal"
        - "ErrDisable - Flap"
        - "Admin Down"

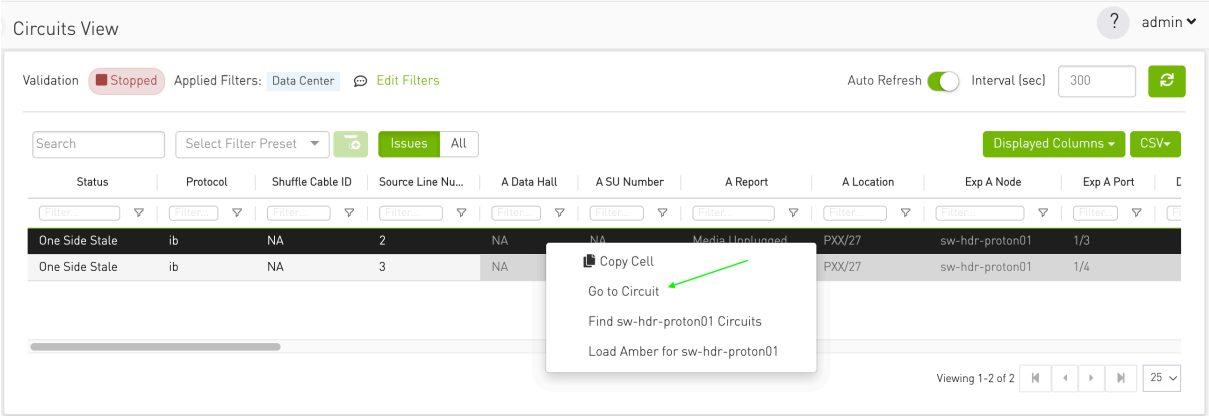
        - "ErrDisable - Rx"
        - "Negotiation Failure"
    - Up: When none of the above conditions apply.
- Signal Status: Shows transmission and reception power levels (Tx/Rx power lanes).
- BER Counters: Provides bit error rate (BER) statistics, such as raw BER and effective BER., These statistics are only available to Admin users and are hidden by default.
- Traffic Counters: Includes traffic statistics, such as errors, drops, and byte counts for incoming and outgoing traffic.
- Physical Port
- Custom NIC Name
- NIC OS Name
- RDMA Name: hidden by default.
- PCI Address: hidden by default.
- Shuffle Cable Information:
    - A Shuffle Cable Connector
    - A Shuffle Cable MPO Connector
    - A Module Part Number
- Recommended Action: Suggests actions to resolve any identified issues.
- Time Since Last Clear: Indicates the time since the counters were last cleared.
- Report Status: Indicates the timeliness of the report, with the following values:
    - No Report: The report has not been updated yet(the associated endpoint will be grayed out).
    - Stale Report: The report has not been updated within the last 15 minutes (the associated endpoint will be grayed out).
    - Latest Report: The report was updated within the last 15 minutes.

By default, only circuits with a Fail status (unhealthy circuits) are displayed. However, users can view all circuits by selecting the All option.



To select the peer-port cable issue user can right-click on the specified circuit and select Go To

Circuit to see the combined circuit details and remediation actions.



10.2.1.1 Rx/Tx Power Lane

In the realm of data transmission, understanding the power values for each lane in the port is crucial for maintaining optimal performance and ensuring efficient signal transmission.

Rx/Tx Power Values for Each Lane

The Rx and Tx power values are essential metrics that provide insight into the performance of each lane in a port. These values are especially significant in high-speed data transmission environments, where maintaining the correct power levels is critical for data integrity and system reliability.

NDR Switches

For NDR switches, the power values for each lane in the port are provided, with each cage containing two ports. The relevant lanes for each port can be identified using the _Lanes_Used' counter. This counter plays a vital role in determining which lanes are active and relevant for data transmission.

The 'Module_Lanes_Used' counter is a binary indicator that specifies the active lanes in a particular module. For instance:

- If the value is '1_1_1_1_0_0_0_0', lanes 4-7 should be considered.
- If the value is '0_0_0_0_1_1_1_1', lanes 0-3 should be taken into account.

The Rx and Tx power values for these lanes are denoted as rx_power_lane_n and tx_power_lane_n, where 'n' can range from 0 to 7, depending on the active lanes.

InfiniBand Technology

When it comes to InfiniBand technology, the approach to handling Rx and Tx power values differs slightly. InfiniBand typically provides power values only for lanes 0-3. As a result, the 'Module_Lanes_Used' mask value is disregarded in this context.

Testing and Verification

Step 1: Identifying Active Lanes

Using the 'Module_Lanes_Used' counter, identify the active lanes for each port. For NDR switches, deter 'Module mine whether lanes 0-3 or 4-7 are active based on the counter's value.

Step 2: Recording Power Values

Record the Rx and Tx power values for the active lanes. Ensure that the values are accurately captured and correspond to the designated lanes.

Step 3: Cross-Verification

Cross-verify the recorded power values with the expected values for the active lanes. This step is crucial for identifying any discrepancies or anomalies in the power readings.
Step 4: Ignoring Irrelevant Lanes
For InfiniBand, ignore the 'Module_Lanes_Used' mask value and focus solely on lanes 0-3. Verify that the power values for these lanes are accurate and within the acceptable range.

10.2.1.2 Circuit History and Remediation

Circuit History and Remediation provides comprehensive tracking of cable circuit issues and the actions taken to resolve them over time. The system automatically monitors the health of your network circuits and maintains a historical record of all issues, user interventions, and resolutions. All actions performed on a circuit—such as status updates or property changes—are automatically logged for traceability.

10.2.1.2.1 Issues and Remediation Workflow

10.2.1.2.1.1 Automatic Issue Detection

CVT continuously monitors circuits via validation reports from network switches. When an issue is detected:

1. The system automatically records the issue with a timestamp.
2. Details include the circuit ID, affected endpoint, issue type (syndrome), and recommended corrective action.
3. Issues are tracked per-circuit, providing a complete history for each circuit.

10.2.1.2.1.2 Issue Resolution Tracking

The system intelligently tracks issue lifecycle:

1. When a previously detected issue is no longer present in subsequent validation reports, the system automatically records a resolution event.
2. This helps teams understand how long issues persist and track the effectiveness of remediation efforts.

10.2.1.2.1.3 User Remediation Records

Network operators and technicians can document their troubleshooting and repair actions:

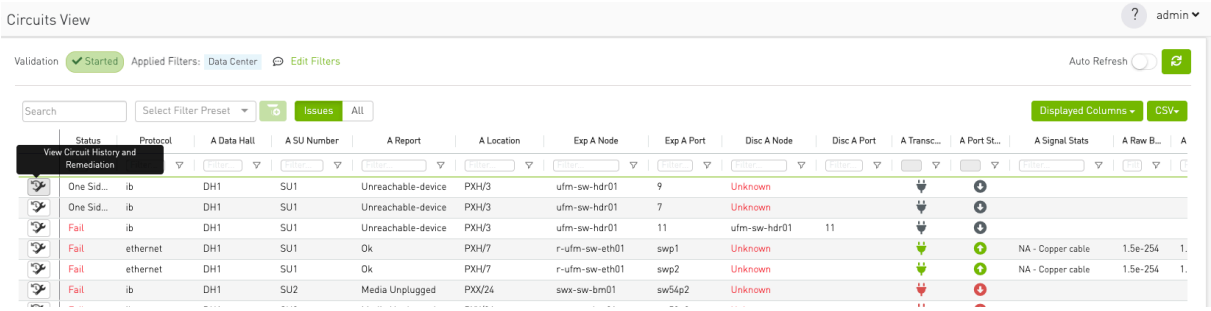
1. Add remediation records through the CVT interface or API.
2. Each remediation record captures:

-
    - Who performed the action
    - When it was performed
    - What circuit and endpoint was affected
    - Description of the action taken (e.g., "Replaced faulty cable", "Reseated connection", "Replaced transceiver")

10.2.1.2.2 Accessing History and Remediation

To review the history or manage remediation for a circuit:

1. Navigate to the Circuits table.
2. Click the history button in the first column for the desired circuit.
3. This opens the Circuit History and Remediation page for that specific circuit.

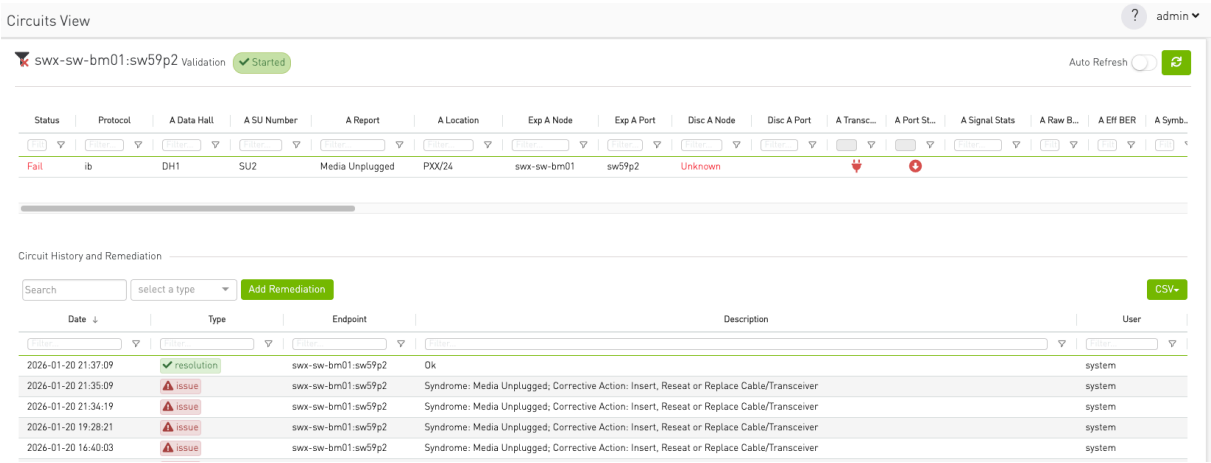


10.2.1.2.2.1 Circuit History Page

On this page, you can view circuit details along with its complete history. The “History” section displays a chronological list of actions, including:

- Date and time of each change
- The issue type
- Endpoint
- Description of the action (e.g., “Status changed from Fail to Pass”)
- The user who performed the action

This audit trail ensures compliance and provides a clear record of all circuit modifications.



10.2.1.2.2.2 Record Types

The circuit history system tracks three types of records:

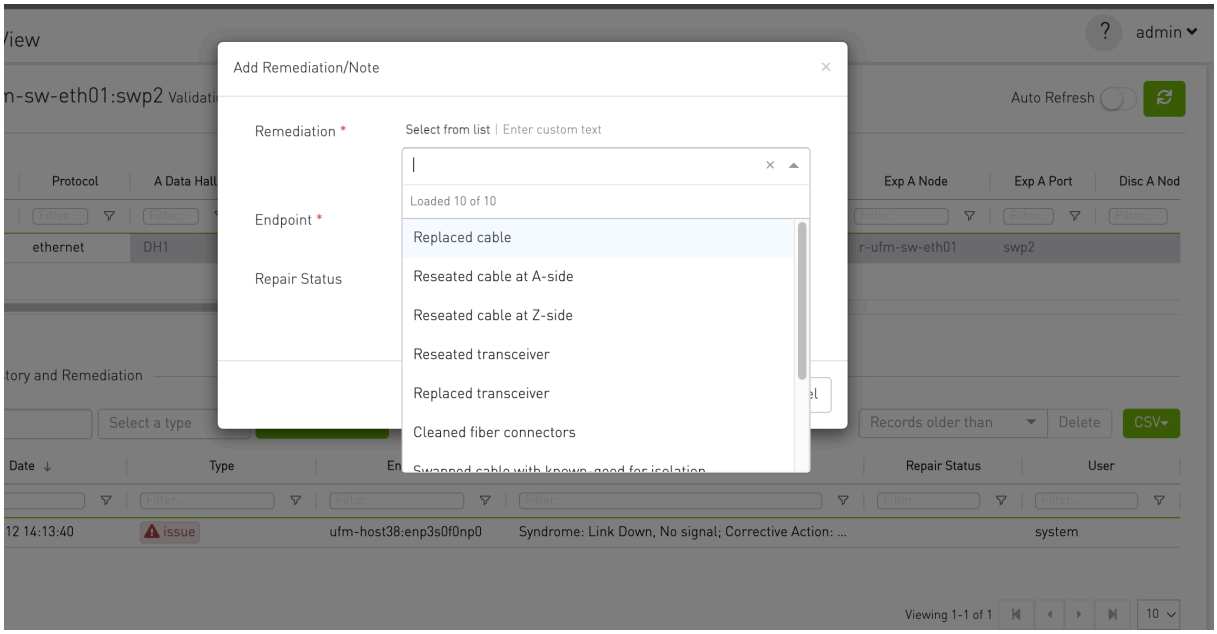
- Issue: Automatically recorded when a validation problem is detected.
- Remediation: User-documented actions taken to fix issues.
- Resolution: Automatically recorded when a previously detected issue is no longer present.

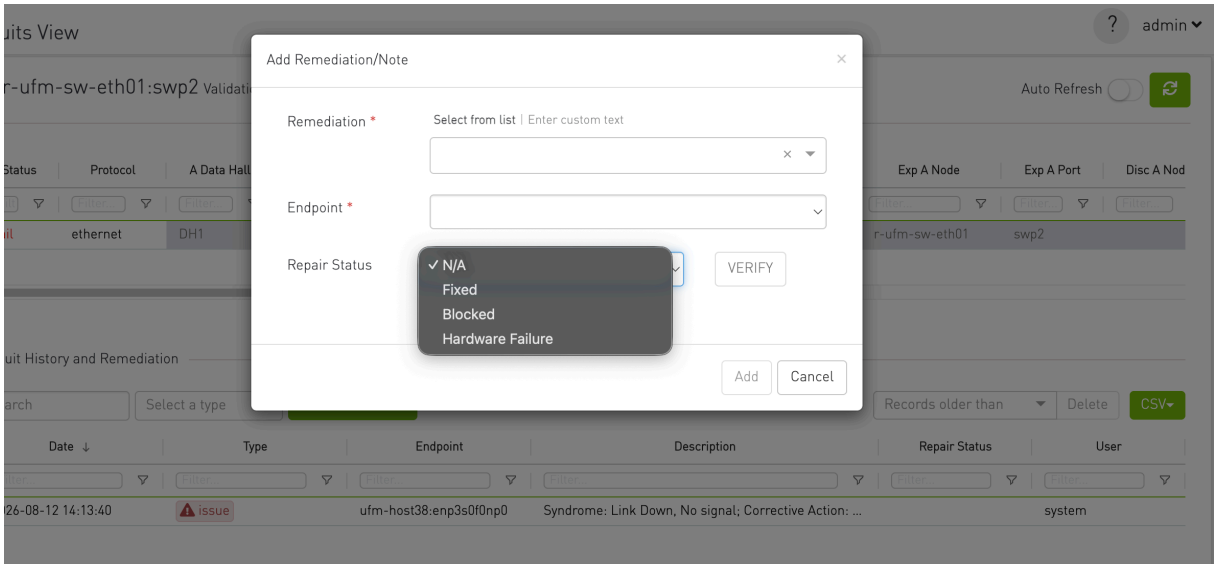
10.2.1.2.2.3 Adding Remediation

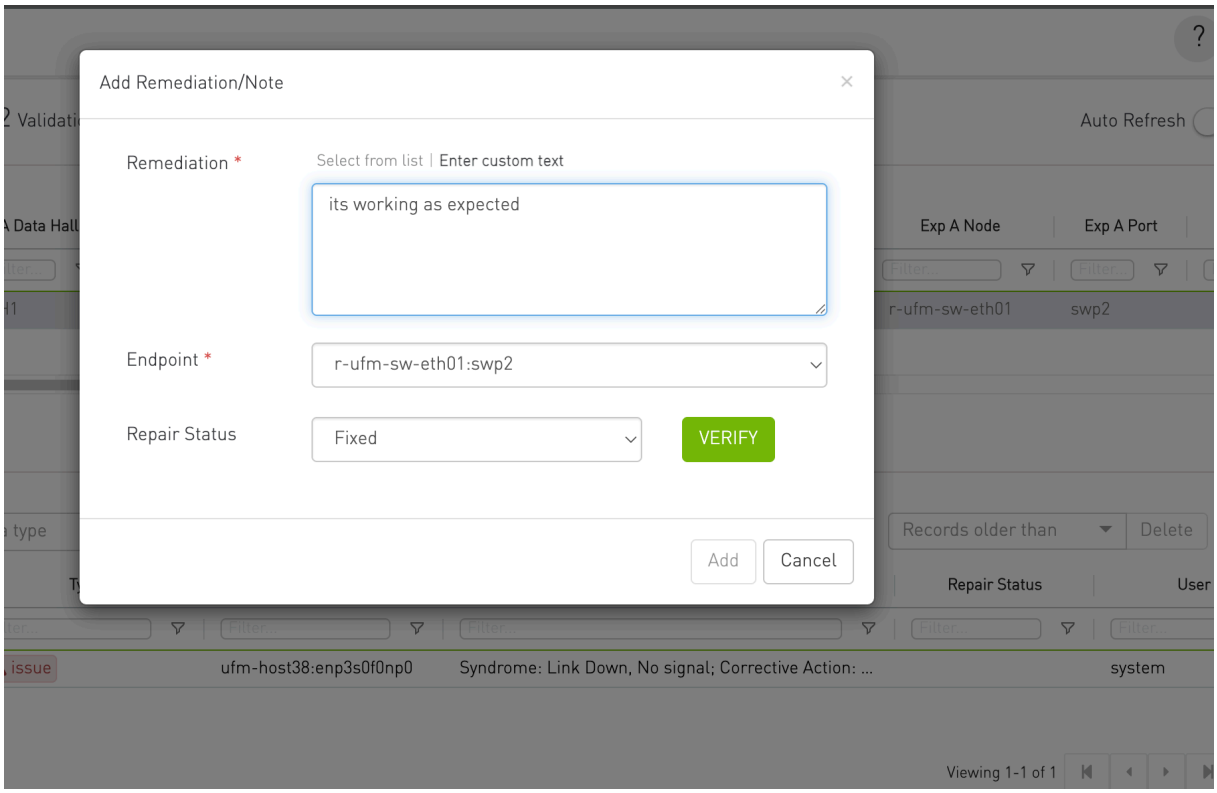
To document troubleshooting and repair actions:

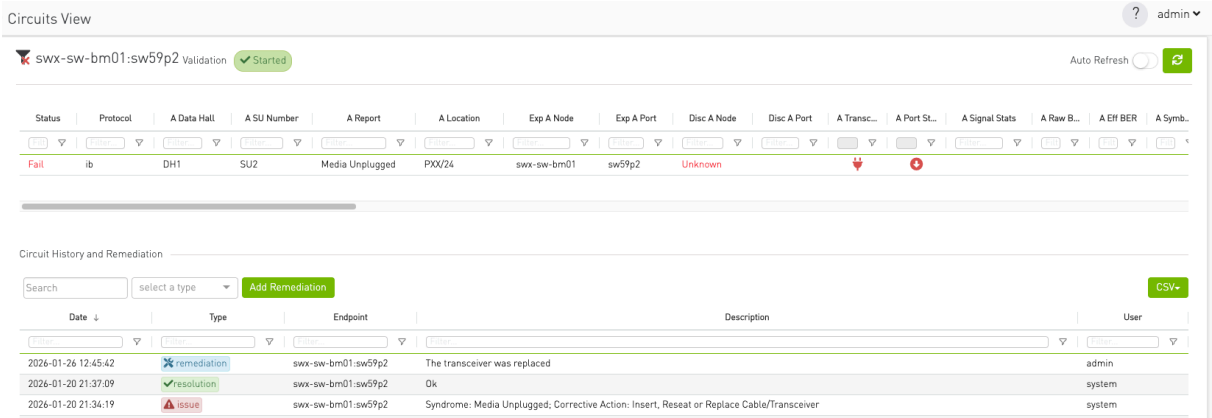
1. Click the 'Add Remediation' button on the Circuit History page.
2. Identify the circuit and endpoint being worked on.
3. Enter a clear description of the remediation action or choose from select list .
4. choose the repair status (Fixed, Blocked,Hardware Failure)
5. in case Fixed is selected , the verify action need to be executed to make sure the endpoint verified and no issue in that endpoint
6. Submit the remediation record.

The system automatically associates your username and timestamp with the record.









10.2.1.2.2.4 Filtering and Searching

Users can filter circuit history by:

1. **Record Type**: View only issues, only remediations, or only resolutions
2. **User**: See all actions performed by a specific team member
3. **Time Range**: Focus on recent activity or historical patterns

10.2.1.3 Amber Simulation

Amber simulation supports the manual injection of amBER files into the Circuits View. These CSV files, collected by NVIDIA's *mlxlink* tool from PCI devices, contain port statistics and counters.

Key Counters Included:

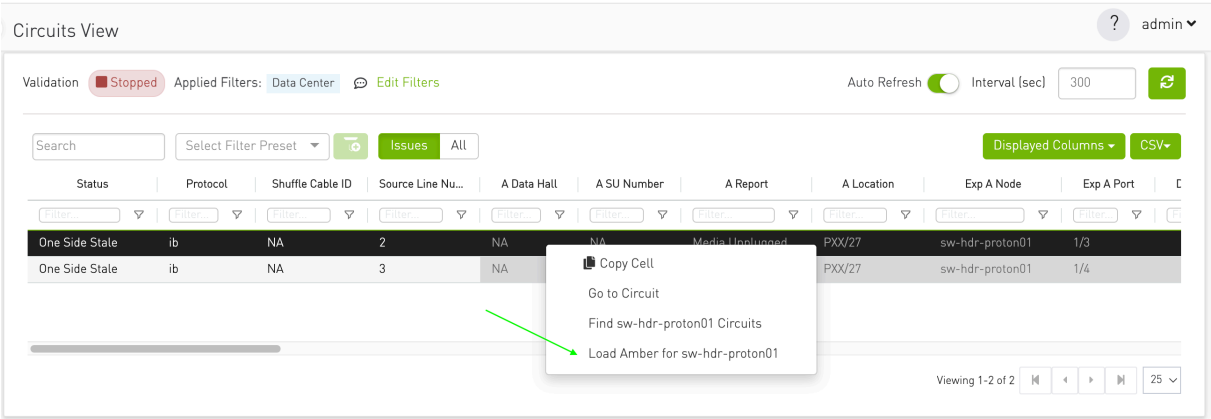
1. **BER Counters** – Indicate bit errors on the link.
2. **RX/TX Power Levels** – Provide power metrics to assess link activity.
3. **Module Temperature**
4. **Module Status** – Indicates whether the module is plugged in or unplugged.
5. **And more...**

This functionality allows users to load amBER files for one or both sides of a circuit, providing visual indicators and filtering options for improved data analysis and management.

It is particularly useful in the following scenarios:

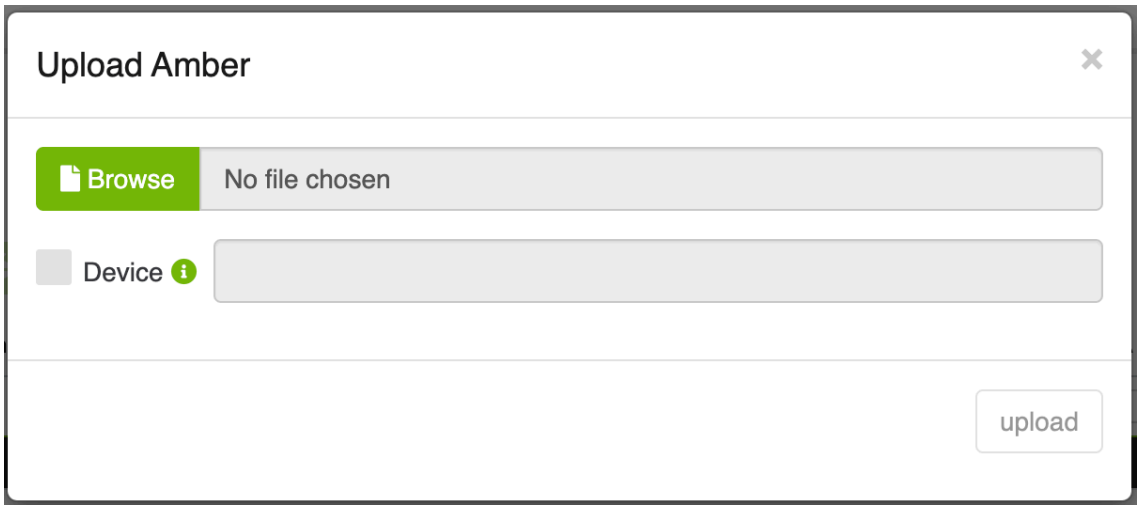
- **No Agent Installed or No Connectivity**: When the node lacks an installed agent or has no active connection.
- **Historical Analysis**: Users can analyze past events that are no longer present in live data by loading an amBER file captured during the event.
- **Debugging and Simulation**: amBER files can be edited to simulate and debug corner-case scenarios.

To simulate any endpoint, the user can right-click on it and select **'Load Amber for <endpoint name>'**. This action is only available to non-cabler users.

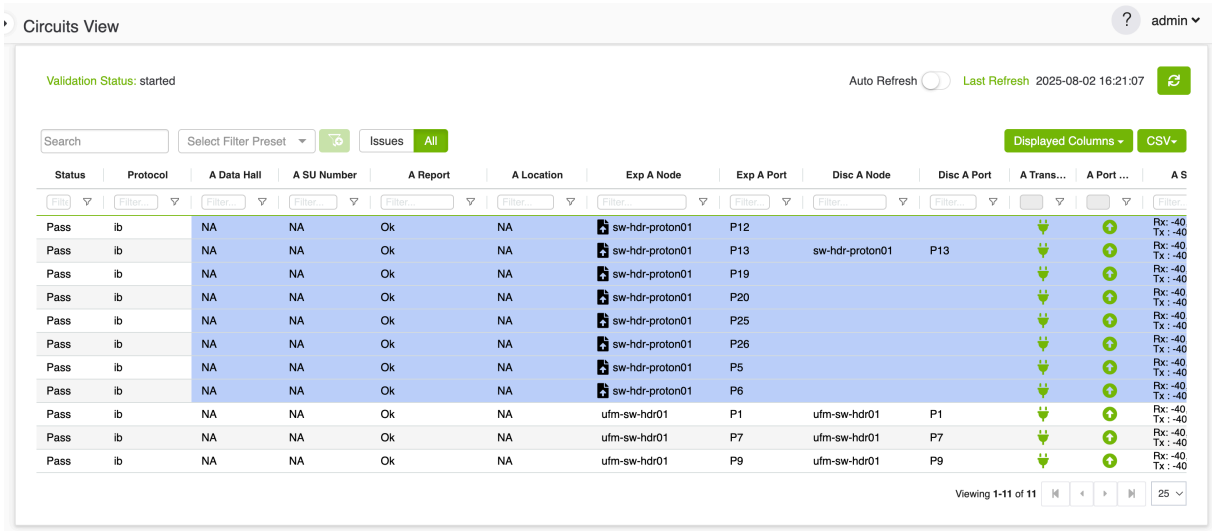


This will open a modal to upload the amBER file. The modal contains two input fields:

1. **Amber File** – Accepts a CSV file or a compressed CSV file (if the file is large).
2. **Device Name** – Required for nodes with multiple devices (e.g., hosts and NVL switches).

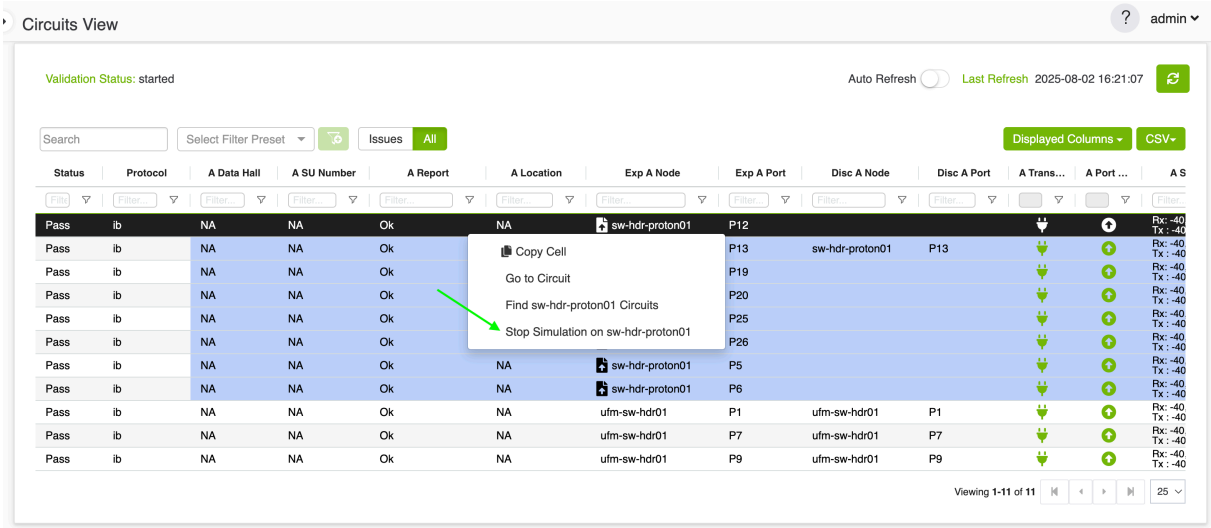


After loading the amBER file, the simulated endpoint row will be highlighted in **blue**.



Status	Protocol	A Data Hall	A SU Number	A Report	A Location	Exp A Node	Exp A Port	Disc A Node	Disc A Port	A Trans...	A Port ...	A S
Pass	ib	NA	NA	Ok	NA	sw-hdr-proton01	P12					Rx: -40 Tx : -40
Pass	ib	NA	NA	Ok	NA	sw-hdr-proton01	P13	sw-hdr-proton01	P13			Rx: -40 Tx : -40
Pass	ib	NA	NA	Ok	NA	sw-hdr-proton01	P19					Rx: -40 Tx : -40
Pass	ib	NA	NA	Ok	NA	sw-hdr-proton01	P20					Rx: -40 Tx : -40
Pass	ib	NA	NA	Ok	NA	sw-hdr-proton01	P25					Rx: -40 Tx : -40
Pass	ib	NA	NA	Ok	NA	sw-hdr-proton01	P26					Rx: -40 Tx : -40
Pass	ib	NA	NA	Ok	NA	sw-hdr-proton01	P5					Rx: -40 Tx : -40
Pass	ib	NA	NA	Ok	NA	sw-hdr-proton01	P6					Rx: -40 Tx : -40
Pass	ib	NA	NA	Ok	NA	ufm-sw-hdr01	P1	ufm-sw-hdr01	P1			Rx: -40 Tx : -40
Pass	ib	NA	NA	Ok	NA	ufm-sw-hdr01	P7	ufm-sw-hdr01	P7			Rx: -40 Tx : -40
Pass	ib	NA	NA	Ok	NA	ufm-sw-hdr01	P9	ufm-sw-hdr01	P9			Rx: -40 Tx : -40

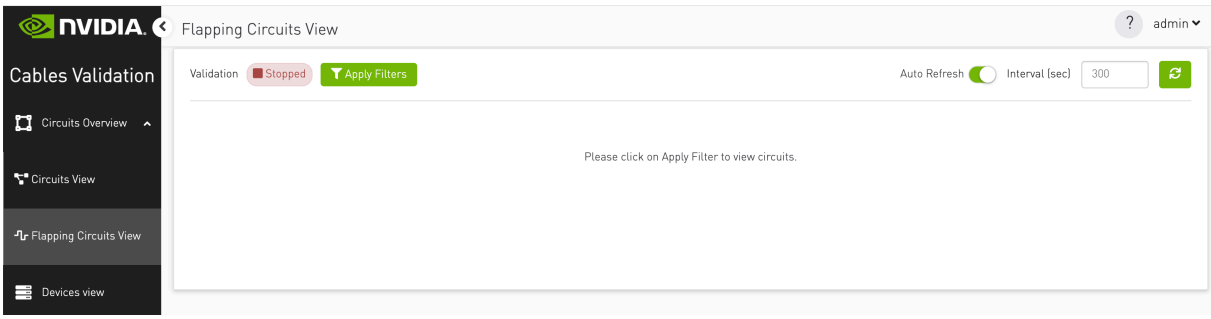
To remove the simulation, the user can simply right-click on the simulated endpoint and select '**Stop Simulation on <endpoint name>**'. This action is also only available to non-cabler users.



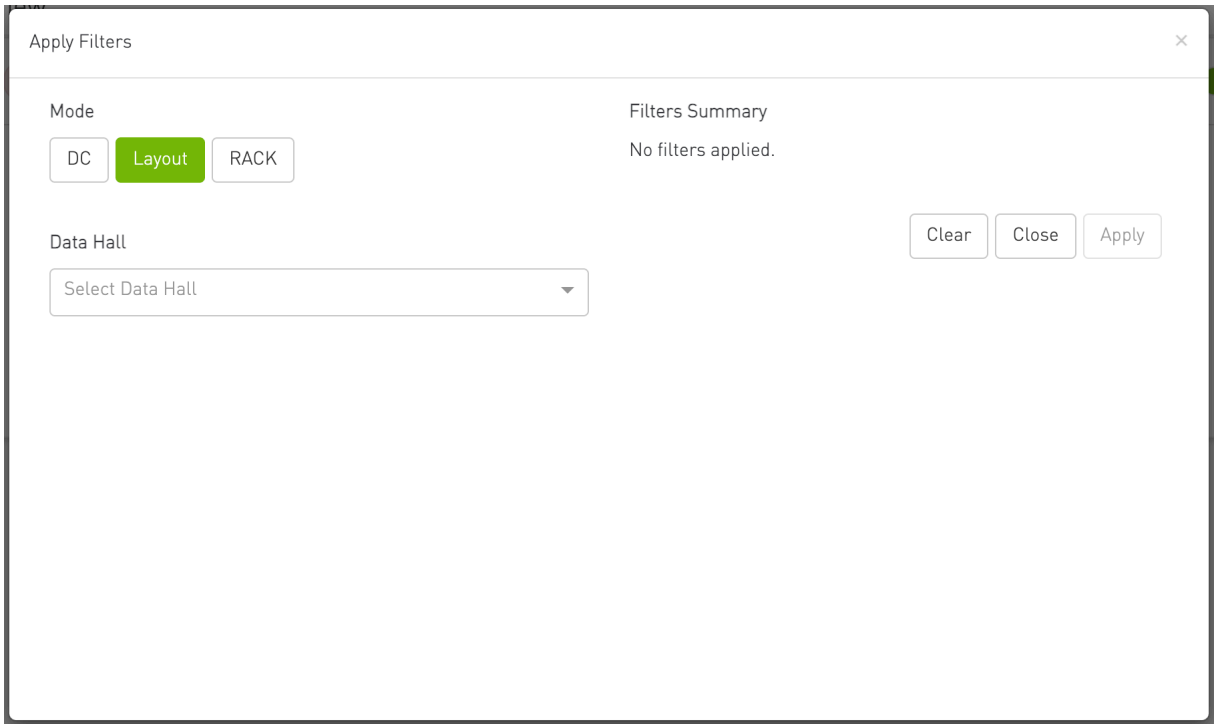
10.2.2 Flapping Circuits View

The **Flapping Circuits View** only supported for Admin users, to help explore the connection flapping details for A & Z endpoints.

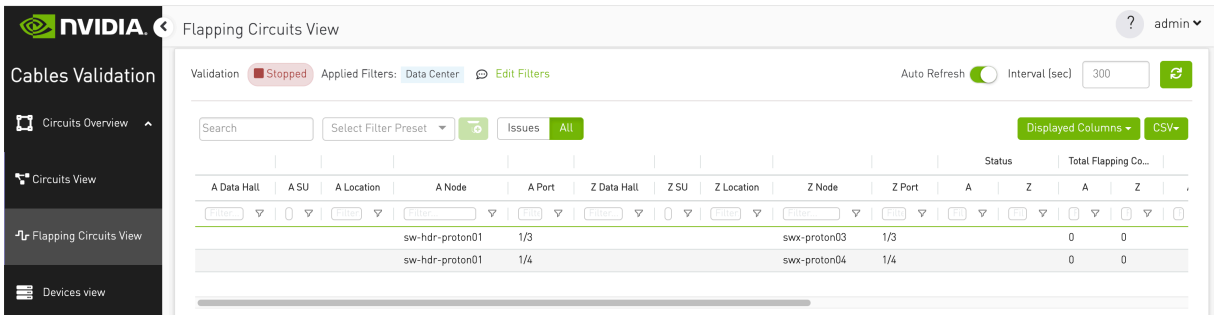
When you open the page, you'll see a message asking you to apply a filter to view the circuits.



Click "**Apply Filters**" to open the filter window, choose your preferred filter, and apply it. For more guidance, see the **Resource Filter** page.



Once your filter is applied, all matching circuits will appear in a table.



On IB and Cumulus switches, carrier transitions are monitored every 10 seconds. If the carrier transitions increments by more than 1, a link flap alarm is raised and the circuit will treated as flapping circuit.
The system provides a detailed view of current and historical flapping events for both endpoints. The historical data spans multiple time intervals, including the last 30 seconds, 1 minute, 5 minutes, 1 hour, 12 hours, and 24 hours. This information is displayed in a table.
The table includes the following columns for both endpoints:

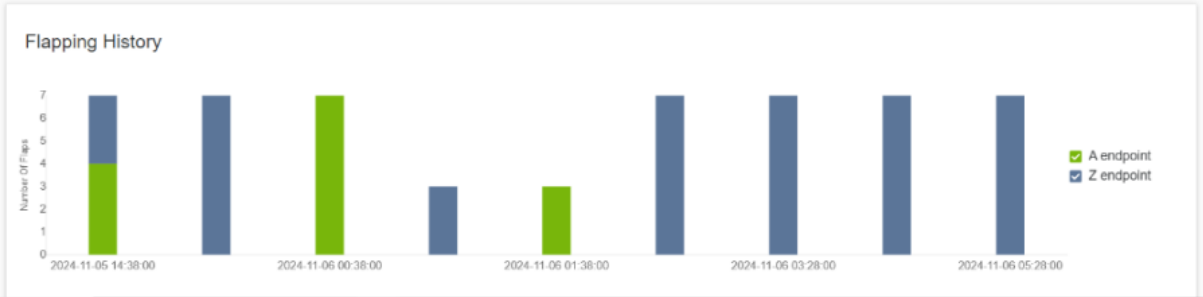
- Data Hall and SU
- Location: Specifies the endpoint's location, Rack, and Unit.
- Node and Port Details: information about the nodes and the associated ports.
- Status: flapping status and it could be one of the following values
    a. Ok - no flapping events since agent started
    b. Flapping - agent detected a flapping event in the last 1 minute

c. **Flapped** - agent detected a flapping event at some point. The Flapping event counter >= 1.

- **Total Flapping Count:** This is the total number of transitions occurring **since the bringup agent** is started on the switch.
- **Flap 30 sec:** how many flaps happened in the last 30 seconds.
- **Flap 1 min:** how many flaps happened in the last 1 minute.
- **Flap 5 min:** how many flaps happened in the last 5 minutes.
- **Flap 1 hour:** how many flaps happened in the last 1 hour.
- **Flap 12 hour:** how many flaps happened in the last 12 hours.
- **Flap 24 hour:** how many flaps happened in the last 24 hours.

10.2.2.1 Flapping History

The circuit maintains a record of the number of flaps occurred, along with the corresponding timestamps, over a 24-hour period. This record is referred to as the **flapping history**. Users can view this history by **selecting any row in the flapping circuits table**.

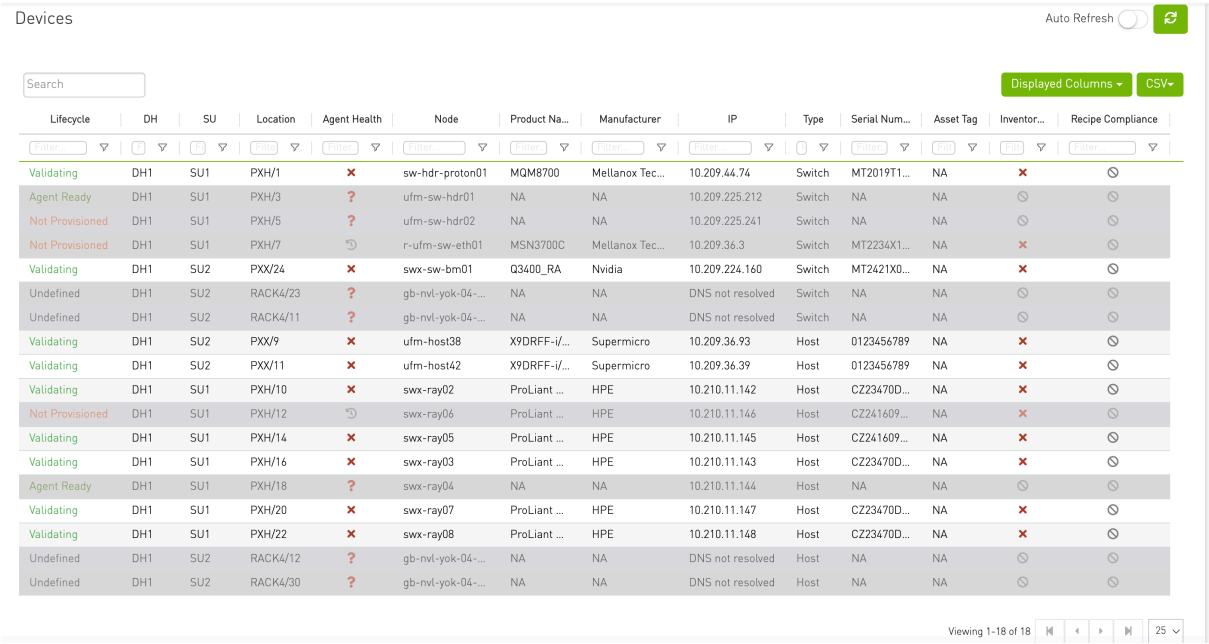


The history is displayed as a bar chart, where **green bars represent the A endpoint** and **blue bars represent the Z endpoint**.
If flaps are detected simultaneously for both the A and Z endpoints, the Z value is stacked on top of the A value in the bar chart. For example, in the first bar of the chart, the A endpoint has a value of 4, and the Z endpoint has a value of 3, making the total height of the bar 7.
Additionally, users can toggle the visibility of individual bars by interacting with the chart legend located on the right-hand side of the chart.

10.3 Devices View

The Devices View presents network devices in a structured table format, allowing users to browse, filter, and select devices for detailed analysis.

Upon device selection, the interface provides detailed information through multiple tabs, including circuits, agent status, and device health metrics.
The system automatically refreshes device information and provides real-time status updates.
Users can export device data, filter results, and navigate through different device details.



The main interface displays devices in a tabular format with the following columns:

Column	Description
Lifecycle	The device's current lifecycle state. See Device Status for supported states and their meanings.
DH	The data hall identifier where the device is located.
SU	The scalable unit within the data hall.
Location	The rack name and rack unit, displayed as Rack/Unit when both values are available.
Agent Health	Agent availability and health based on whether the device is managed, whether an agent is installed, report freshness, and reported failures. See Agent Status.
Node	The device's display node name; the canonical node name is used when a display name is unavailable.
Product Name	The product name reported in the device's system information.
Manufacturer	The manufacturer reported in the device's system information.
IP	The device IP address.
Type	The device type.
Serial Number	The serial number reported in the device's system information.
Asset Tag	The device asset tag.
Inventory Health	The summarized hardware inventory state: Active, Failed, No Data, or Unmanaged.
Recipe Compliance	The device recipe-validation result: Pass, Fail, Not Validated, No Recipe Mapped, No Recipe File, No Data, or Invalid Recipe File.

10.3.1 Agent Status

Agent status could be one of the following:

- Unmanaged (🚫 Gray background): Agent installation is not supported on this device.
- Agent Not Installed (❓ Gray background): No agent is installed.
- Stale Agent (🕒 Gray background): No update has been received from the agent in the last 15 minutes.
- Agent Reporting Failures (❌): The agent has reported either hardware issues (e.g., fans, power supplies) or port issues.
- No Failures on Agent (✅): No hardware or port issues detected by the agent.

10.3.2 Device Selection and Navigation

When a device is selected from the table, a details panel appears on the right side of the screen.

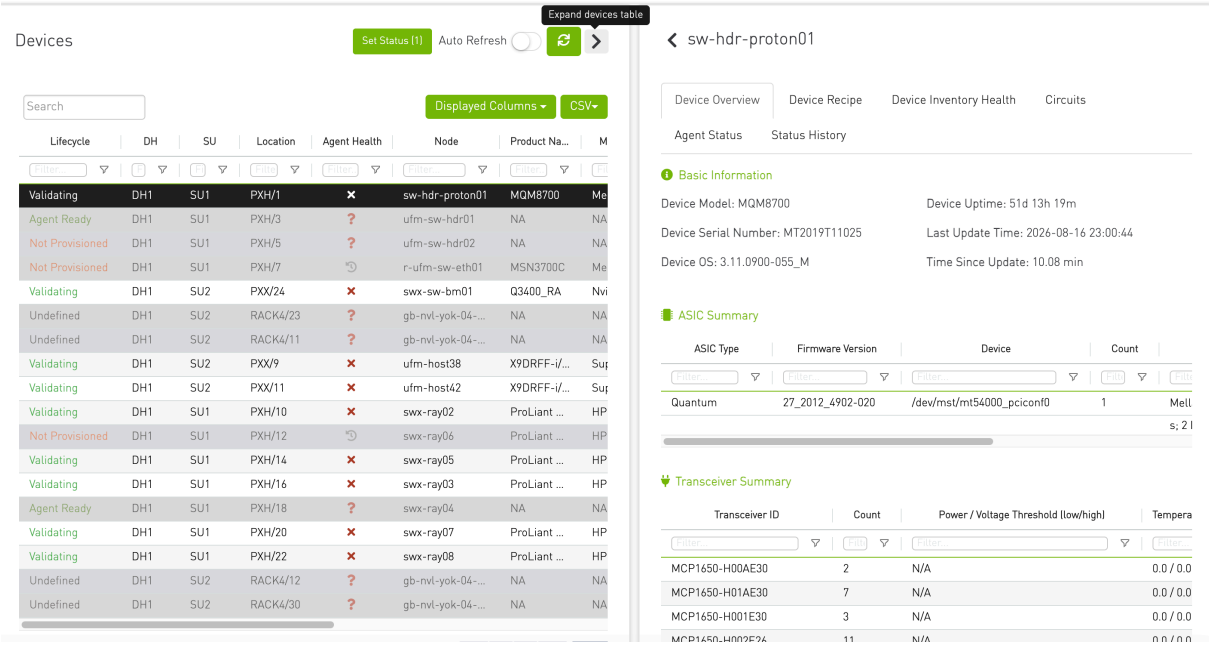
The selected device's information is highlighted in the device header, and the available tabs are dynamically determined based on the device's capabilities.

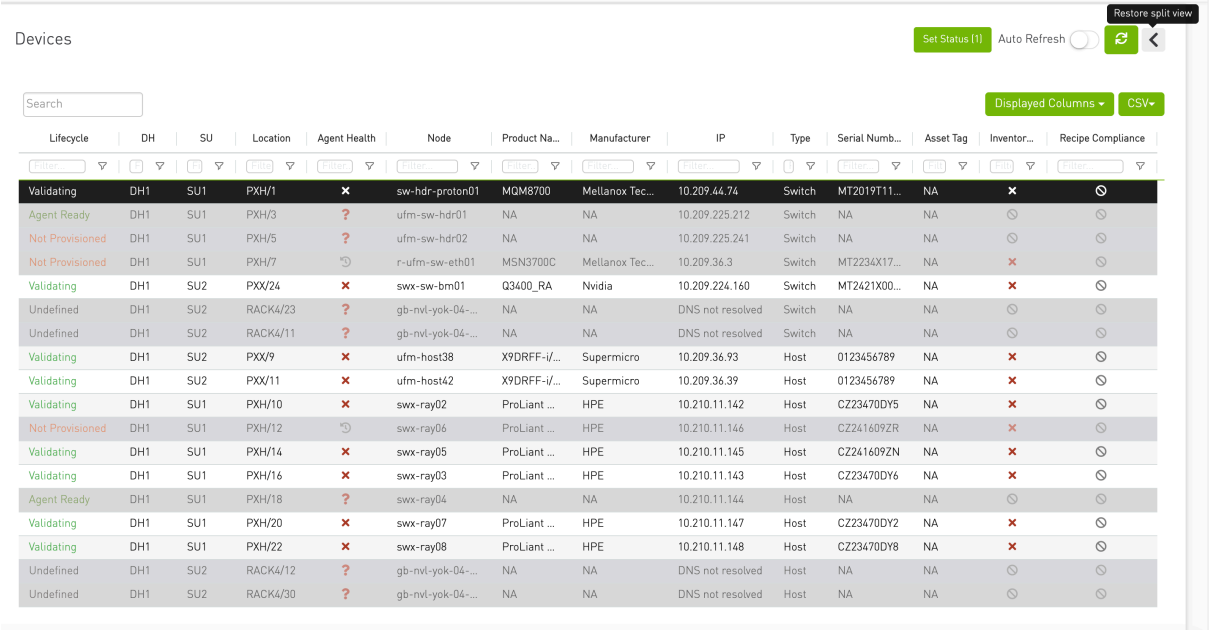
10.3.2.1 Expanding the Devices Table

When exactly one device is selected, a chevron control appears on the right side of the Devices toolbar. Use it to give the table more horizontal space while keeping the selected row active.

1. Select one device row. The table and the selected device's details are displayed side by side.
2. Select the right chevron (Expand devices table). The details panel is hidden and the table expands to the right into the available space.
3. Select the left chevron (Restore split view) to display the selected device's details panel again.

The table-expansion control is available only when one device is selected. Clearing the selection or selecting multiple devices exits the expanded-table view.



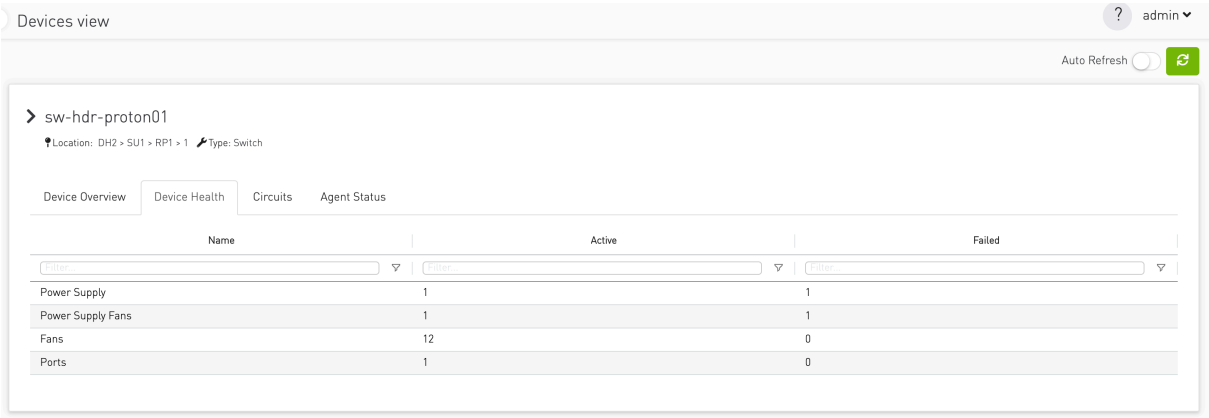


10.3.2.2 Expanding the Device Details

The device header displays the device name with expand/collapse functionality on the left side of the name.

In expanded mode, the device details panel stretches to fit the page and displays the device's location hierarchy — Data Hall > Scalable Unit > Rack > Unit (when available).

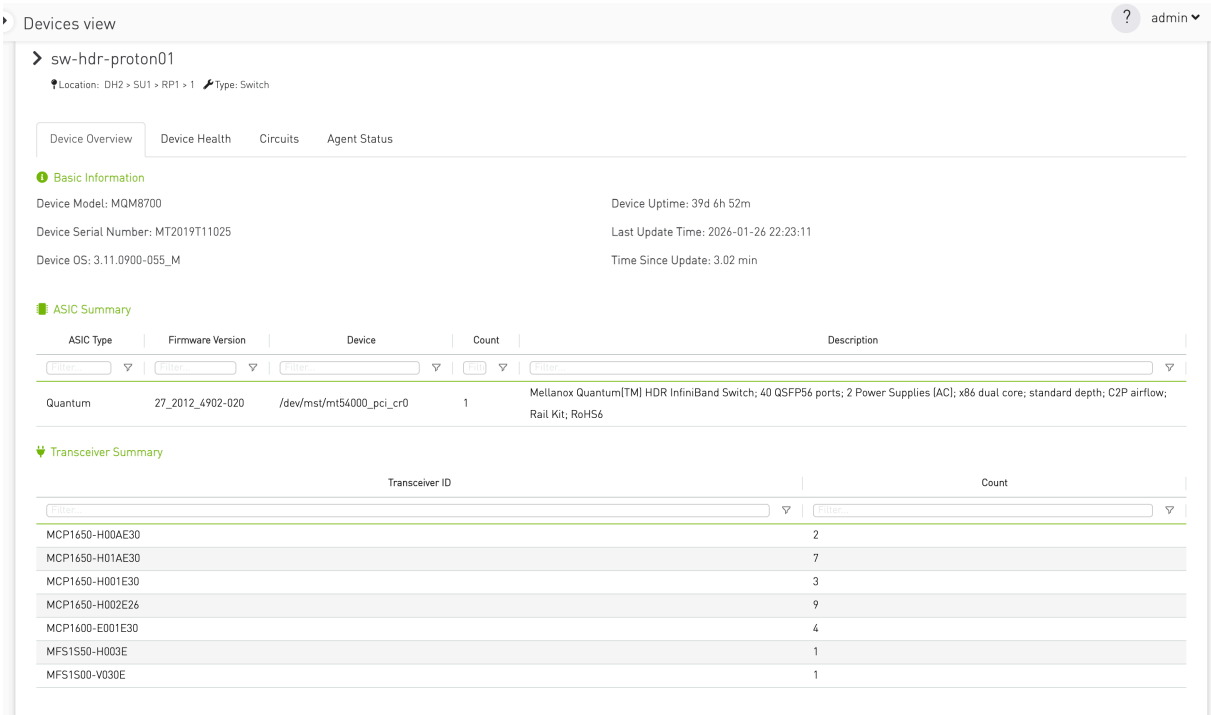
It also includes information about the device type.



10.3.3 Device Tabs

10.3.3.1 Device Overview Tab

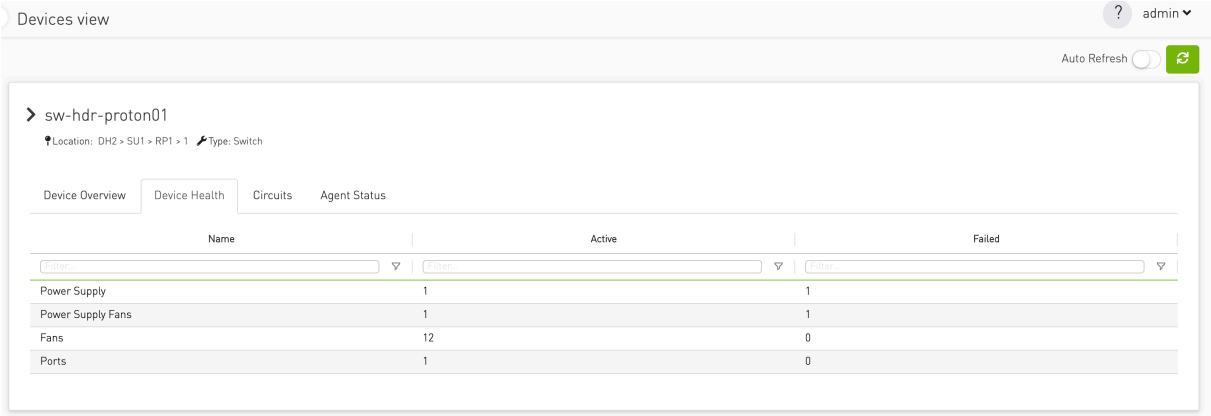
The **Device Overview** tab provides an overview of the physical aspects of the device like Device Model, OS, Serial Number, Uptime, ASIC Summary and the Transceiver models.



10.3.3.2 Device Health Tab

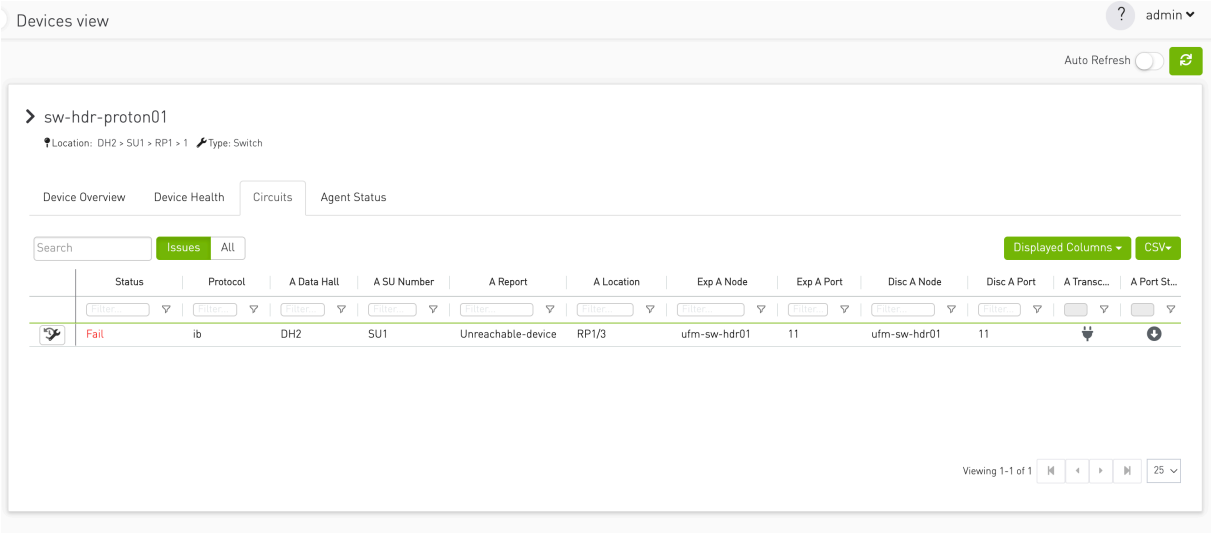
The **Device Health** tab provides an overview of the selected device's health metrics in a tabular format.

> **Note:** This tab is only visible for manageable devices.



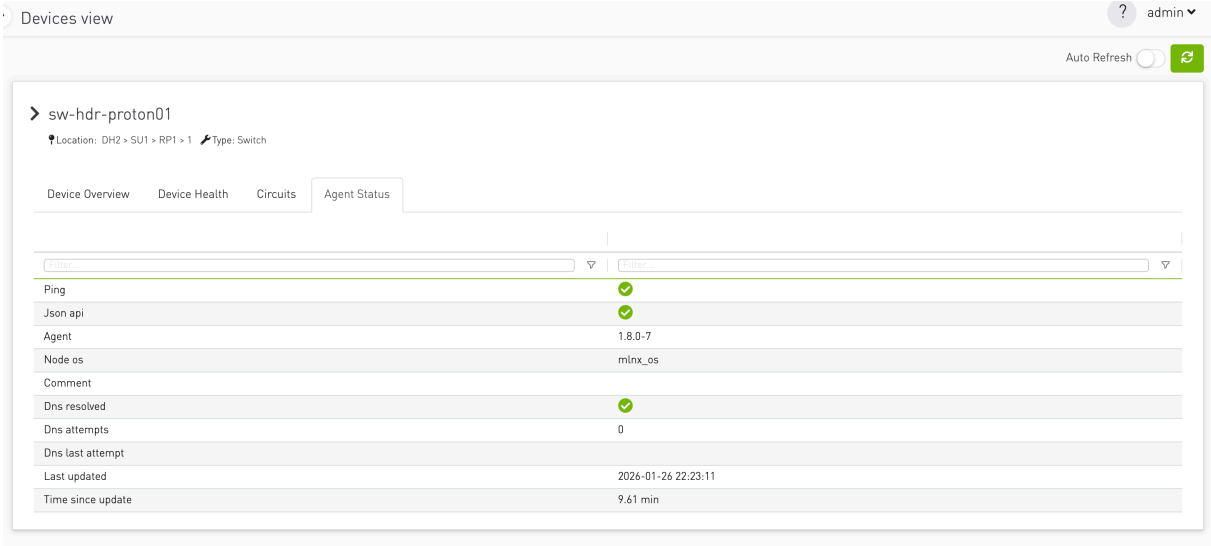
Circuits Tab

The **Circuits** tab displays all circuits related to the selected device.



10.3.3.3 Agent Status Tab

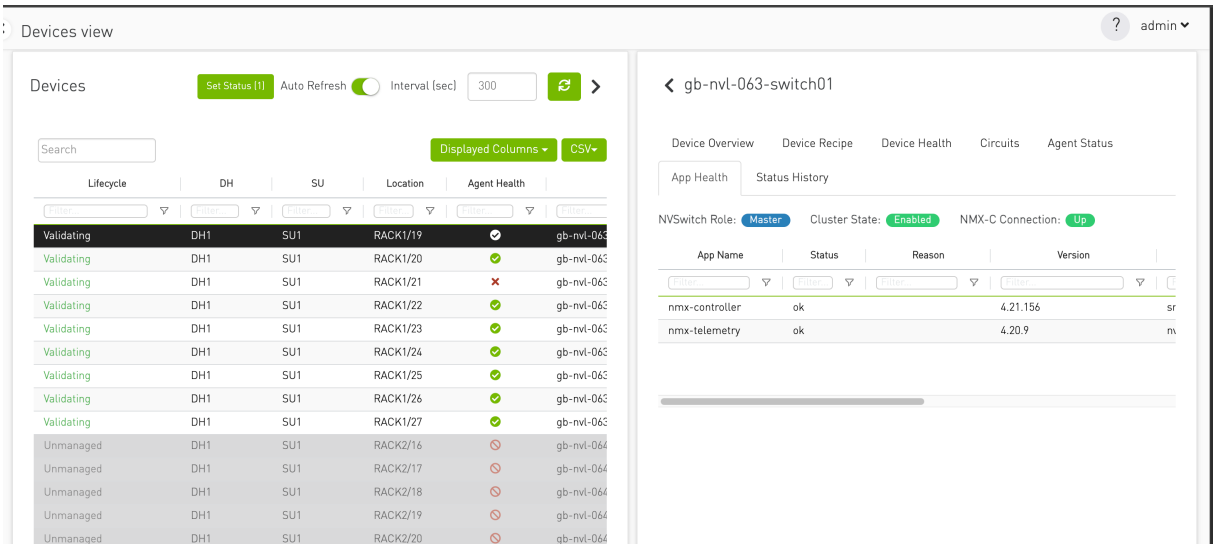
The **Agent Status** tab presents detailed information about the software agent running on managed devices. It includes the agent's connectivity status, version information, and the timestamp of the last communication.
Note: This tab is only visible for manageable devices and is accessible only to users with administrative privileges.



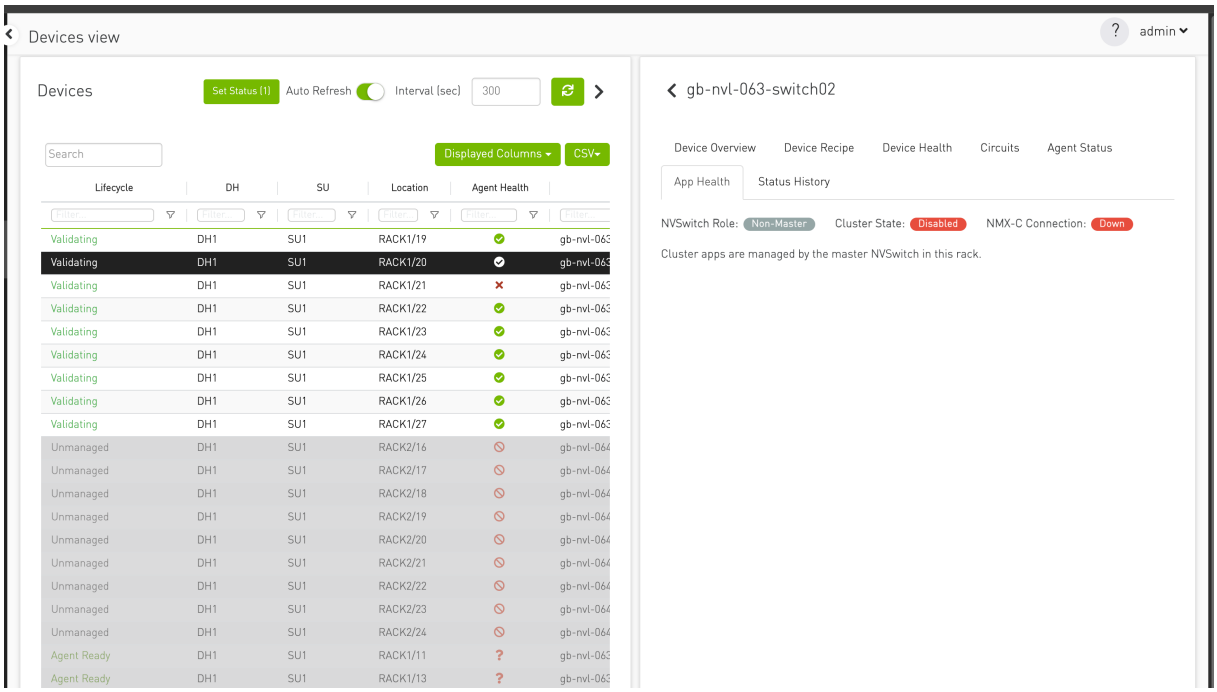
10.3.3.4 App Health Tab

The **App Health** tab displays the NVSwitch role, cluster state, NMX-C connection status, and cluster application information. This tab is available for managed devices running NVOS. Cluster application details are shown on the master NVSwitch; non-master devices indicate that the master manages them.

master role view:



non master role view:



Note: This tab is only visible for NVSwitch ,manageable devices and is accessible only to users with administrative privileges.

10.3.3.5 Device Recipe

Recipe validation lets CVT operators confirm that devices in a fabric are running the **expected software and firmware versions** defined in YAML recipe files. CVT compares **actual versions reported by agents** against **expected versions in the recipe**, and surfaces pass/fail status at the device and rack level.

10.3.3.6 How it works

1. **Recipe files** — YAML files listing expected software and firmware versions (e.g. switch image, ASIC firmware).
2. **Mapping file** — A YAML mapping assigns recipes to devices using rules (by data hall, scalable unit, rack, hostname pattern, role, etc.) or explicit per-device overrides. Resolution order is: override → rack → SU → DH → pattern → role → default.
3. **Validation** — When agent data is available, the collector compares each version field in the recipe to what the device reports and marks each field Pass or Fail. A device-level status is derived from those results.

10.3.3.7 Where it appears in the UI

Location	Purpose
System Admin → Agent Management → Recipe Management	Upload/manage recipe and mapping YAML files
Devices View → Device Recipe tab (described in this section)	See assigned recipe, mapping rule, and field-level results; assign or override per device

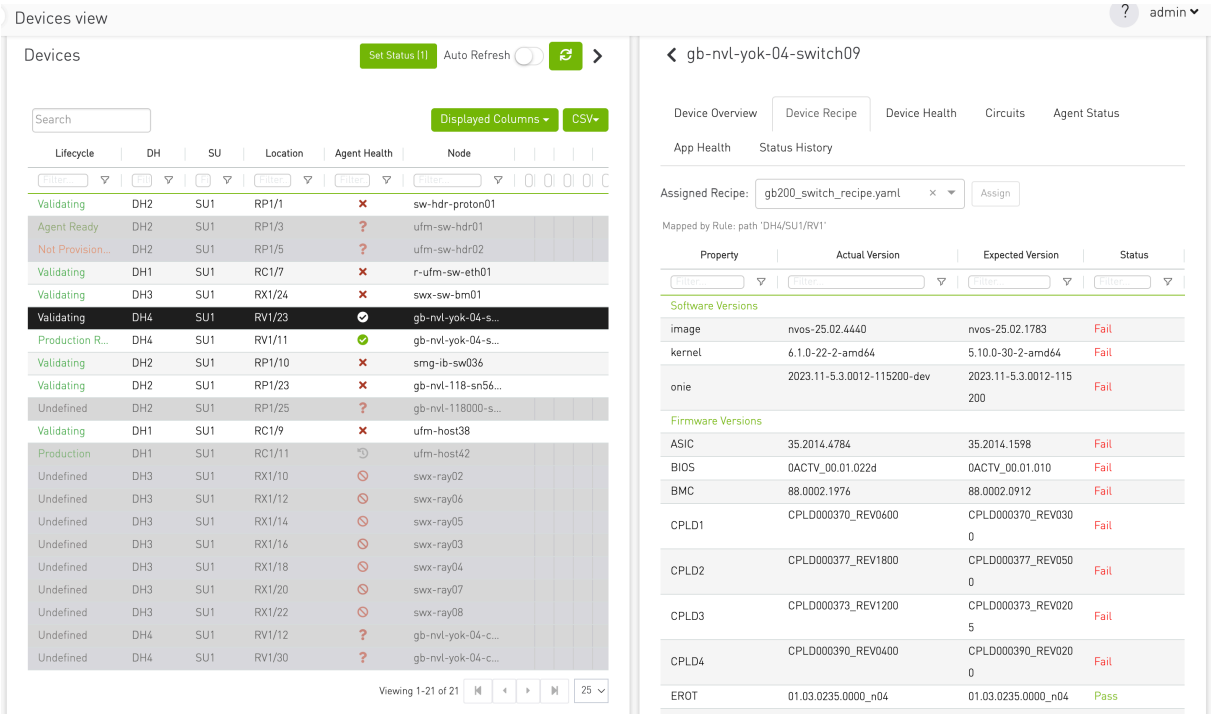
Location	Purpose
Devices View grid (described in this section)	**Recipe Compliance** column for fleet-wide scanning
Rack View	Per-rack compliance summary and per-unit recipe badges

10.3.3.8 Device statuses

- Pass / Fail — All version fields match / at least one mismatch
- No Recipe Mapped — No recipe assigned via rules or override
- No Data — Recipe assigned but agent has not reported version info yet
- Not Validated — Validation not yet run

10.3.3.9 Important notes

- Recipe validation is **additive** — it runs alongside cable validation and does **not** gate validation cycles.
- Mapping is re-resolved when topology is loaded or the mapping file changes; only affected devices are revalidated.
- Per-device overrides from the UI are persisted in the mapping file and take highest priority over rules.



10.3.4 Recipe Assignment Row

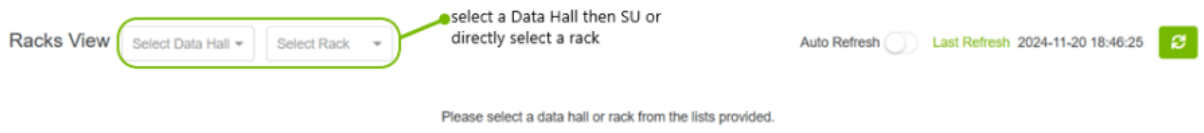
Element	Description
Label	"Assigned Recipe:"
Dropdown	Lists recipe files
Assign button	Enabled when selection differs from current assignment
Spinner	Shown while assignment request is in flight

Assignment behavior:

- Selecting a recipe and clicking Assign writes an entry to mapping `overrides` and persists the mapping file.
- Clearing selection (empty recipe) removes the device from `overrides` and re-resolves from rules/default if applicable.
- Backend immediately revalidates the device if `system_info` is available.

10.4 Rack View

The Rack View provides a simulated visual representation of a selected rack, including its devices, ports, and statuses. This view is populated using data from a PTP Excel file.

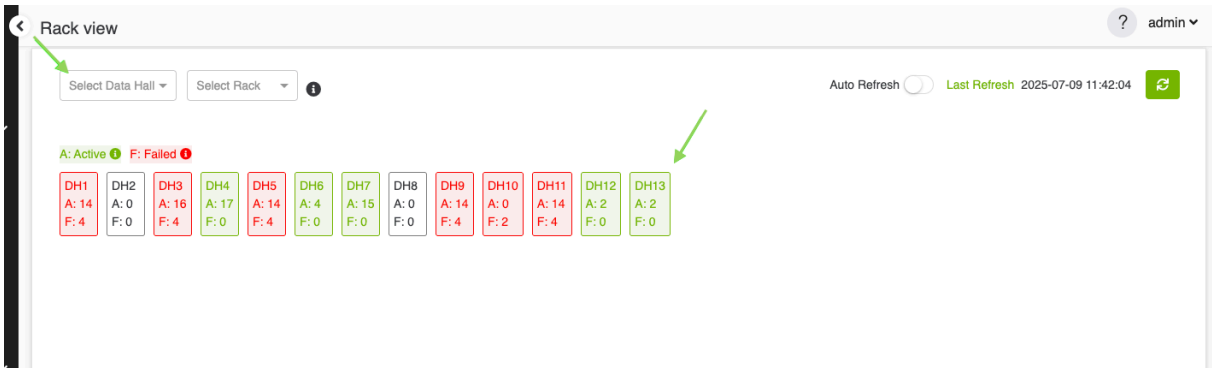


The page is initially blank, and user has 2 options:

10.4.1 Hierarchical Selection Workflow

This hierarchical selection process allows users to navigate from a Data Hall to SU and then to specific Racks. This feature is only available if the user has provided the **DC Floor Layout file** while loading the P2P file. Below are the steps the user should follow:

1. Select a Data Hall: Begin by selecting a Data Hall from the dropdown menu or by clicking on one of the DH squares displayed below, which show the number of active and failed components within each Data Hall. This selection determines the scope of available Scalable Units and Racks.



2. **Display and Select SU:** Once a Data Hall is selected, all associated **Scalable Units (SUs)** will be displayed in two forms:

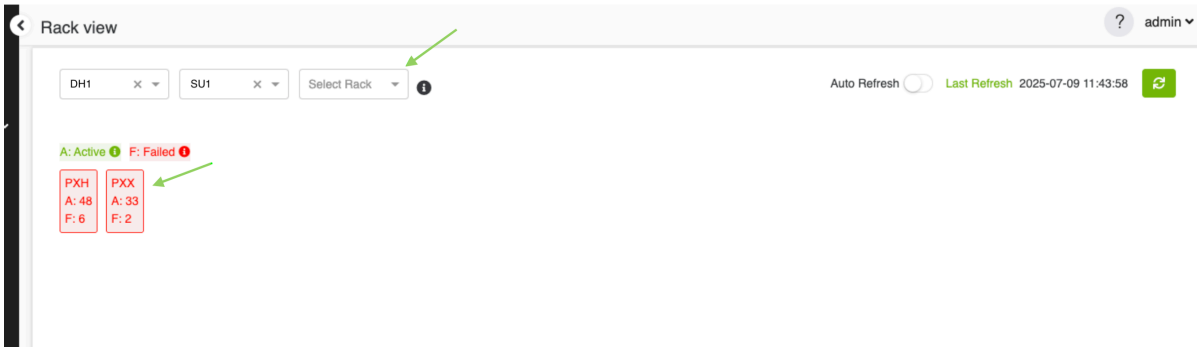
- - **Dropdown Menu** – A list of available SUs for easy selection.
  - **Visual Squares** – Each square represents an SU and displays the number of active and failed components.

The user can either select an SU from the dropdown or click on one of the SU squares in the visual layout.

3. **Display and Select Racks:** the same as SU, after selecting an SU, all associated **Racks** within that SU will be displayed in two forms:

- - **Dropdown Menu** – A list of all racks within the selected SU.
  - **Visual Squares** – Each square represents a Rack and includes the number of active and failed components.

The user can either select a rack from the dropdown menu or click on a rack name to view the simulated visual representation of the selected rack.

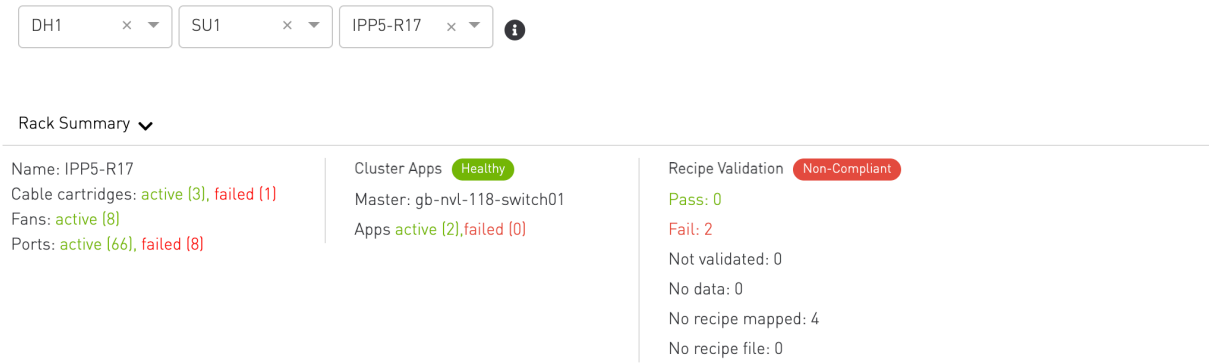


10.4.2 Select a Rack Directly

If the DC Floor layout is not loaded, user can directly select a rack from the rack menu to see the simulated visual representation for it.
The rack menu will contain all racks recognized by the CVT tool.

10.4.3 Rack-Level Summary

At the top of the view, an aggregated summary for the entire rack is displayed.

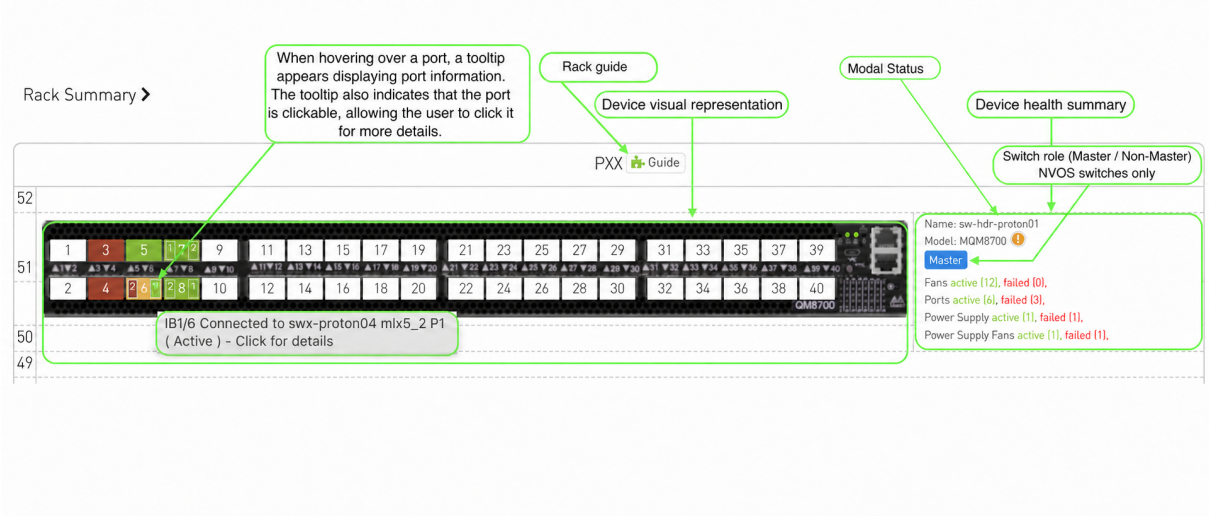


10.4.4 Front View

Once a rack is selected, the corresponding front view of the rack will be displayed.

The front view is the default Rack View and displays the forward-facing side of the rack.

It shows all devices (switches, servers, and compute trays) mounted in the rack along with their physical port layout, port status coloring, status LEDs, and device summaries.

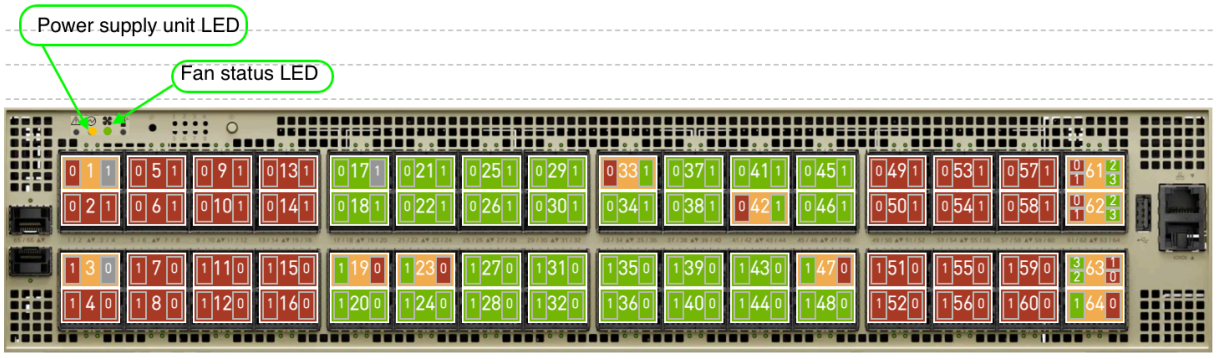


The tool support visualization for the following switches and hosts:

10.4.4.1 Ethernet

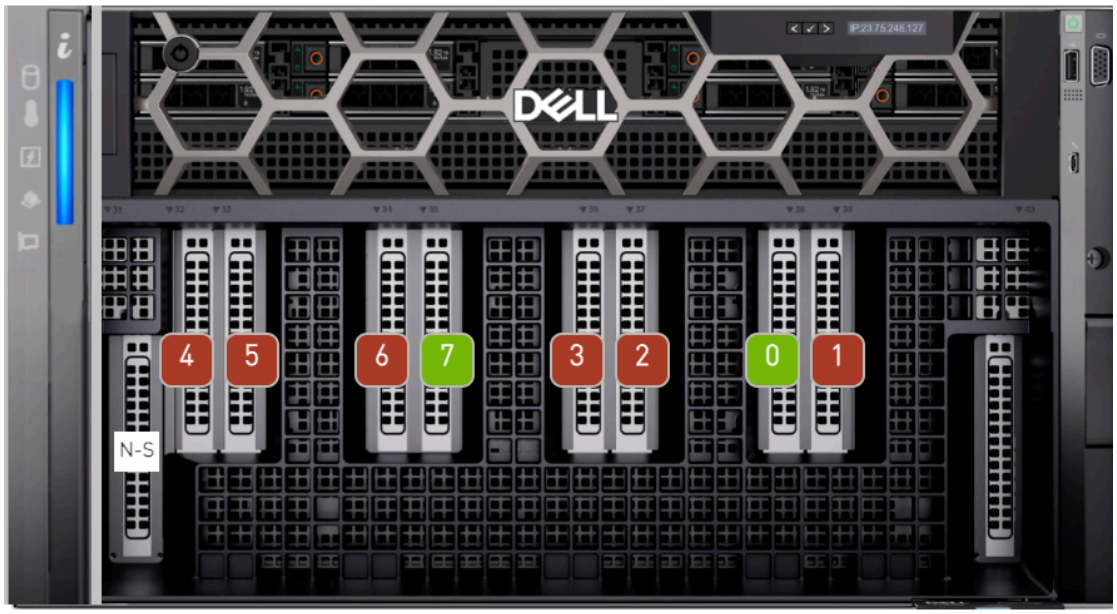
10.4.4.1.1 SN5600 SN5600D Switch

10.4.4.1.2 SN5610 Switch



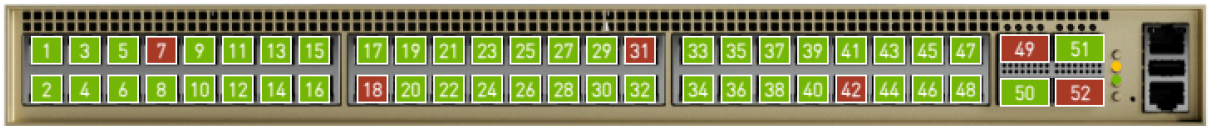
10.4.4.1.3 SMC SuperServer

10.4.4.1.4 XE9680 Server

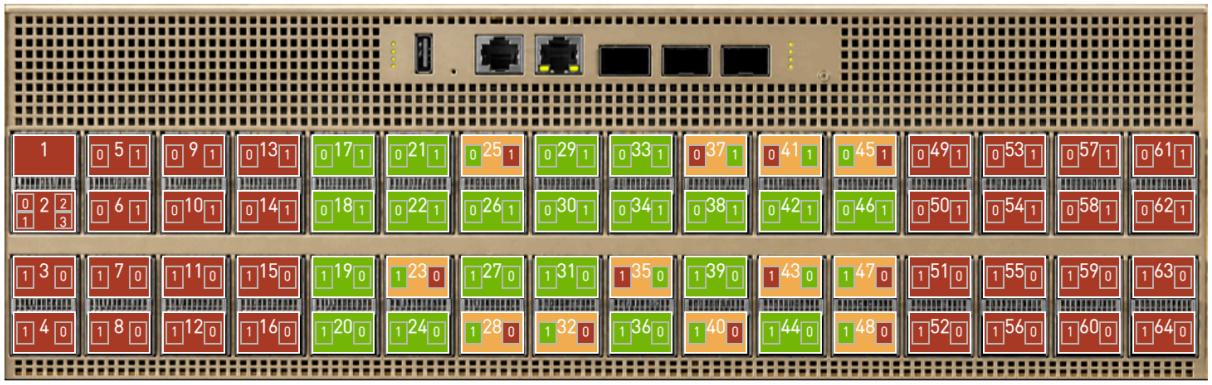


10.4.4.1.5 SN2201 Switch

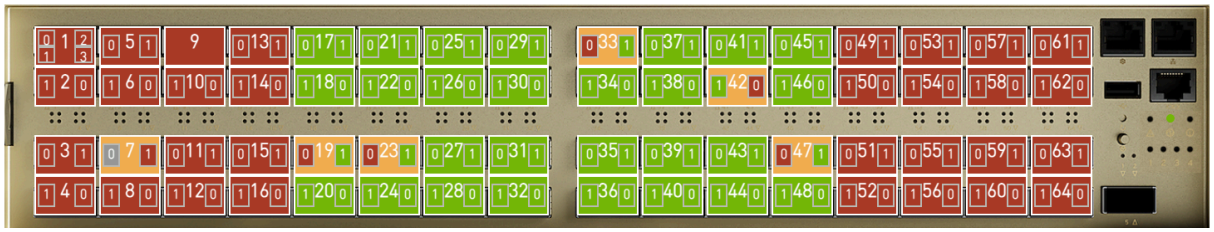
10.4.4.1.6 SN2201_M Switch



10.4.4.1.7 SN6600 Switch



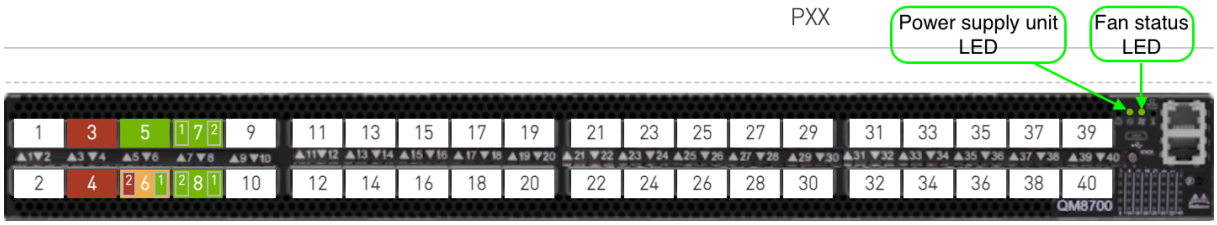
10.4.4.1.8 SN6600-LD Switch



10.4.4.2 InfiniBand

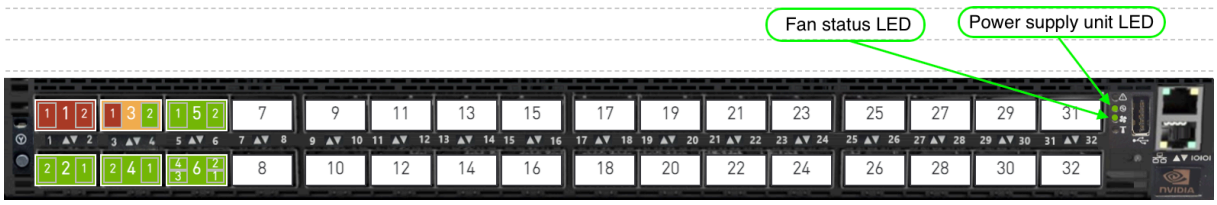
10.4.4.2.1 MQM8700 Switch

The MQM8790 model uses the same Rack View visualization as MQM8700.

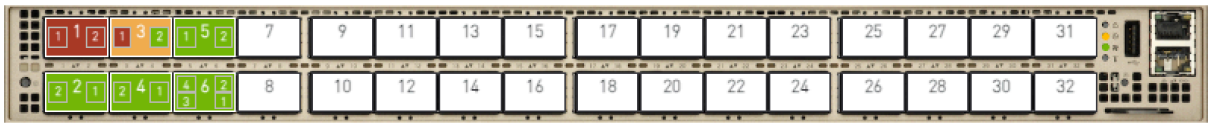


10.4.4.2.2 MQM9700 Switch

The MQM9790 and QM9700 model names use the same Rack View visualization as MQM9700.



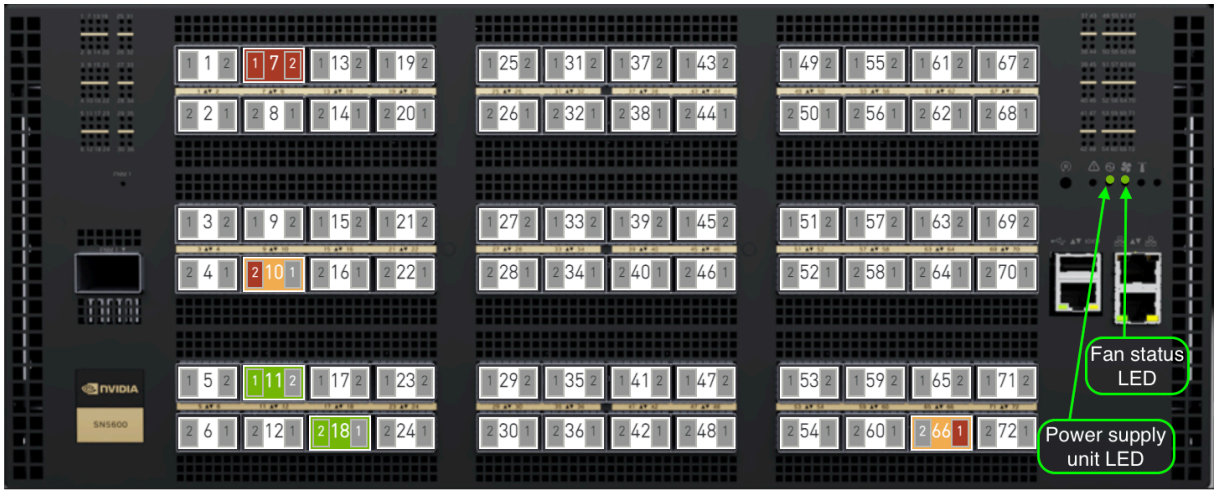
10.4.4.2.3 MQM9701 / QM9701 Switch



10.4.4.3 XDR

10.4.4.3.1 Quantum-3 XDR Switch

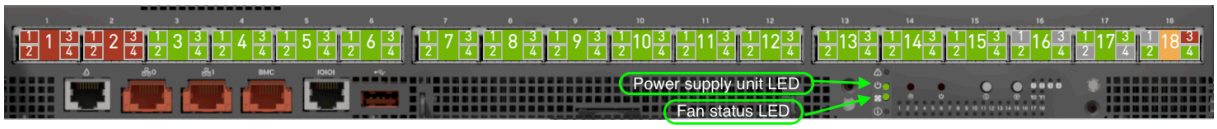
Supported model names for this visualization are Q3400, Q3400_RA, and Q3400-RA.



10.4.4.4 NVLink

10.4.4.4.1 Scaleout Switch

Supported scaleout switch models are N5110_LD, N5112_LD, and N5113_LD.



10.4.4.4.2 Non-Scaleout Switch

Supported non-scaleout switch models are N5200_LD, N5201_LD, and N5700_LD.

10.4.4.4.3 Non-Scaleout DGX Switch

The supported non-scaleout DGX switch model is N5400_LD.

10.4.4.4.4 N6100_LD Switch

The Rack View provides a dedicated 1U front-panel visualization for the Vera Rubin N6100_LD NVLink switch.

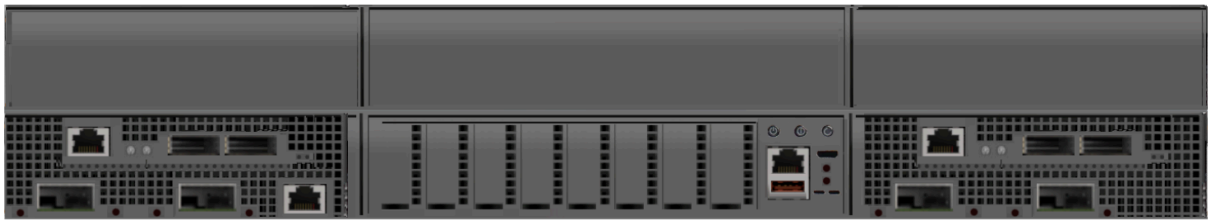
10.4.4.4.5 HGX Compute Tray

The GB200 NVL model uses the HGX Compute Tray visualization.

10.4.4.4.6 DGX Compute Tray

The DGX GB200 Compute Tray uses this visualization. DGX GB300 Compute Tray is supported as an alias of the same layout.

10.4.4.4.7 2U GB200 Compute Tray



10.4.4.4.8 Dell Server 9712a

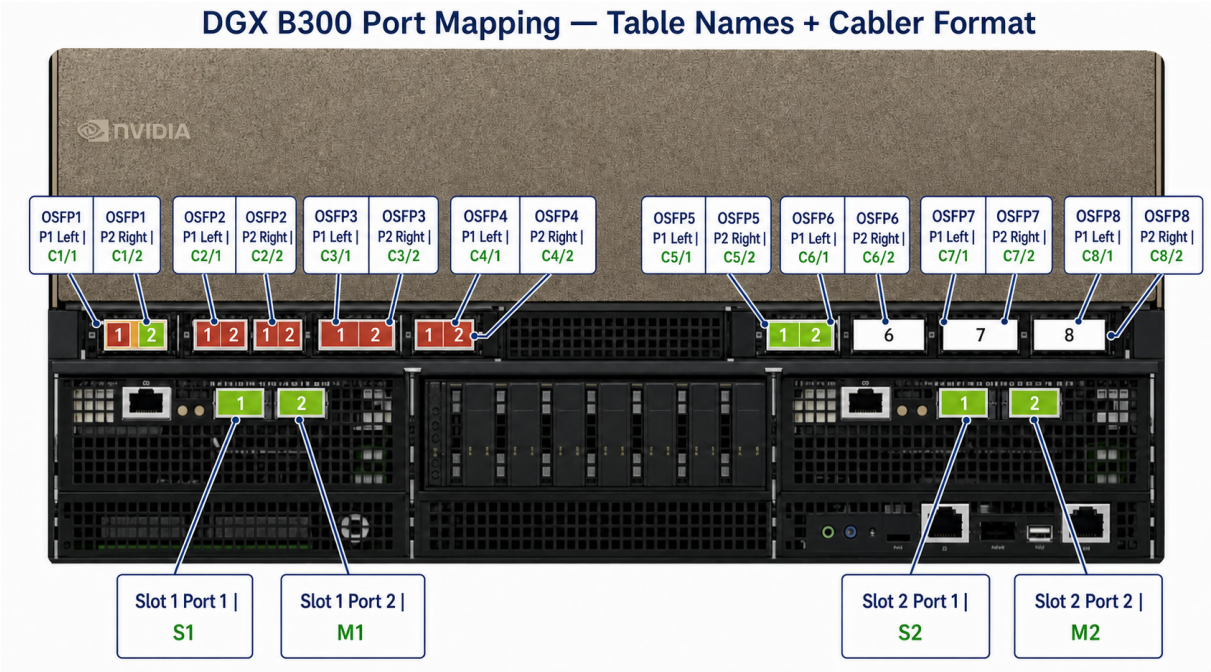
10.4.4.5 DGX-B300 Server

The DGX-B300 server visualization displays two port groups:

- OSFP Ports — Up to 8 OSFP connector ports (ConnectX-8) arranged in a single row, with a spacing gap after every 4th port to match the physical layout of the device.
- BlueField NIC Slots (Slot 1, Slot 2) — 2 BlueField NIC slot ports.

The DGX-B300 device occupies 10 rack units in the rack view.

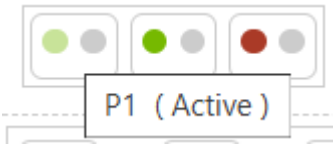
Prerequisite: To view the DGX-B300 correctly in the Rack View, users must fill in the physicalPort column in the Server Profile sheet of the P2P file. Port names should follow the physical port naming convention as the following:



10.4.4.6 Default

The default view is used to display devices that don't have a product name.

Default Host



Default Switch

10.4.4.7 Port Status Coloring

Each port on the device is color-coded to visually represent its status, making it easier to identify and troubleshoot issues. The status of each port is indicated by the following colors:
Ports are color-coded based on their related issues:

-

White: Null (no connection or data).
-

Green: Active and functioning properly.
-

Light Red: Indicates issues such as:
  - Unknown Neighbor.
  - Underperforming Link (BER).
  - Extra Cable.
-

Dark Red: Issues not covered by the above categories.

Split Port Status
Split ports occur when a single physical port is divided into multiple logical ports. For example, in the BlackMamba switch, a single physical port may be divided into two logical ports, named sw<num>p1 and sw<num>p2.
The color-coding system for split ports follows these guidelines:

1. Uniform Status Across Logical Ports: If all logical ports of a split port share the same status, the coloring will match that of a non-split port.
2. Mixed Issues with Different Severities: If the logical ports have issues of varying severity, the background color will reflect highest severity.
3. Mixed Active and Issue States: If one or more logical ports have issues while others are active, the background color will be orange

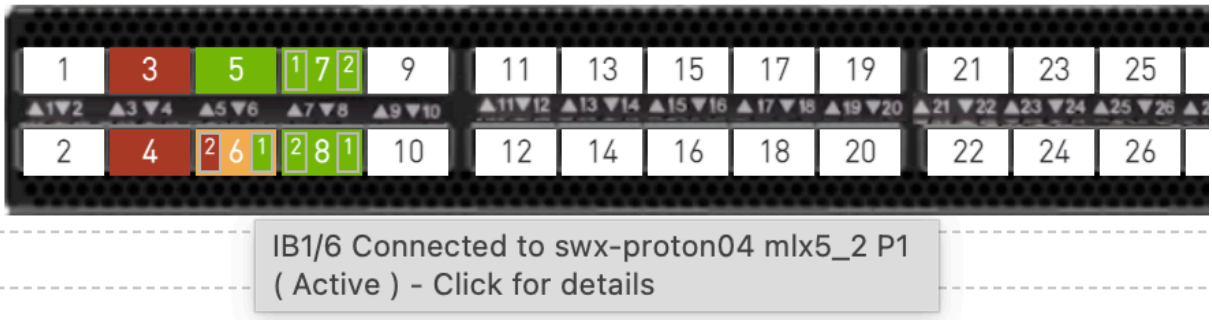
, indicating a mixed state.

Port Details

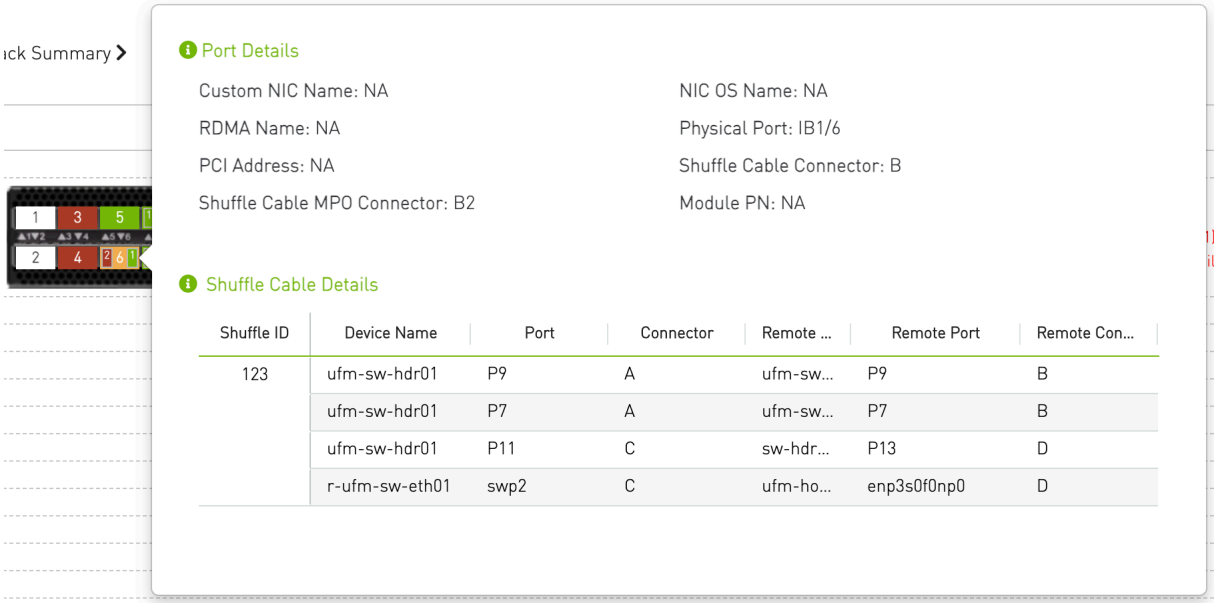
When hovering over a port, a tooltip appears displaying:

- Port-specific information
- Details about the associated issue (if any)

- An indication that the port is clickable



Clicking on a port opens a popover that displays detailed information about the selected port, including shuffle cable details (if available).



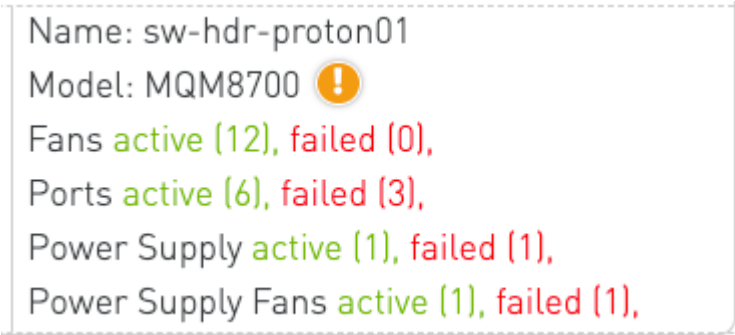
10.4.4.8 Status LEDs

Symbol	Name	Description	Color
	Power Supply Unit LED	Shows the health of the power supply units.	Amber if fail, Green if everything ok
	Fan Status LED	Shows the health of the fans	Amber if fail, Green if everything ok

10.4.4.9 Device Summary

Beside each device in the rack, a summary panel provides an overview of its status, including:

1. Name
2. Model
3. Fans: Count of active and failed fans.
4. Ports: Count of active and failed ports.
5. Power Supplies: Count of operational and failed power supply units.
6. Power Supply Fans: Count of operational and failed power supply fans.

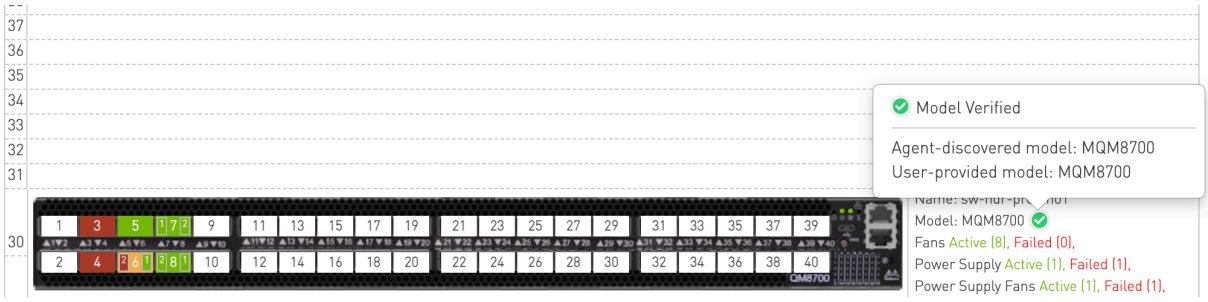


10.4.4.9.1 Device Model

The Device Summary displays device images based on the available device model information. The image selection logic uses data from both the system discovery process and the node_model field in the P2P file. Based on the availability and consistency of these values, the Rack View assigns a Model Status to each device.

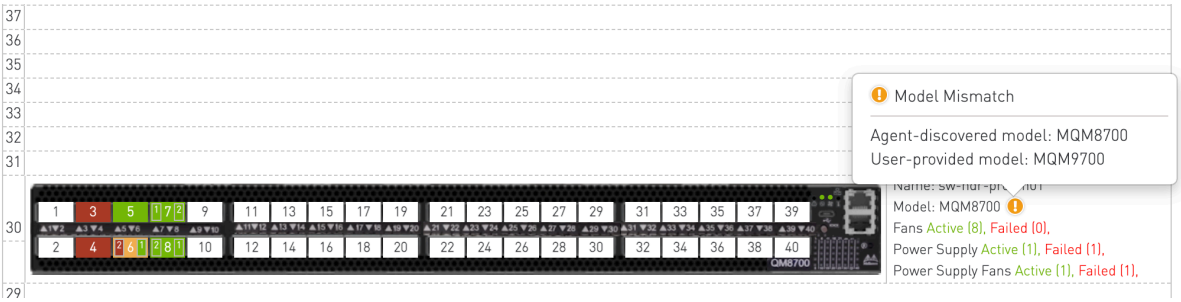
10.4.4.9.1.1 Model Status Scenarios

- Model Status: Verified
  Displayed when both the agent-discovered model and the P2P `node_model` value exist and match.
  This indicates that the device model information has been successfully validated.

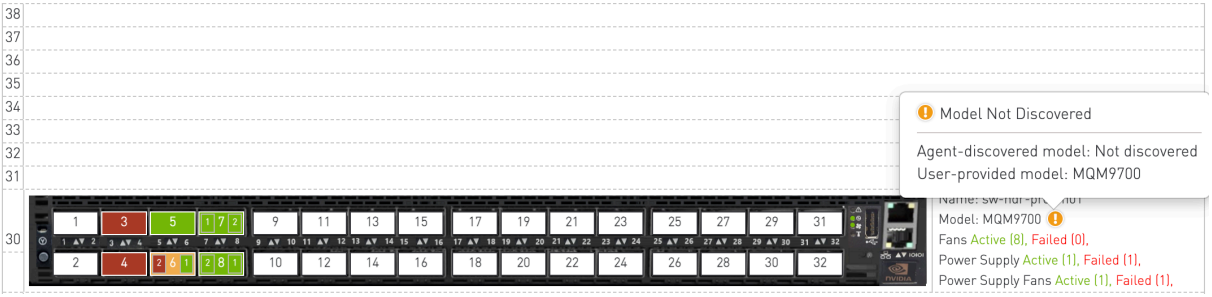


- Model Status: Inferred
  Displayed when only the agent-discovered model is available.
  The device model is assumed based on system discovery data.

- Model Status: Mismatch
  Displayed when both the agent-discovered model and the P2P **node_model** value exist but do not match.
  This indicates a discrepancy that may require user review.

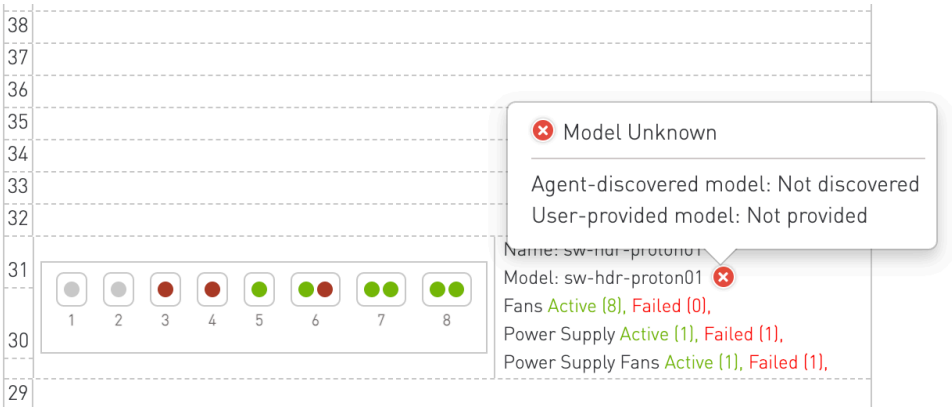


- Model Status: Not Discovered
  Displayed when only the P2P **node_model** value is available and no agent-discovered data exists.
  The device model is taken from the P2P file without system verification.



- Model Status: Unknown
  Displayed when neither the agent-discovered model nor the P2P **node_model** value is

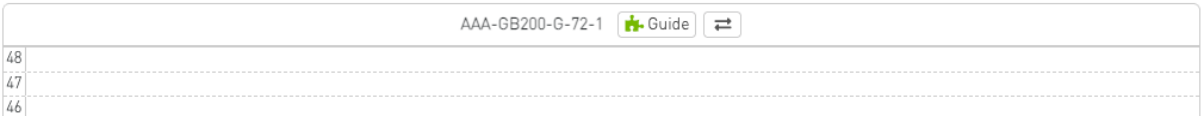
available.
The device model cannot be determined.



10.4.5 Front / Rear View Toggle

For supported rack types (currently GB200 NVLink racks), the Rack View provides both a **front view** and a **rear view** of the rack. A toggle button appears in the rack header when rear-view data is available.

- Clicking the toggle button switches between the **front view** (device ports and cables) and the **rear view** (Cable Cartridge visualization).
- The active view is persisted while navigating between racks of the same type.
- For racks without rear-view data, the toggle button is hidden and the front view is shown by default.



10.4.6 Rear View — Cable Cartridge Visualization

The rear view displays the **Cable Cartridge (CC)** layout for racks. Cable Cartridges are physical components accessible from the rear of the rack that connect devices across rack units.

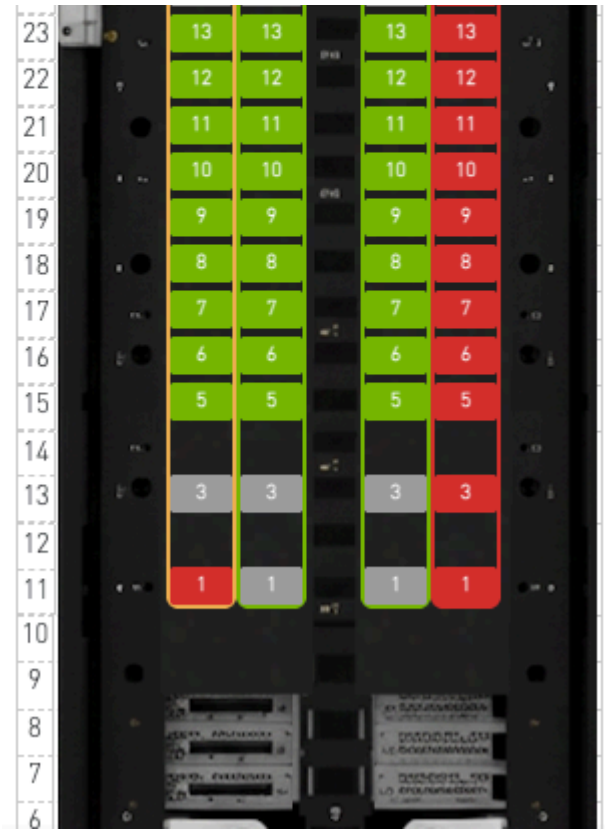
10.4.6.1 Availability

The rear view is available for the following rack types:

- GB200 36x1
- GB200 36x2
- GB200 72x1

10.4.6.2 Cable Cartridge Layout

The rear view renders a background image of the rack's rear panel with Cable Cartridge columns overlaid. Each CC column (CC#1 through CC#4) shows the slot blocks corresponding to the devices (switches and computes) connected through that cartridge. Slot blocks are vertically aligned with the rack unit grid.



10.4.6.3 Cable Cartridge Status Coloring

Each cable cartridge and its individual slots are color-coded based on validation status:

Color	Status	Description
Green	ok	All validations passed for this CC/slot.
Red	failed	Validation has detected an issue (e.g., EEPROM read failure, bad CC).
Orange	partial_failure	Some slots on this CC have failed while others are OK.
Grey	not_discovered	No agent has reported data for this CC/slot yet.

The status color is applied to the CC header, column border, and individual slot blocks.

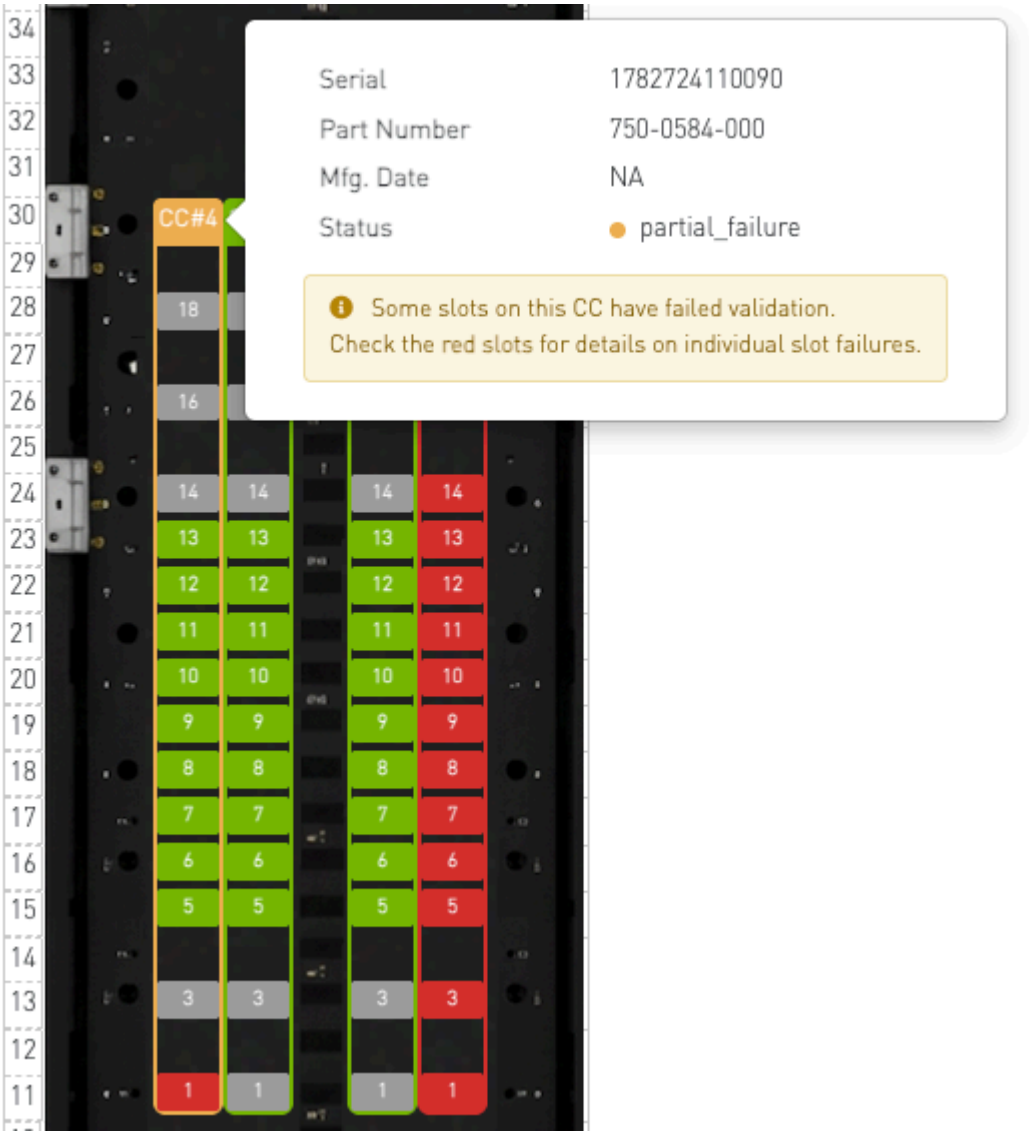
10.4.6.4 Cable Cartridge Details (CC-Level Popover)

Clicking on a **CC header** (e.g., "CC#1") opens a popover with cartridge-level details:

- **Serial** — Cartridge serial number.
- **Part Number** — Cartridge part number.
- **Mfg. Date** — Manufacturing date.
- **Status** — Current validation status (with color indicator).
- **Syndrome** — Validation syndrome name (if any), e.g., CC-EEPROM-Misprogrammed, CC-Bad. Shown only when a syndrome is reported.
- **Description** — Brief description of the issue. Shown only when available.

- Recommended Actions — Numbered remediation steps. Shown only when available.

For partial_failure status, a warning note is displayed indicating that some slots have failed and the user should check the red-colored slots for individual failure details.

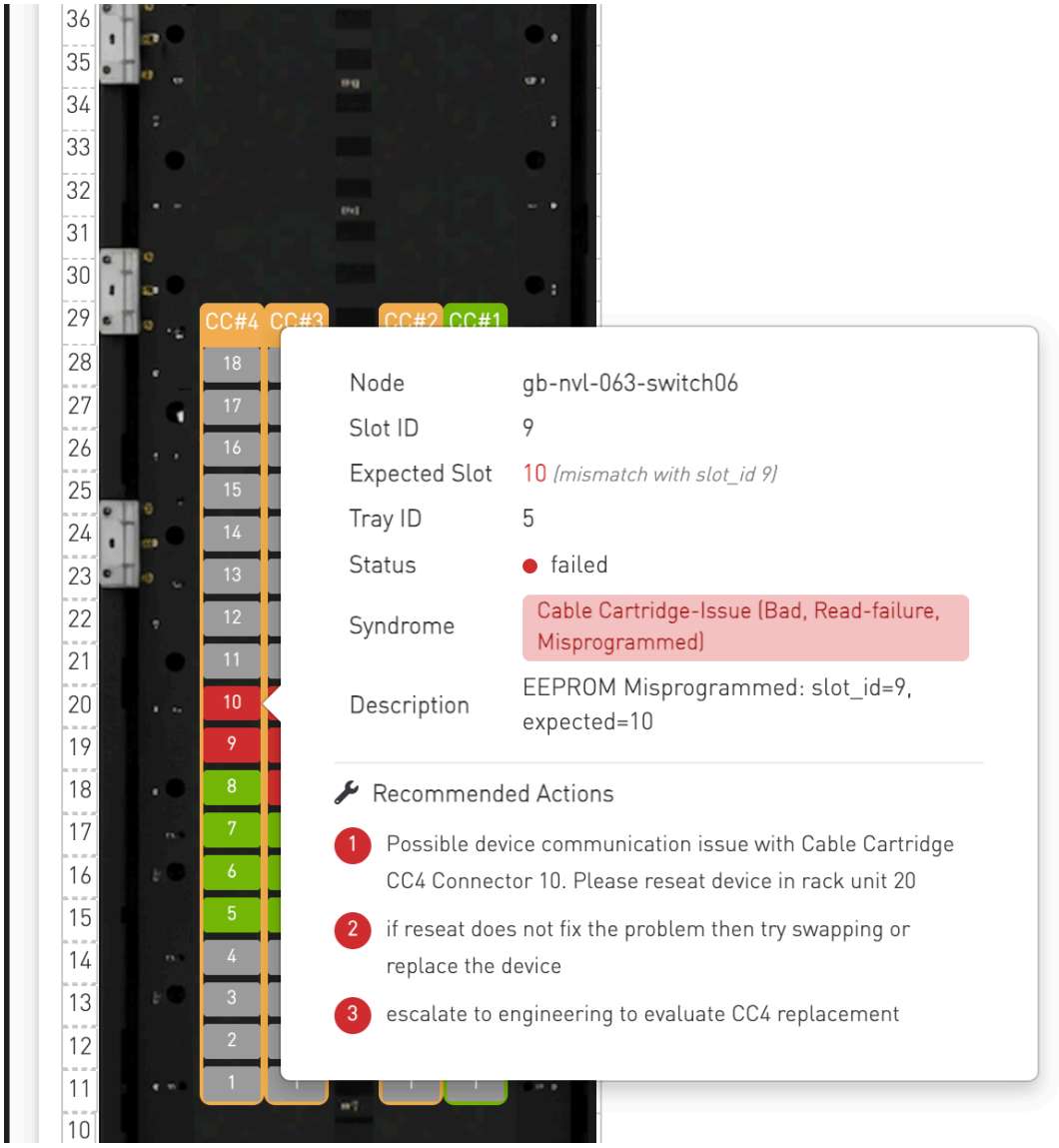


10.4.6.5 Slot-Level Details (Slot Popover)

Clicking on an individual slot block within a CC column opens a popover with device-level details:

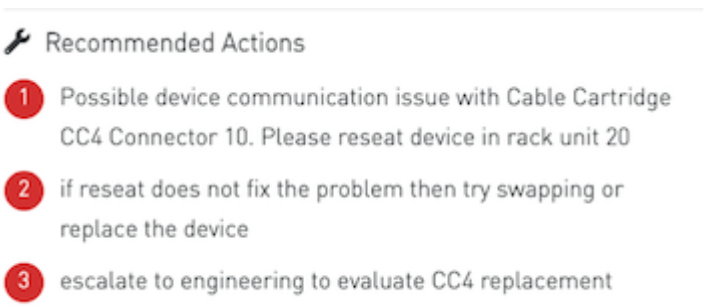
- Node — Device/node name.
- Slot ID — Reported slot ID from the device EEPROM.
- Expected Slot — Expected slot ID derived from the device's rack unit position. If there is a mismatch between the reported slot ID and the expected slot, the expected slot cell is highlighted in red with a mismatch note.
- Tray ID — Reported tray ID.
- Status — Slot-level validation status.

- Syndrome — Slot-level syndrome (e.g., CC-EEPROM-Read-Failure). Shown only when a syndrome is reported.
- Description — Brief description of the issue. Shown only when available.
- Recommended Actions — Numbered remediation steps. Shown only when available.


CC#4
CC#3
CC#2
CC#1
Node
gb-nvl-063-switch06
Slot ID
9
Expected Slot
10 (mismatch with slot_id 9)
Tray ID
5
Status
failed
Syndrome
Cable Cartridge-Issue (Bad, Read-failure, Misprogrammed)
Description
EEPROM Misprogrammed: slot_id=9, expected=10
Recommended Actions
1 Possible device communication issue with Cable Cartridge CC4 Connector 10. Please reseat device in rack unit 20
2 if reseat does not fix the problem then try swapping or replace the device
3 escalate to engineering to evaluate CC4 replacement

10.4.6.6 Recommended Actions

When a validation issue is detected, the popover displays a **Recommended Actions** section with numbered steps guiding the operator through remediation. These actions are generated by the Cable Cartridge validation engine based on the specific syndrome detected.



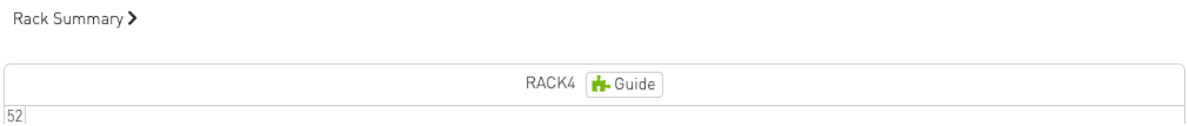
10.4.7 Shuffle Box Visualization

When shuffle-box data is available, the Rack View renders each Shuffle Box as a compact rack element occupying one rack unit.

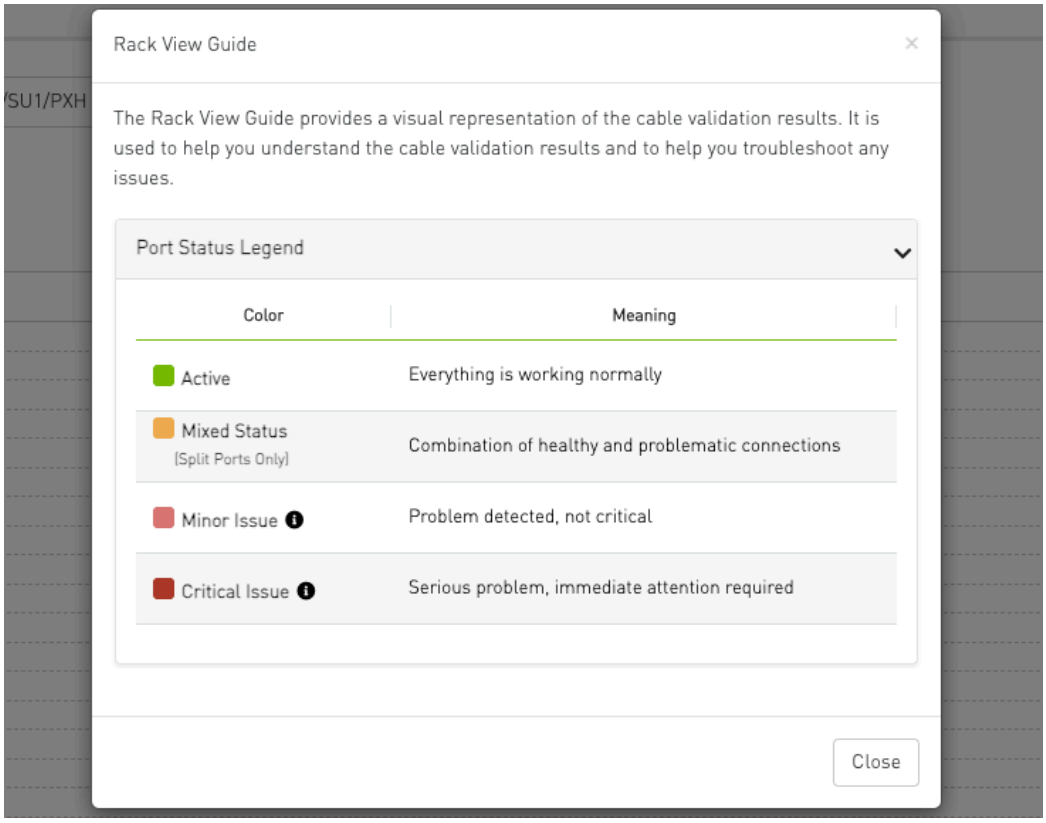
The front face displays cassette LC ports, while the rear face displays MTP ports. Clicking a Shuffle Box or one of its ports opens details such as identity, placement, connection status, validation status, and related circuits.

10.4.8 Rack Guide

The Rack Guide View helps users understand the rack visualization and port status indicators through an interactive modal guide.



It can be accessed by clicking the Rack Guide button in the Rack View, opening a user-friendly modal without leaving the current screen.



Interactive tooltips next to Minor and Critical issues provide detailed lists of specific problems.

10.4.9 Custom Device Configuration

The Rack View supports a user-extension device configuration file (**devices-configuration.user.json**) that allows users to add new device models or override existing built-in device visualizations without modifying the CVT source code.

10.4.9.1 How It Works

On startup, the Rack View loads the built-in device configurations and then loads the user-extension file. User-defined devices are merged into the built-in configuration:

- New devices: Added to the Rack View automatically.
- Existing devices: User entries override the built-in configuration for the same device key.
- Merge scope: Only the **devices** section is merged into the built-in configuration.
  The **regexTemplates** and **modelAliases** sections are used only to resolve references within the user-extension file itself and are not merged into the built-in configuration.
- Validation: The user-extension file is strictly validated on load. If the file contains errors (malformed JSON or invalid structure), it is skipped entirely and a toast notification is displayed with a description of the first error found. If the file is missing, the Rack View silently falls back to the built-in configuration without any notification. In both cases the Rack View continues to function with the built-in configuration only.

10.4.9.2 File Location

The user-extension file is located in the CVT bringup container at::

/opt/ufm/cablevalidation/media/assets/rack-configurations/devices-configuration.user.json

10.4.9.3 Configuration Schema

The file follows this JSON structure:

```
{
  "regexTemplates": {
    "<templateName>": "<regex pattern>"
  },
  "modelAliases": {
    "<aliasDeviceKey>": "<existingDeviceKey>"
  },
  "devices": {
    "<DEVICE_KEY>": {
      "deviceModel": "<string>",
      "rackUnits": <integer>,
      "frontView": {
        "backgroundImage": "<filename.png>",
        "styles": {
          "width": "<px>",
          "height": "<px>"
        },
        "portGroups": {
          "<groupId>": {
            "maxPorts": <integer>,
            "maxRows": <integer>,
            "reverseInterval": <integer>,
            "portRegex": "<regex or $templateRef>",
            "portStyleType": "normal | compressed-vertical | compressed-horizontal",
            "portNumberOffset": <integer>,
            "columnSpacing": [{ "interval": <integer>, "spacing": "<px>" }],
            "rowSpacing": [{ "interval": <integer>, "spacing": "<px>" }],
```

```
            "styles": {
              "container": { },
              "portColumn": { },
              "port": { },
              "splitPort": { }
            }
          }
        },
        "ledsGroups": {
          "<groupId>": {
            "backgroundImage": "<filename.png>",
            "maxRows": <integer>,
            "columnSpacing": [{ "interval": <integer>, "spacing": "<px>" }],
            "rowSpacing": [{ "interval": <integer>, "spacing": "<px>" }],
            "styles": {
              "container": { },
              "leds": { },
              "led": { }
            },
            "items": [
              { "title": "<string>", "healthKeys": ["<key1>", "<key2>"] }
            ]
          }
        }
      }
    }
  }
}
```

10.4.9.4 Field Reference

10.4.9.4.1 Top-Level Fields

Field	Required	Description
`regexTemplates`	No	Reusable regex patterns that can be referenced in `portRegex` fields using `$templateName` syntax.
`modelAliases`	No	Maps alias device keys to existing device keys, allowing multiple hardware variants to share a single configuration.
`devices`	No	Map of device configurations keyed by device model identifier.

10.4.9.4.2 Device Fields

Field	Required	Description
`deviceModel`	Yes	Device model identifier string (must match the model reported by the system or specified in the P2P file).
`rackUnits`	No	Number of rack units the device occupies. Defaults to 2 for switches, 1 for hosts.
`frontView`	No	Front-view layout configuration. Contains the visual layout for the front-facing side of the device.

10.4.9.4.3 View Configuration Fields (frontView)

Field	Required	Description
`backgroundImage`	Yes	Filename of the device background image (placed in `assets/img/`).
`styles`	Yes	CSS styles for the device container (typically `width` and `height` in pixels).
`portGroups`	No	Map of port group configurations (see below).
`ledsGroups`	No	Map of LED group configurations (see below).

10.4.9.4.4 Port Group Fields

Field	Required	Description
`maxPorts`	Yes	Maximum number of ports in the group.
`maxRows`	Yes	Number of rows in the port layout.
`reverseInterval`	No	Number of consecutive rows sharing the same reverse state. E.g., 1 = alternate every row, 2 = alternate every 2 rows.
`portRegex`	Yes	Regex pattern (or array of patterns) with named capture groups `port` and optionally `s1` for single-level splits and `s2` for two-level splits. Use `$templateName` to reference a regex template defined in the same file's `regexTemplates` section.
`portStyleType`	No	Port rendering style: `normal` (default), `compressed-vertical`, or `compressed-horizontal`.
`portNumberOffset`	No	Offset to add to extracted port numbers (default: 0). Set to 1 for 0-indexed split numbering.
`columnSpacing`	No	Array of spacing levels. Each entry has `interval` (apply spacing every N columns) and `spacing` (CSS value, e.g., `"245px"`).

Field	Required	Description
`rowSpacing`	No	Array of spacing levels for rows (same format as `columnSpacing`).
`styles`	Yes	CSS styles at four levels: `container` (positions the group on the device), `portColumn`, `port`, and `splitPort`. The validator requires `styles` to be a non-empty object; individual keys (`container`, `portColumn`, etc.) are not enforced but `container` is expected by the renderer.

10.4.9.4.5 LED Group Fields

Field	Required	Description
`backgroundImage`	No	Optional background image for the LED panel.
`maxRows`	No	Number of LED rows per column. Defaults to 1 (horizontal). Set to items.length for vertical.
`columnSpacing`	No	Spacing between LED columns (same format as port group `columnSpacing`). Note: LED spacing fields are not validated on load, unlike port group spacing.
`rowSpacing`	No	Spacing between LED rows within a column (same format as port group `rowSpacing`). Note: LED spacing fields are not validated on load, unlike port group spacing.
`styles`	Yes	CSS styles at three levels: `container` (positions the LED panel on the device), `leds` (dot wrapper), and `led` (individual dot).
`items`	Yes	Array of LED items (must be non-empty). Each item may include an optional `title` (tooltip text) and an optional `healthKeys` (array of health summary keys to evaluate, e.g., `["Power Supply", "Fans"]`).

10.4.9.5 Example

Below is an example configuration for the DGX-B300 server, which has two port groups (OSFP and BlueField) and occupies 10 rack units:

```
{
  "regexTemplates": {
    "osfp": "OSFP(?<port>\\d+)(?:\\s*P(?<s1>\\d+))?",
    "slot": "Slot\\s*(?<port>\\d+)(?:\\s*Port\\s*(?<s1>\\d+))?",
    "slashSeparatedPort": "[A-Za-z]+(?<port>\\d+)/(?<s1>\\d+)"
  },
  "modelAliases": {
  },
  "devices": {
    "DGXB300": {
      "deviceModel": "DGXB300",
      "rackUnits": 10,
      "frontView": {
        "backgroundImage": "dgx-b300.png",
        "styles": {
          "width": "724px",
          "height": "492px"
        },
        "portGroups": {
          "osfp": {
            "maxPorts": 8,
            "maxRows": 1,
            "portRegex": ["$osfp", "$slashSeparatedPort"],
            "columnSpacing": [
              { "interval": 4, "spacing": "245px" }
            ],
            "styles": {
              "container": { "top": "347px", "left": "61px" },
              "portColumn": { "width": "5.1%", "margin-right": "9px" },
              "splitPort": { "font-size": "12px" }
            }
          },
          "bluefield": {
            "maxPorts": 2,
            "maxRows": 1,
            "portRegex": "$slot",
            "columnSpacing": [
              { "interval": 1, "spacing": "370px" }
            ],
            "styles": {
              "container": { "bottom": "81px", "left": "120px" },
              "port": { "background-color": "transparent !important", "color": "transparent !important", "width":
"78px" },
              "splitPort": { "font-size": "12px" }
            }
          }
        }
      }
    }
  }
}
```

10.4.9.6 Validation Rules

The user-extension file is validated on load. If any of the following conditions are not met, the entire file is skipped:

- The file must be valid JSON.
- **deviceModel** is required for each device.
- Within **frontView**: **backgroundImage** and **styles** are required. **portGroups** and **ledsGroups** are optional.
- Each port group must have **maxPorts**, **maxRows**, **portRegex**, and **styles**.
- All **portRegex** patterns must be valid regular expressions. Template references (**$templateName**) must resolve to an entry in the same file's **regexTemplates** section (built-in templates are not available to the user-extension file).
- Spacing arrays must contain objects with **interval** (integer) and **spacing** (string) fields.
- LED groups must have **styles** and **items** fields.

When validation fails, a toast notification is displayed with a description of the first error found.

10.5 Tests

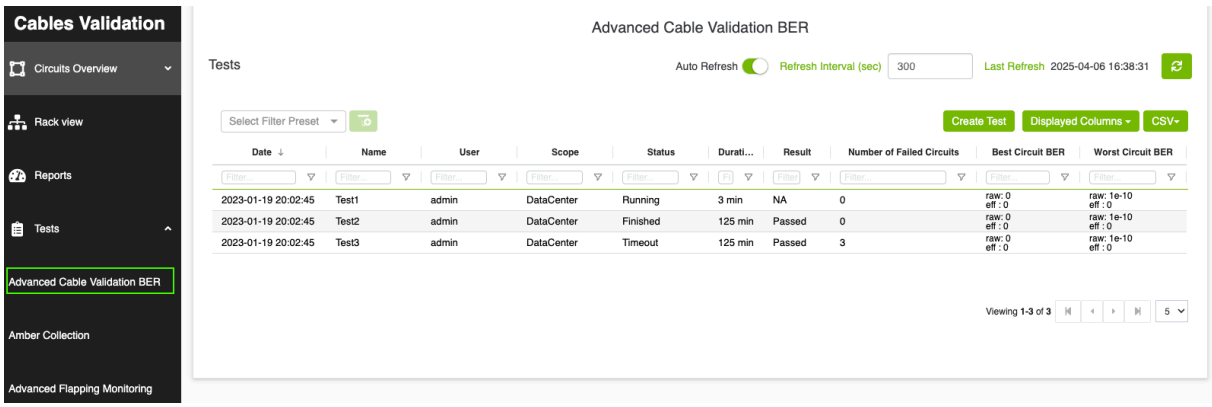
The Cable Validation Tool (CVT) allows authorized Admin users to run several types of network validation tests and manage their results efficiently. Access to these features is limited to users with administrative permissions.

The tool saves the test results automatically, so you can always review both current and past test outcomes—even after restarting the application.

10.5.1 Test Types and Workflows

10.5.1.1 Golden BER Test

The Golden BER Test is designed to monitor Bit Error Rates (BER) and analyze the interface counters. The process begins by resetting interface counters on connected agents using the command sudo mlxlink –pc. The test then runs for a user-specified duration, during which the agents periodically report updated counter values every 10 minutes to a centralized collector. When the test completes, the final Amber file is sent to the collector.



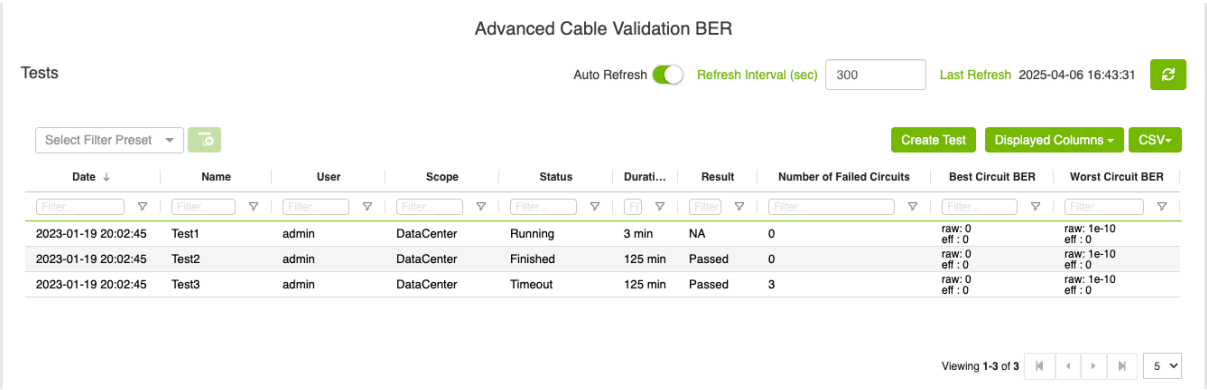
In the UI, the test is presented in a tabular format with the following columns:

1. Date: The creation date of the test.
2. Name: The name assigned to the test.
3. User: The creator of the test.
4. Scope: Test scope, which usually includes SU<n>/DH<n>, the DataCenter, or comma-separated racks or nodes.
5. Duration: The total runtime of the test.
6. Result: Displays the test's outcome, which can be:
    a. NA: Initial state, test not yet completed.
    b. Failed: If any circuit's BER status is not "Good." BER status is determined by comparing factors such as SerDes Technology (16nm/7nm/5nm), Link Speed Active,

Active FEC, Raw BER, and Effective BER against specific thresholds(the same logic used in the Underperforming link). Depending on these values, the status is categorized as Good, Poor, or Marginal.

c. Passed: All circuits have a "Good" BER status.

7. Status: test status, and it could be one of the following values:
   a. Running: test is running.
   b. Timeout: The test running time exceeded the test duration
   c. Finished: The test is successfully finished, and the collector has received the collected Amber files.
   d. Pending: The test is in its initial state.
   e. Stopping: A stop request has been acknowledged, but the test is still winding down (unsubscribing agents and releasing nodes).
   f. Stopped: An error occurred while the test is running.
8. Number of Failed Circuits: The count of circuits with a "Failed" BER status.
9. Best Circuit BER: The lowest value of Raw BER or Effective BER during the test.
10. Worst Circuit BER: The highest value of Raw BER or Effective BER during the test.

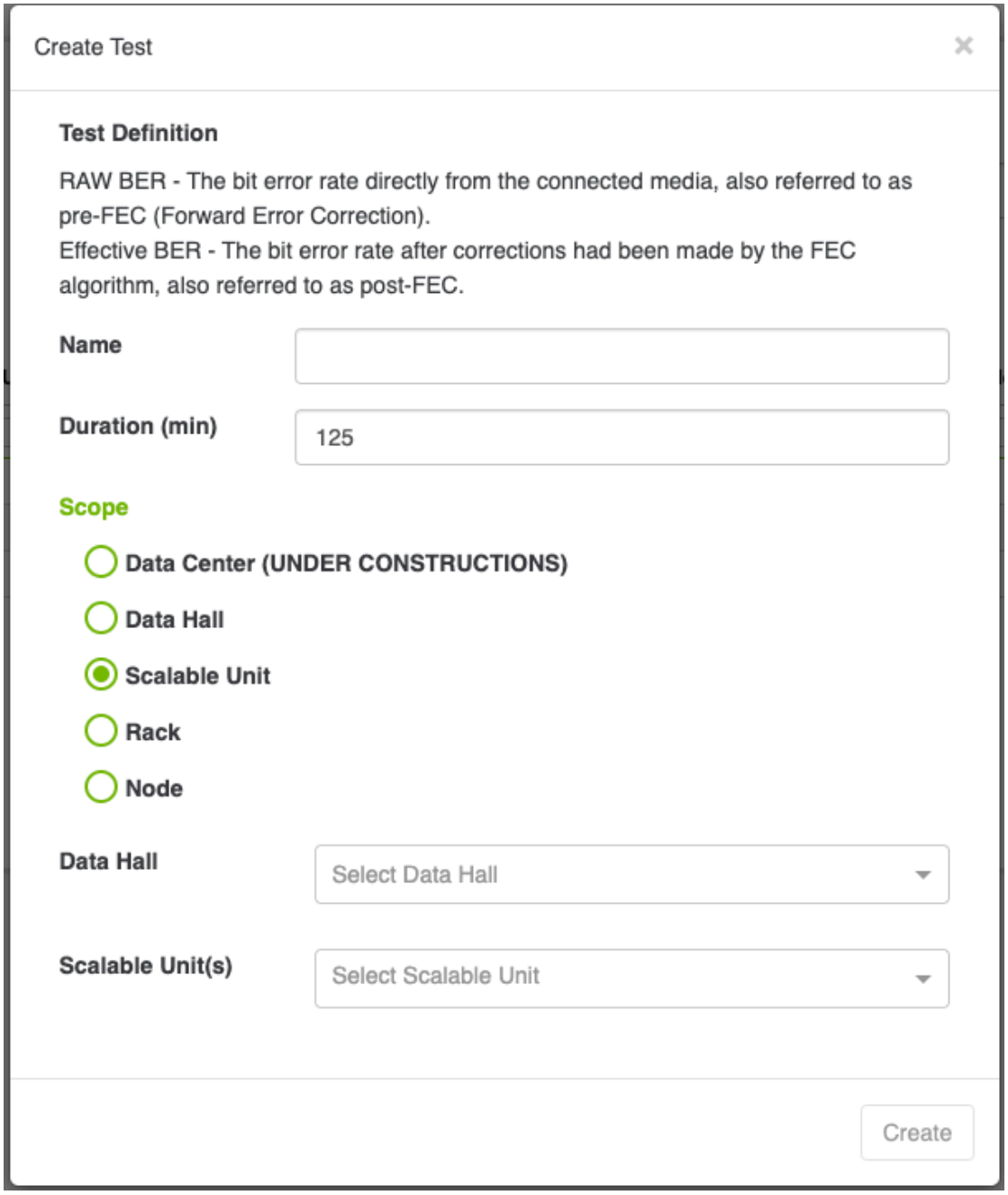


10.5.1.1.1 Create New Test

To initiate a new test, the user clicks the Create Test button, which opens a modal dialog.
The tool can create up to five tests. If the maximum limit is reached and a new test is initiated, the system will automatically remove the oldest test to maintain a maximum of five tests in memory.

Multiple tests should be able to run simultaneously if they are on different nodes. For example, if the Golden BER Test is already running on [node1, node2], and an Advanced Flapping Test is requested for [node1, node4], the flapping test will fail because node1 is already occupied by the Golden BER Test.

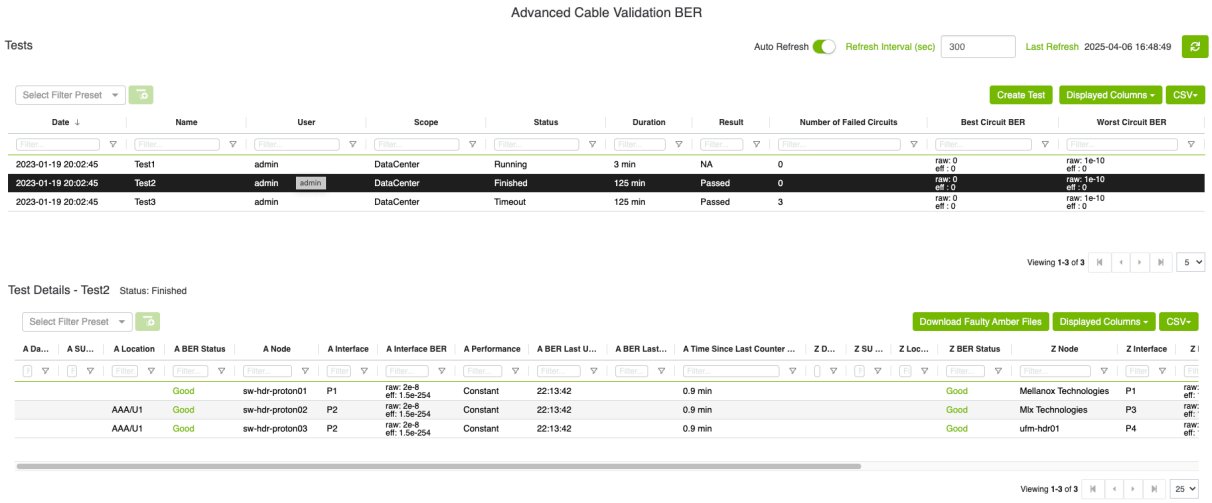
The collected amber will be saved for only 6 hours, after which it will be deleted.
Please note that the user should load the DC layout file in order to use the Scalable Unit and Data Hall options



After creation, the test is added to the table. Clicking on the test allows users to view its detailed results.

10.5.1.1.2 Test Details

Detailed test results are displayed in a table, covering both A and Z endpoints.

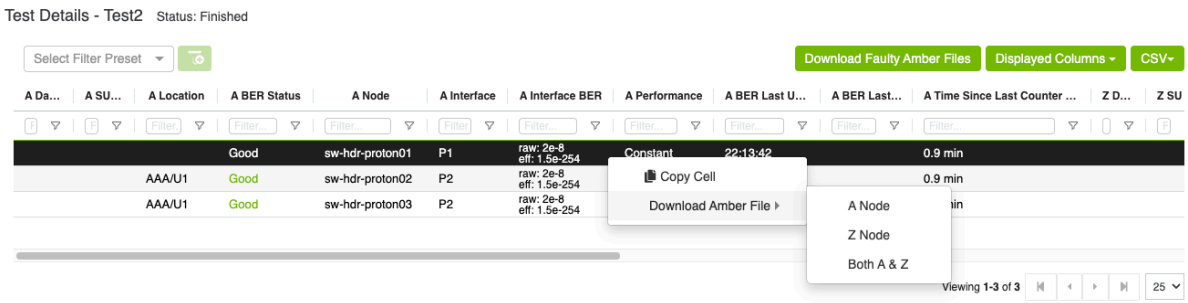


The columns include:

1. **Protocol**: circuit protocol (IB, Ethernet or NVLink).
2. **Data Hall / SU Number**: Data Hall and Scalable Unit
3. **Location**: Specifies the rack and unit location of the endpoint.
4. **BER Status**: Calculated based on the methodology mentioned earlier.
5. **Node**: The name of the node.
6. **Interface**: The port under analysis.
7. **BER Counters**: Displays **Raw BER** and **Effective BER** counters.
8. **Performance**: Indicates performance trends:
    - **NA**: No available **Raw BER** value or the first recorded value.
    - **Improved**: If the **Raw BER** value has decreased.
    - **Degraded**: If the **Raw BER** value has increased.
    - **Constant**: If the value remains unchanged.
9. **BER Last Update**: The timestamp of the last BER update.
10. **BER Last Update Duration**: Time since the most recent BER update.
11. **Time Since Last Counter Clear**: The elapsed time since the counters were last cleared.

10.5.1.1.2.1 Downloading Amber Files

After the test finished, user can download the Amber file for each circuit by clicking on the circuit then right-click

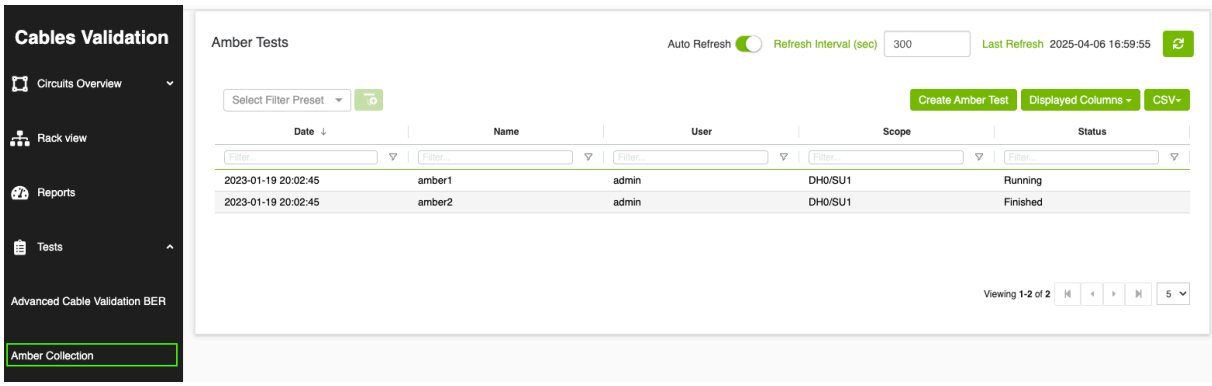


User can download the amber file for A, Z or both endpoints.

The download context menu items (A Node, Z Node, Both A & Z) are individually enabled or disabled based on whether the corresponding amber file actually exists on disk. If a file is not available, the corresponding menu item is grayed out and cannot be clicked.
Also, user can download faulty Amber files by click on Download Faulty Amber Files button, which will be hidden when the faulty circuits file does not exist on disk, instead of being shown and potentially returning a download error.

10.5.1.2 Amber Collection

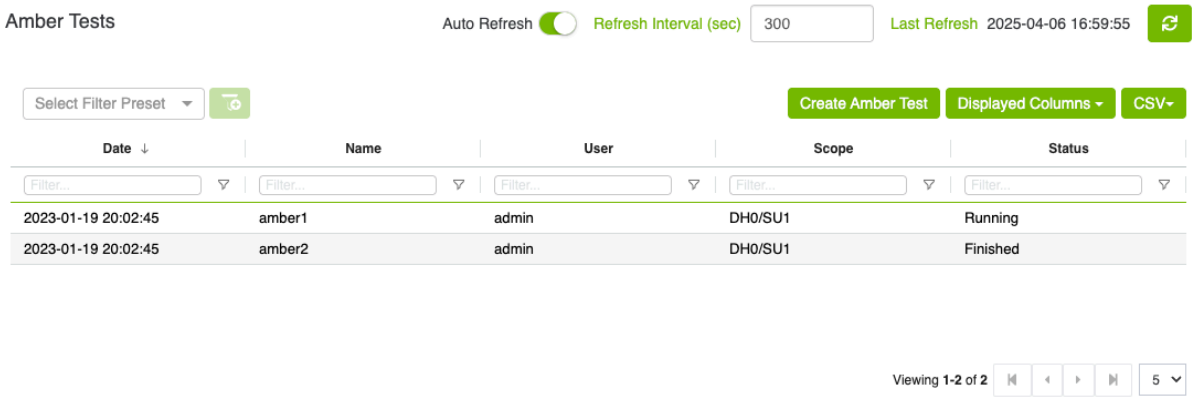
The Amber Collection test is designed to Collecting Amber File upon request. The process starts by utilizing the mlxlink command to gather amber files from connected agents within the specified test scope. Once all agents complete the collection and send the amber files to the collector, these files are compressed into a single tar file and stored in the directory:/cable_bringup_root/data/tests/amber/<test_name> and the file name will be <test_name>.tar.gz. Afterward, the collector marks the test as Finished.



In the UI, the test is presented in a tabular format with the following columns:

1. Date: The creation date of the test.
2. Name: The name assigned to the test.
3. Scope: Test scope, which usually includes SU<n>/DH<n>, the DataCenter, or comma-separated racks or nodes.

4. **Status:** The current status of the test, which can be one of the following:
    a. **Running**: The test is ongoing, and not all agents have returned their amber files.
    b. **Timeout**: The test has exceeded the 5-minute time limit (test timeout = 5 min), and not all agents have returned their amber files.
    c. **Finished**: All agents have successfully returned their amber files.
    d. **Pending**: The test is in its initial state, awaiting execution.
    e. **Stopping:** A stop request has been acknowledged, but the test is still winding down (unsubscribing agents and releasing nodes).
    f. **Stopped:** An error occurred while the test is running.



10.5.1.2.1 Create New Test

To initiate a new test, the user clicks the Create Test button, which opens a modal dialog.
The tool can create up to five tests. If the maximum limit is reached and a new test is initiated, the system will automatically remove the oldest test to maintain a maximum of five tests in memory.

Please note that the user should load the DC layout file in order to use the Scalable Unit and Data Hall options
After creation, the test is added to the table. Clicking on the test allows users to view its detailed results.

10.5.1.2.2 Test Details

Amber Tests

Auto Refresh | Refresh Interval (sec) 300 | Last Refresh 2025-04-06 17:04:20

Select Filter Preset | Create Amber Test | Displayed Columns | CSV

Date ↓	Name	User	Scope	Status
2023-01-19 20:02:45	amber1	admin	DH0/SU1	Running
2023-01-19 20:02:45	amber2	admin	DH0/SU1	Finished

Viewing 1-2 of 2 | 5

Test Details - amber2 Status: Finished

Select Filter Preset | Download Amber Files | Displayed Columns | CSV

Data Hall	SU	Rack Location	Node Name	IP	Node Type	Status
DH1	SU2	AAA/1	sw-hdr-proton01	10.209.44.74	Switch	Finished
DH1	SU2	AAA/1	sw-hdr-proton02	10.209.44.75	Switch	Finished
DH1	SU2	AAA/1	sw-hdr-proton03	10.209.44.76	Switch	Finished
DH1	SU2	AAA/1	sw-hdr-proton03	10.209.44.76	Switch	Finished
DH1	SU2	AAA/1	sw-hdr-proton01	10.209.44.74	Switch	Finished

Viewing 1-5 of 16 | 5

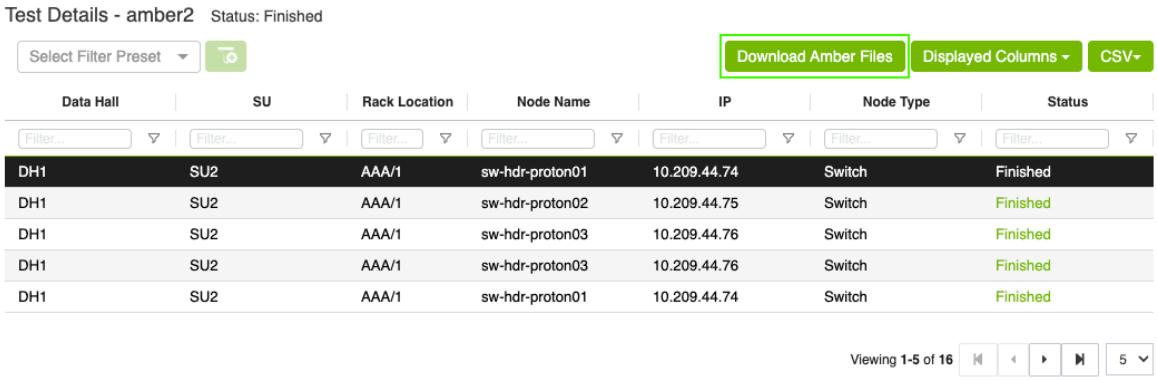
Detailed test results are displayed in a table, covering the node information where the test is running. The columns include:

1. Data Hall / SU Number: Data Hall and Scalable Unit
2. Location: Specifies the rack and unit location of the endpoint.
3. Node Name: The name of the node.
4. IP: The IP address.
5. Node Type: switch or host.
6. Status: Indicates the status of amber file collection for each agent:
    - Running: Amber file collection is in progress.
    - Timeout: The amber file collection has exceeded the 5-minute time limit without completion.
    - Finished: The amber file is successfully collected, and the collector has received it.
    - Pending: The test is in its initial state.
    - Stopped: An error occurred during the amber file collection process.

10.5.1.2.3 Downloading Amber Files

Once a test is marked as **Finished**, users can download the amber files by clicking the **Download Amber Files** button. This will download the compressed tar file containing all the collected amber files.

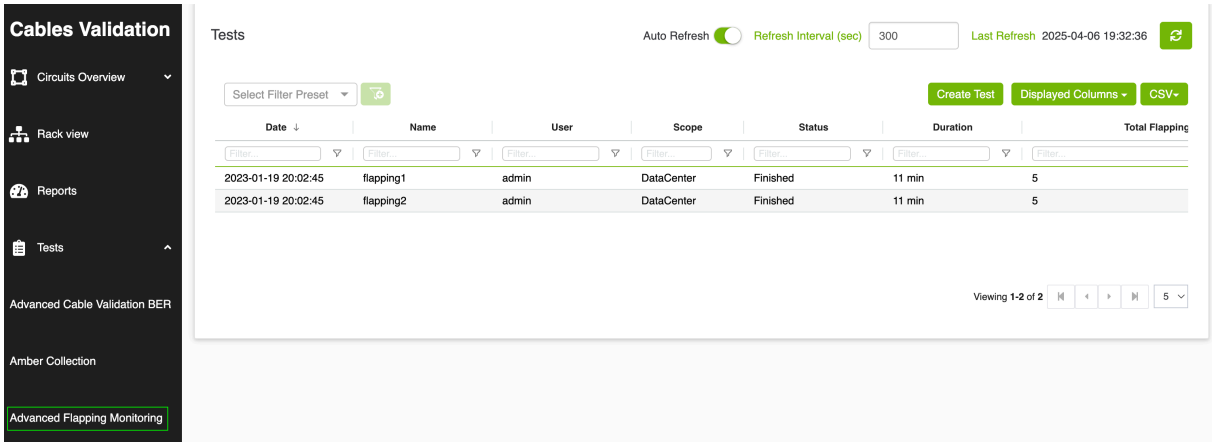
The **Download Amber Files** button will be hidden when the compressed amber tar file does not exist on disk, instead of being shown and potentially returning a download error.



10.5.1.3 Advanced Flapping Monitoring

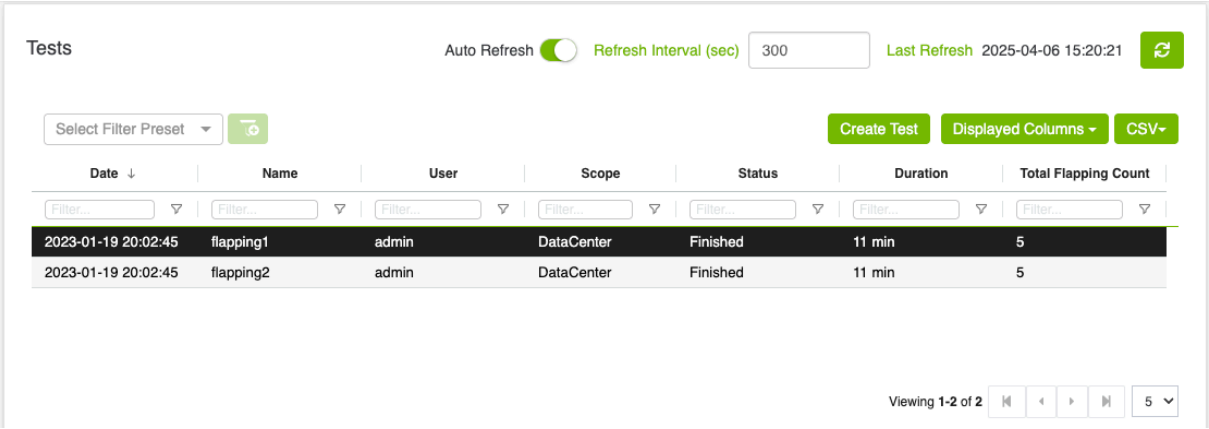
The **Advanced Flapping Monitoring** is designed to analyze the metrics that could lead to a flapping event. It tracks the circuit flapping events along with Bit Error Rates (BER), temperature, and Rx/Tx counters.

The process begins by resetting interface counters on connected agents using the command sudo mlxlink –pc, along with resetting the **flap counter** in **CVT memory**. Once initialized, the test runs for a user-defined duration, during which agents periodically report updated counter values to the collector every **10 minutes**.



In the UI, the test is presented in a tabular format with the following columns:

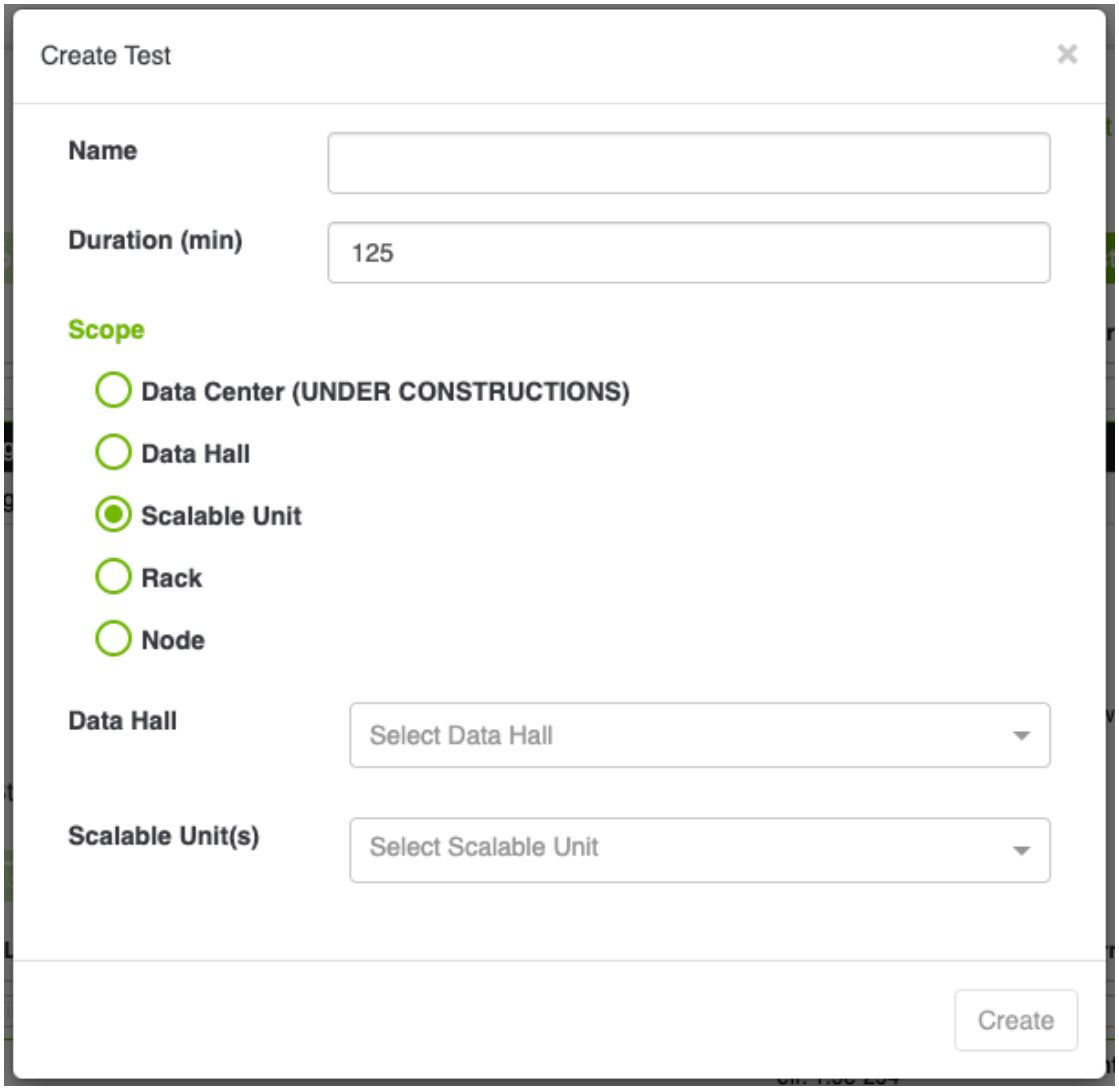
1. Date: The creation date of the test.
2. Name: The name assigned to the test.
3. User: The creator of the test.
4. Scope: Test scope, which usually includes SU<n>/DH<n>, the DataCenter, or comma-separated racks or nodes.
5. Duration: The total runtime of the test.
6. Status: test status, and it could be one of the following values:
    a. Running: test is running.
    b. Timeout: The test running time exceeded the test duration
    c. Finished: The test is successfully finished, and the collector has received the advanced stats.
    d. Pending: The test is in its initial state.
    e. Stopping: A stop request has been acknowledged, but the test is still winding down (unsubscribing agents and releasing nodes).
    f. Stopped: An error occurred while the test is running.
7. Total Flapping Count: The total flaps occurred across all switches for all ports from the time the test started.



10.5.1.3.1 Create New Test

To initiate a new test, the user clicks the Create Test button, which opens a modal dialog. The tool can create up to five tests. If the maximum limit is reached and a new test is initiated, the system will automatically remove the oldest test to maintain a maximum of five tests in memory.

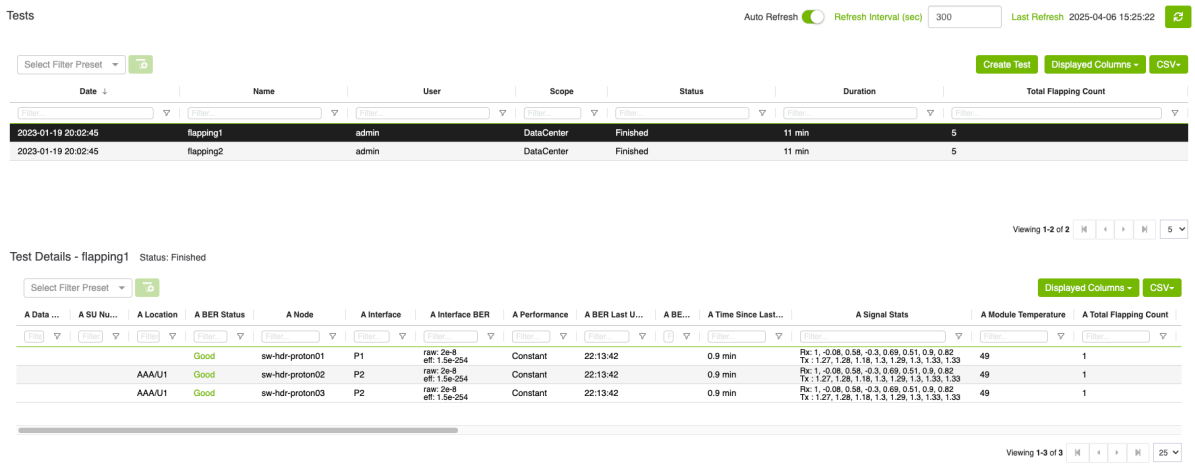
Multiple tests should be able to run simultaneously if they are on different nodes. For example, if the Golden BER Test is already running on [node1, node2], and an Advanced Flapping Test is requested for [node1, node4], the flapping test will fail because node1 is already occupied by the Golden BER Test.



Please note that the user should load the DC layout file in order to use the Scalable Unit and Data Hall options
After creation, the test is added to the table. Clicking on the test allows users to view its detailed results.

Note: If the test duration expires while the test is still incomplete due to errors or lack of responses from any agent within the test scope, a 5-minute buffer time will be added. As a result, the total test duration will be the original test time plus 5 minutes.

10.5.1.3.2 Test Details



Detailed test results are displayed in a table, covering both A and Z endpoints. The columns include:

1. Protocol: circuit protocol (IB, Ethernet or NVLink).
2. Data Hall / SU Number: Data Hall and Scalable Unit
3. Location: Specifies the rack and unit location of the endpoint.
4. BER Status: BER status is determined by comparing factors such as SerDes Technology (16nm/7nm/5nm), Link Speed Active, Active FEC, Raw BER, and Effective BER against specific thresholds(the same logic used in the Underperforming link). Depending on these values, the status is categorized as Good, Poor, or Marginal.
5. Node: The name of the node.
6. Interface: The port under analysis.
7. BER Counters: Displays Raw BER and Effective BER counters.
8. Performance: Indicates performance trends:
    a. NA: No available Raw BER value or the first recorded value.
    b. Improved: If the Raw BER value has decreased.
    c. Degraded: If the Raw BER value has increased.
    d. Constant: If the value remains unchanged.
9. BER Last Update: The timestamp of the last BER update.
10. BER Last Update Duration: Time since the most recent BER update.
11. Time Since Last Counter Clear: The elapsed time since the counters were last cleared.
12. Signal Stats: Shows transmission and reception power levels (Rx/Tx Power Lane)
13. Module Temperature: module temperature value.
14. Total Flapping Count: This is the total number of transitions occurring during the test.
15. Flapping Status: flapping status and it could be one of the following values:
    a. Ok - no flapping events during the test.
    b. Flapping - agent detected a flapping event in the last 1 minute
    c. Flapped - agent detected a flapping event at some point while the test is running (The Flapping event counter >= 1)

16. **Flap 30 sec:** how many flaps happened in the last 30 seconds (hidden by default).
17. **Flap 1 min:** how many flaps happened in the last 1 minute (hidden by default).
18. **Flap 5 min:** how many flaps happened in the last 5 minutes (hidden by default).
19. **Flap 1 hour:** how many flaps happened in the last 1 hour (hidden by default).
20. **Flap 12 hour:** how many flaps happened in the last 12 hours (hidden by default).
21. **Flap 24 hour:** how many flaps happened in the last 24 hours (hidden by default).

10.5.2 Managing Tests

- You can have up to 5 active tests at a time. If you start a new test beyond this limit, the oldest test's Amber files are deleted, but the test details remain available.
- Amber files in the golden BER tests are kept for 6 hours after test completion. After that, only the test details remain.
- Tests without test files will be shown in gray to indicate that the test file was deleted.
- Multiple tests can run at the same time if they use different nodes.
- Make sure to load the DC layout file to use Scalable Unit and Data Hall options.
- If a test exceeds its duration, a 5-minute buffer is automatically added before the test is finalized.
- When stopping a test, the status transitions through **Stopping** before reaching **Stopped**, giving users clear visual feedback that the system is still winding down.

10.5.3 Test Results, Storage, and Retention

All test results and related information are automatically saved in a database, ensuring that your test history remains accessible at any time, even after restarting CVT.

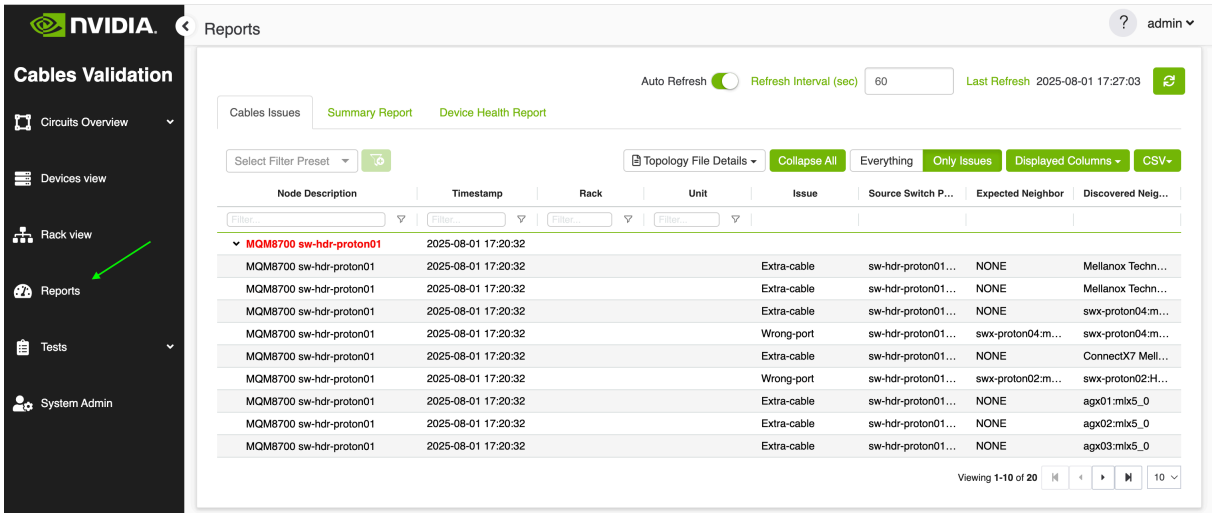
What Happens When Limits Are Reached:

- If you exceed 5 active tests, the oldest test's Amber files are deleted automatically, but the test record stays available.
- In golden BER tests, If a test runs longer than 6 hours, its Amber files are deleted, but test details are still stored.

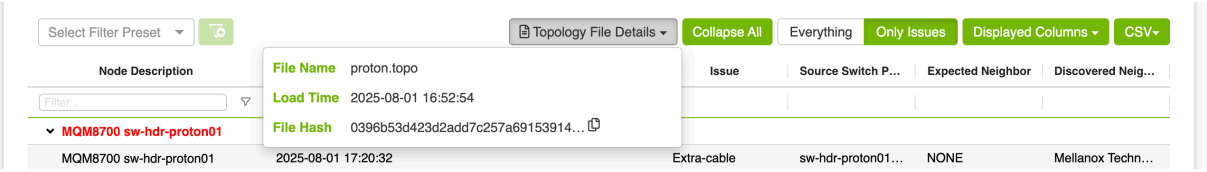
10.6 Reports

10.6.1 Cables Issues

This tab is only supported for Admin users



The **Cable Issues View** provides a table-based overview of node related issues, enabling users to quickly identify and resolve problems. The view organizes all issues related to a specific node under a single collapsible row.
By default, all rows are fully expanded, showing details for each issue, users can manually collapse individual nodes to hide their associated issues or click the **'Collapse All'** button to collapse all nodes at once.
Only nodes with issues are displayed by default. And the description of a node with issues is highlighted in **red** for easy identification.
To view all nodes, including those without issues, click the **'Everything'** button.
To see topology details file you can click on **'Topology File Details'** which include the File Name, Load Time and the File Hash.



Navigate to Circuit

Users can directly navigate to the circuit associated with a specific issue:

1. **Right-click** on the issue in the table.
2. Select **'Go To Circuit'** to open the circuit view for that issue.

Select Filter Preset | Topology File Details | Collapse All | Everything | Only Issues | Displayed Columns | CSV

Node Description	Timestamp	Rack	Unit	Issue	Source Switch P...	Expected Neighbor	Discovered Neig...
MQM8700 sw-hdr-proton01	2025-08-01 17:20:32						
MQM8700 sw-hdr-proton01	2025-08-01 17:20:32			Extra-cable	sw-hdr-proton01...	NONE	Mellanox Techn...
MQM8700 sw-hdr-proton01	2025-08			Extra-cable	sw-hdr-proton01...	NONE	Mellanox Techn...
MQM8700 sw-hdr-proton01	2025-08			Extra-cable	sw-hdr-proton01...	NONE	swx-proton04:m...
MQM8700 sw-hdr-proton01	2025-08-01 17:20:32			Wrong-port	sw-hdr-proton01...	swx-proton04:m...	swx-proton04:m...
MQM8700 sw-hdr-proton01	2025-08-01 17:20:32			Extra-cable	sw-hdr-proton01...	NONE	ConnectX7 Mell...
MQM8700 sw-hdr-proton01	2025-08-01 17:20:32			Wrong-port	sw-hdr-proton01...	swx-proton02:m...	swx-proton02:H...
MQM8700 sw-hdr-proton01	2025-08-01 17:20:32			Extra-cable	sw-hdr-proton01...	NONE	agx01:mlx5_0
MQM8700 sw-hdr-proton01	2025-08-01 17:20:32			Extra-cable	sw-hdr-proton01...	NONE	agx02:mlx5_0
MQM8700 sw-hdr-proton01	2025-08-01 17:20:32			Extra-cable	sw-hdr-proton01...	NONE	agx03:mlx5_0

Copy Cell
Go To Circuit

Viewing **1-10** of **20** 10

10.6.2 Summary Report

The **Summary Report View** provides a table summarizing all detected issues or syndromes across the tool. The table includes the number of occurrences and number of affected nodes for each issue.

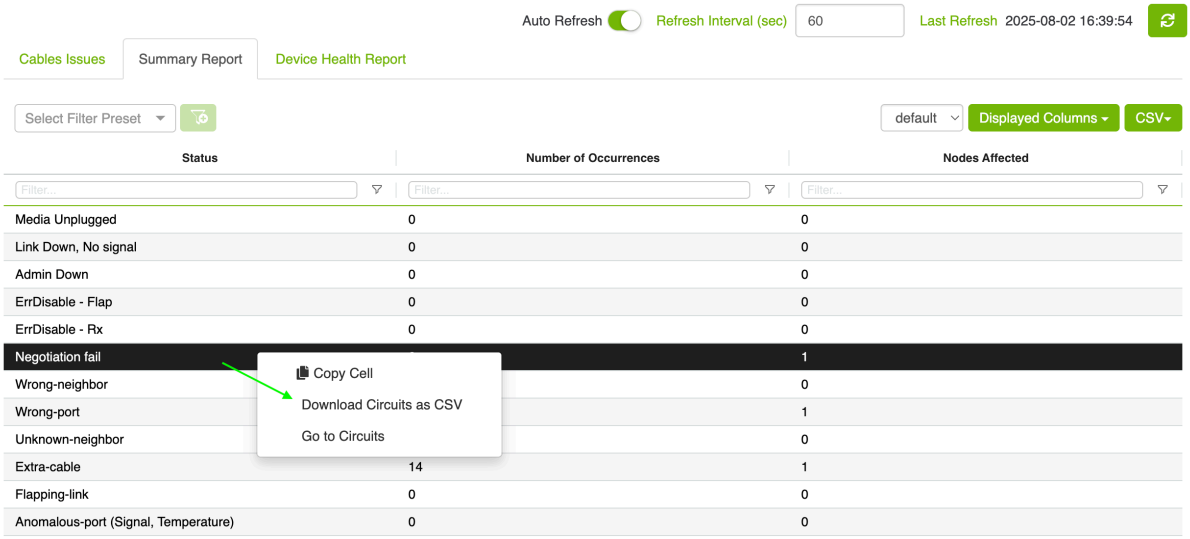
Cabling Issues | Summary Report | Device Health Report

Select Filter Preset | Displayed Columns | CSV

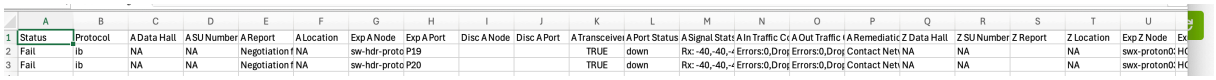
Status	Number of Occurrences	Nodes Affected
Media Unplugged	0	0
Link Down, No signal	3	3
Admin Down	4	2
ErrDisable - Flap	0	0
ErrDisable - Rx	0	0
Negotiation fail	0	0
Wrong-neighbor	4	1
Wrong-port	0	0
Unknown-neighbor	1	1
Extra-cable	37	2
Flapping-link	0	0
Anomalous-port (Signal, Temperature)	0	0
NIC name-mismatch	6	6
Module part number mismatch	0	0
RX/TX power mismatch	0	0
Cable Cartridge-Issue (Bad, Read-failure, Misprogrammed)	0	0
Cluster App-Issue (No master, App failure, Degraded)	0	0
Correctly-wired		0
No report		7

10.6.2.1 Download Circuits as CSV

Users can download all circuits associated with a specific syndrome as a CSV file by right-clicking on the syndrome and selecting "Download as CSV."



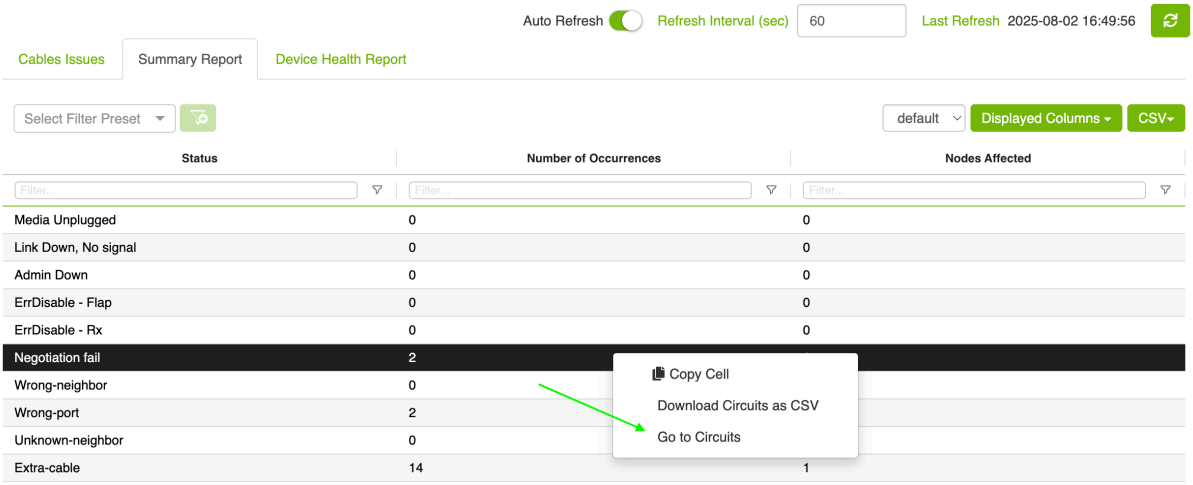
For example, selecting "Download as CSV" for the *Negotiation Fail* syndrome will generate a file that includes only circuits with that specific issue.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1	Status	Protocol	A Data Hall	A SU Number	A Report	A Location	Exp A Node	Exp A Port	Disc A Node	Disc A Port	A Transceiver	A Port Status	A Signal Stats	A In Traffic Co	A Out Traffic (	A Remediatic	Z Data Hall	Z SU Number	Z Report	Z Location	Exp Z Node
2	Fail	ib	NA	NA	Negotiation f	NA	sw-hdr-proto	P19			TRUE	down	Rx: -40,-40,-4	Errors:0,Dro	Errors:0,Dro	Contact Net	NA	NA		NA	swx-proton0:
3	Fail	ib	NA	NA	Negotiation f	NA	sw-hdr-proto	P20			TRUE	down	Rx: -40,-40,-4	Errors:0,Dro	Errors:0,Dro	Contact Net	NA	NA		NA	swx-proton0:

10.6.2.2 Go to Circuits

Users can also navigate to the Circuits View filtered by a specific syndrome by right-clicking on the syndrome and selecting "Go to Circuits."
Within the Circuits View, users can further download the filtered data as a CSV file if needed.

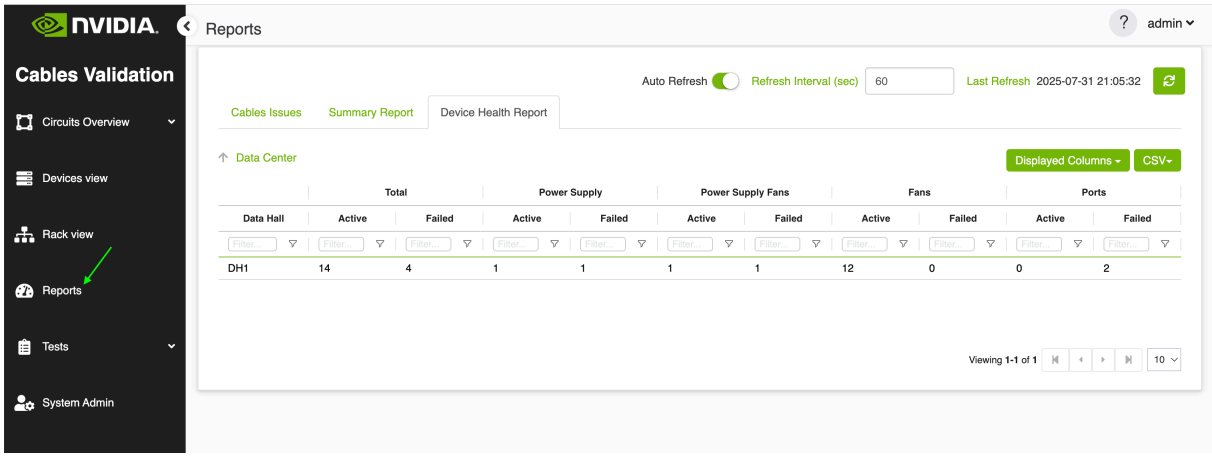


10.6.3 Device Health Report

The **Device Health Report** provides a comprehensive, hierarchical view of device health status across the data center infrastructure. It enables users to monitor key device components—such as power supplies, fans, power supply fans, and ports—across various organizational levels, including Data Center, Data Hall, Scalable Unit, and Rack. Through a drill-down interface, users can navigate these levels to examine the health status in detail. At each level, the report displays values indicating the number of active and failed components (e.g., fans, power supplies) within the corresponding Data Hall, Scalable Unit, Rack, or individual device—depending on the selected level.

The feature displays device health information in a tabular format with the ability to:

- Navigate through hierarchical levels (DC → DH → SU → Rack)
- View detailed health status for Power Supply, Power Supply Fans, Fans, and Ports
- visual indicators for device status



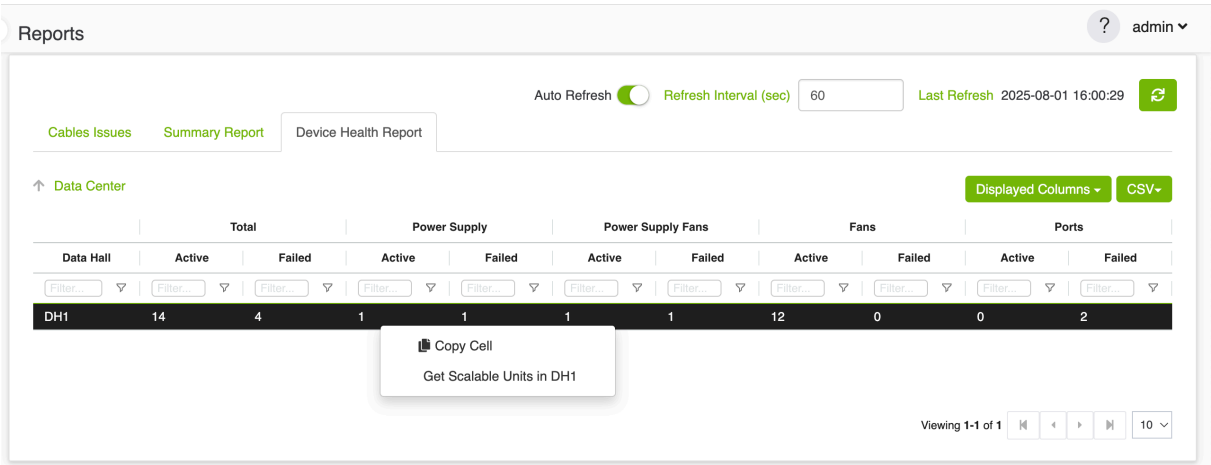
10.6.3.1 Topology Levels

The system provides four levels of hierarchy, listed from the largest to the smallest:

1. Data Center (DC): This level is always available and displays data based on the loaded topology. For example, if the user loads a DC floor layout file, the DC level will show all Data Halls (DHs) in the topology.
   This level will be described in detail later in this document.
2. Data Hall (DH):
   Displays all Scalable Units (SUs) within the selected Data Hall.
   This level is not always available and is only accessible when a DC floor layout file is loaded.
3. Scalable Unit (SU):
   Displays all Racks within the selected SU.
   Like the DH level, this level is also only available if a DC floor layout file has been loaded.
4. Rack:
   Displays all nodes within the selected Rack.
   This level is not always available either; it becomes accessible when the topology is loaded in P2P format.

10.6.3.2 Navigating Deeper

Users can right-click on a row to drill down into a more detailed view.



After selecting "Get sus in DH1", the view changes to show all SUs in DH1. user can keep drilling down until they reach the last available level/scope

Reports

admin

Auto Refresh | Refresh Interval (sec) 60 | Last Refresh 2025-08-01 16:08:56

Cables Issues | Summary Report | Device Health Report

Data Center > Data Hall: DH1

Displayed Columns | CSV

		Total		Power Supply		Power Supply Fans		Fans		Ports	
Data Hall	SU	Active	Failed	Active	Failed	Active	Failed	Active	Failed	Active	Failed
DH1	SU1	14	4	1	1	1	1	12	0	0	2
DH1	SU2	0	0	0	0	0			0	0	0

Copy Cell

Get Racks in SU1

Viewing 1-2 of 2 | 10

Reports

admin

Auto Refresh | Refresh Interval (sec) 60 | Last Refresh 2025-08-01 16:09:30

Cables Issues | Summary Report | Device Health Report

Data Center > Data Hall: DH1 > Scalable Unit: SU1

Displayed Columns | CSV

			Total		Power Supply		Power Supply Fans		Fans		Ports	
Data Hall	SU	Location	Active	Failed	Active	Failed	Active	Failed	Active	Failed	Active	Failed
DH1	SU1	PXX	14	4	1	1	1	1	12	0	0	2

Copy Cell

Get Nodes in PXX

Viewing 1-1 of 1 | 10

Reports

admin

Auto Refresh | Refresh Interval (sec) 60 | Last Refresh 2025-08-01 16:10:30

Cables Issues | Summary Report | Device Health Report

Data Center > Data Hall: DH1 > Scalable Unit: SU1 > Rack: PXX

Displayed Columns | CSV

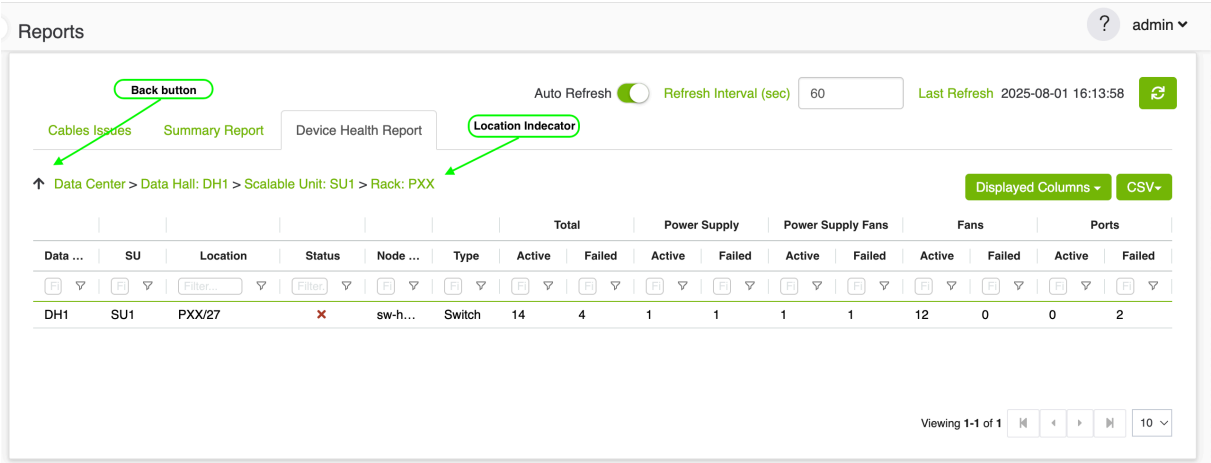
						Total		Power Supply		Power Supply Fans		Fans		Ports	
Data ...	SU	Location	Status	Node Name	Type	Active	Failed	Active	Failed	Active	Failed	Active	Failed	Active	Failed
DH1	SU1	PXX/27	✗	sw-hdr-proton01	Switch	14	4	1	1	1	1	12	0	0	2

Viewing 1-1 of 1 | 10

10.6.3.3 Location Indicator

On the left side of the table, a location indicator shows the current path within the hierarchy. Users can:

- Click on any level in the path to go back to it.
- Use the Back button to return to the previous level.



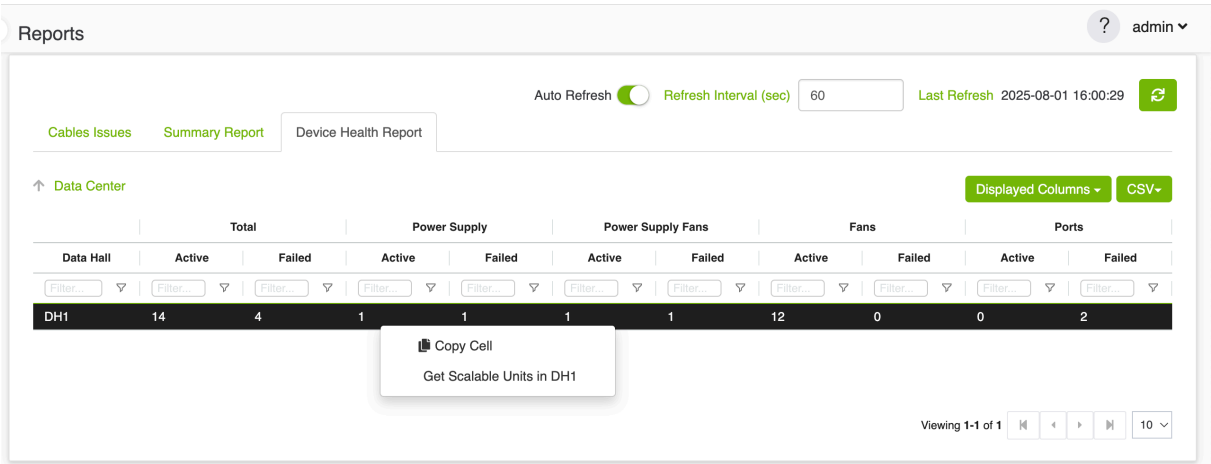
10.6.3.4 Initial Level — Data Center Level (DC)

Not all topologies in CVT include Data Halls (DHs), Scalable Units (SUs), and Racks — it depends on the type of topology the user loads.

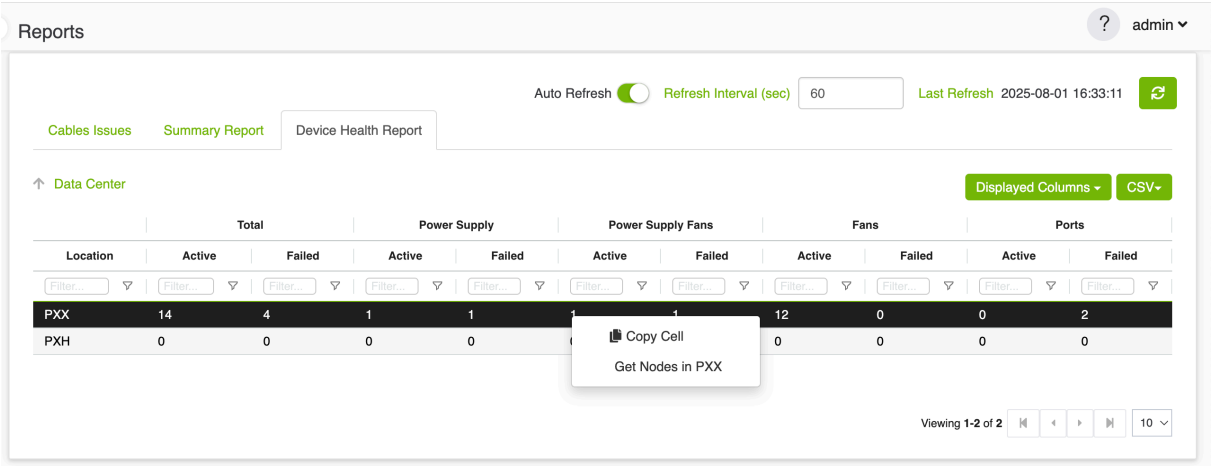
The system initially adjusts the hierarchy based on available data.

For example:

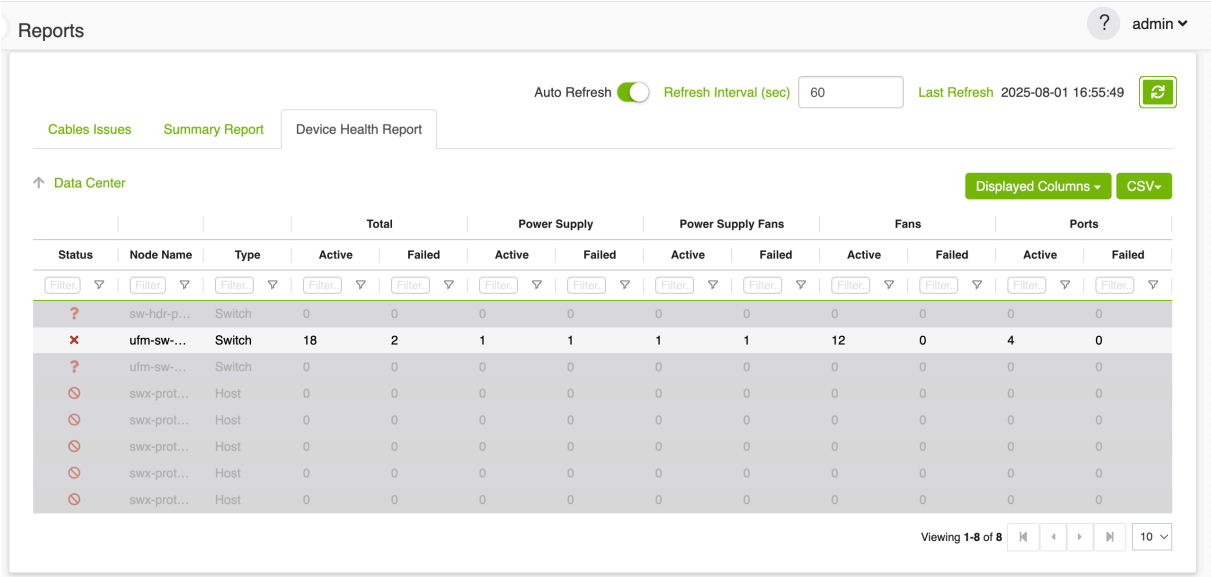
- If the user loads a **P2P file along with a DC floor layout**, the topology will include all levels: DC → DHs → SUs → Racks.



- If the user loads a **P2P file without the DC floor layout**, the topology will only include **DC → Racks**.



- If the user loads a **topo file**, the topology will not support racks at all, so the available level will be only **DC**.



> ⚠ The Status column represents the agent status. For more information, please refer to Agent Status.

10.7 System Admin

The **System Admin** area is the administrative control center for CVT. It is available only to users with the **Admin** role and is reached from the main sidebar under **System Admin**.

10.7.1 Sidebar navigation

System Admin is organized into four top-level sections:

Sidebar item	Purpose
Setup & Validate	End-to-end setup wizard (default landing page)
Agent Management	Agent status, summary reports, and recipe management
Collector Management	Validation/service controls, resource monitoring, collector logs
Users Management	Create, edit, and delete CVT users *(standalone mode only)*

Users Management is hidden when CVT runs embedded in UFM (non-standalone mode).

10.7.2 Section details

10.7.2.1 Setup & Validate

A vertical stepper wizard guides administrators through cluster setup:

1. Load Topology — Upload or select topology and DC layout files
2. Scope — Define validation scope *(visible when* `SCOPE_ENABLED` *is true)*
3. Credentials — Manage node credentials
4. Pre-Check — Run connectivity and readiness checks
5. Deploy Agents — Deploy or remove agents on selected devices
6. Validate — Review summary and start/stop validation

Steps can be navigated freely. A Skip to Validation shortcut is available from any non-final step. Topology metadata (file name, path, last updated) is shown above the stepper. More details are available in Setup & Validate page.

10.7.2.2 Agent Management (Reporting)

Tabbed view with four tabs:

Tab	Description
Summary Report	Device status summary badges and agent summary grid
Agent Status Report	Per-agent status with context-menu actions (deploy, remove, view logs, manage device status)
Recipe Management	Upload, preview, and manage recipe and mapping YAML files
Agent Configuration	Placeholder for future per-agent settings

More details are available in Agent Management page.

10.7.2.3 Collector Management (Service Management)

Tabbed view with four tabs:

Tab	Description
Service Controls	Start/stop validation, stop CVT service, service information
Resource Utilization	CPU, memory, and disk usage cards, charts, and tables
Collector Configuration	Placeholder for future collector-level settings
Collector Logs	Tail and browse collector log files by type

More details are available in Collector Management page.

10.7.2.4 Users Management

A grid-based user table with create, edit, and delete actions via a user modal. Supports single-user creation and bulk upload from file.

More details are available in Users Management page.

10.7.3 Setup & Validate

A vertical stepper wizard that walks administrators through cluster setup before validation. Completed steps are marked with a checkmark; steps can be visited in any order. Skip to Validation jumps directly to the final step from any earlier step.

When a topology is loaded, metadata (file name, server path, last updated time) appears above the stepper, with download links for the topology and DC layout files.

10.7.3.1 Step 1 — Load Topology

Load the fabric topology that defines the devices CVT will manage.

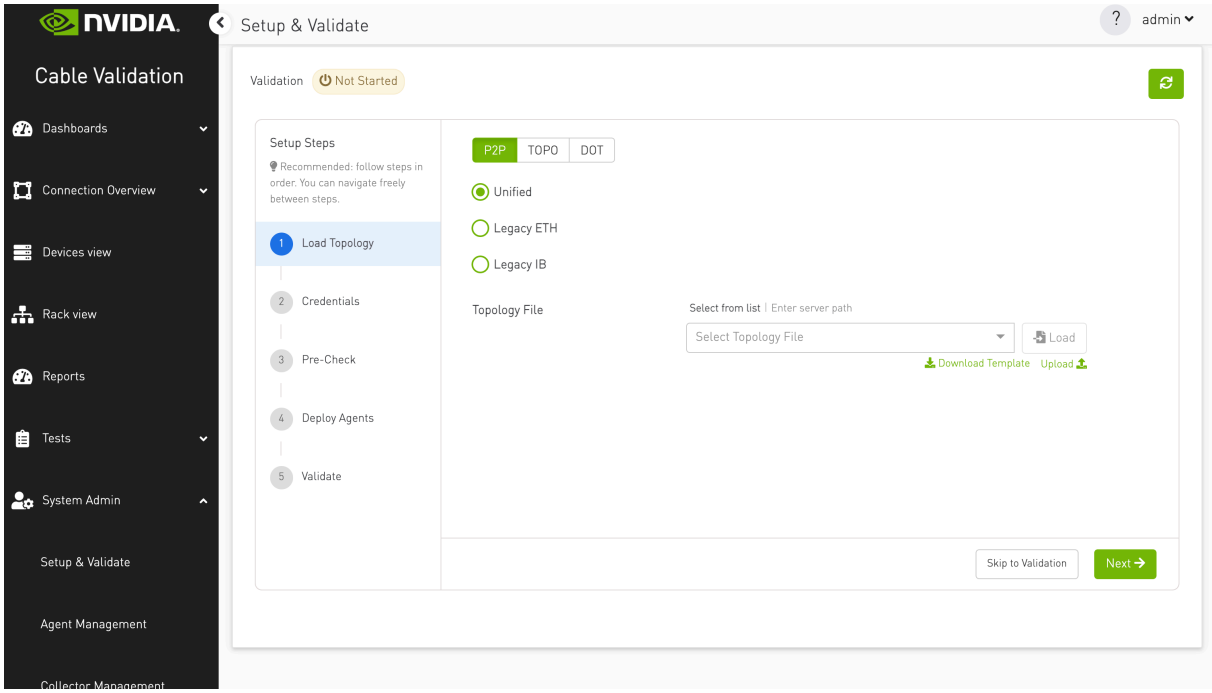
- Load modes: P2P (all fabric types), TOPO (IB fabrics), DOT (ETH fabric), and UFM (when running embedded in UFM).
- P2P formats: Unified (recommended), Legacy ETH, Legacy IB. Legacy formats show a deprecation warning.
- File source: Select from an uploaded file list, or enter an absolute server path.
- Additional files: DC layout file, IP mapping file (where applicable).
- Constraints: Loading a new topology is blocked while validation is running. A warning banner is shown if validation is active.

10.7.3.1.1 P2P File

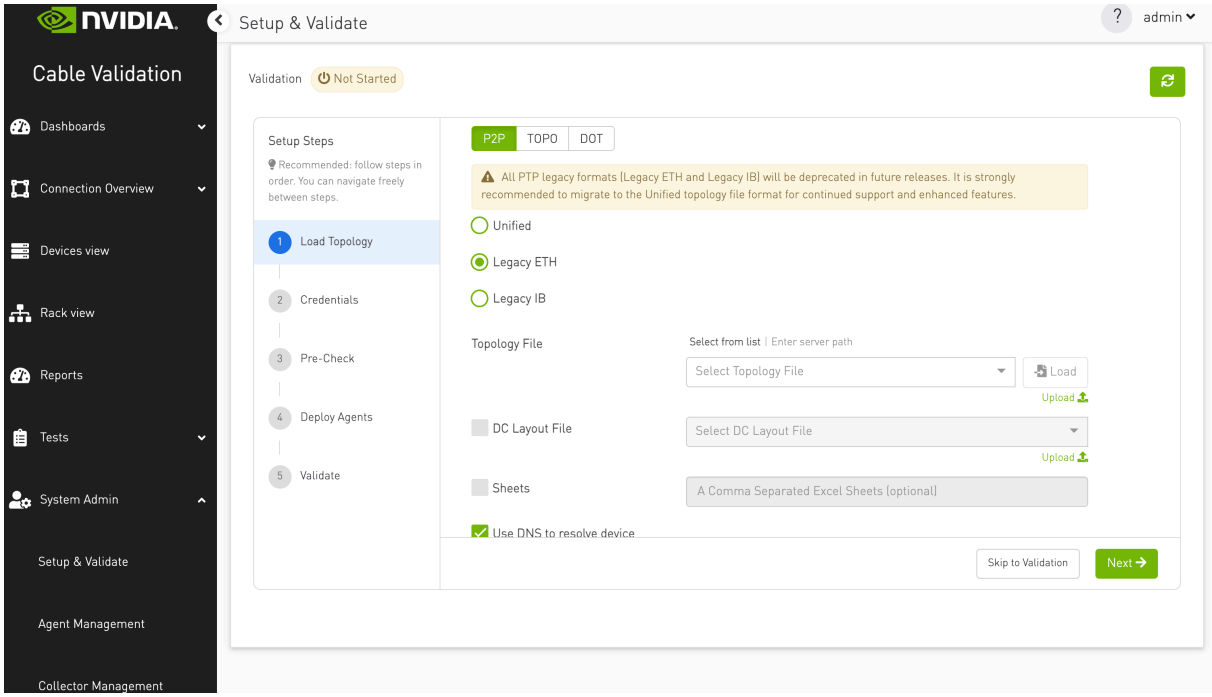
Users must provide the following inputs:

- Format: PTP file format, which can be one of the following:

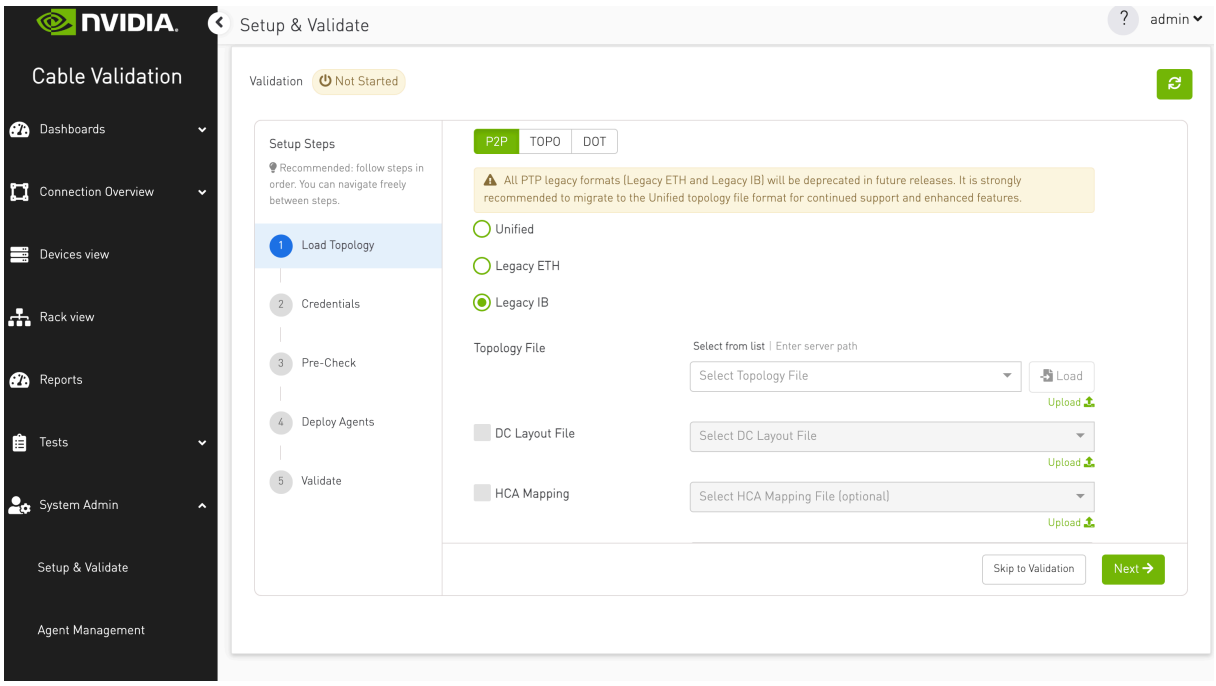
- Unified: It accepts various cluster types, including InfiniBand, Ethernet, and NVLink. For more information, please visit [Unified Topology Format]((2.0.1) P2P File#Unified Topology Format).



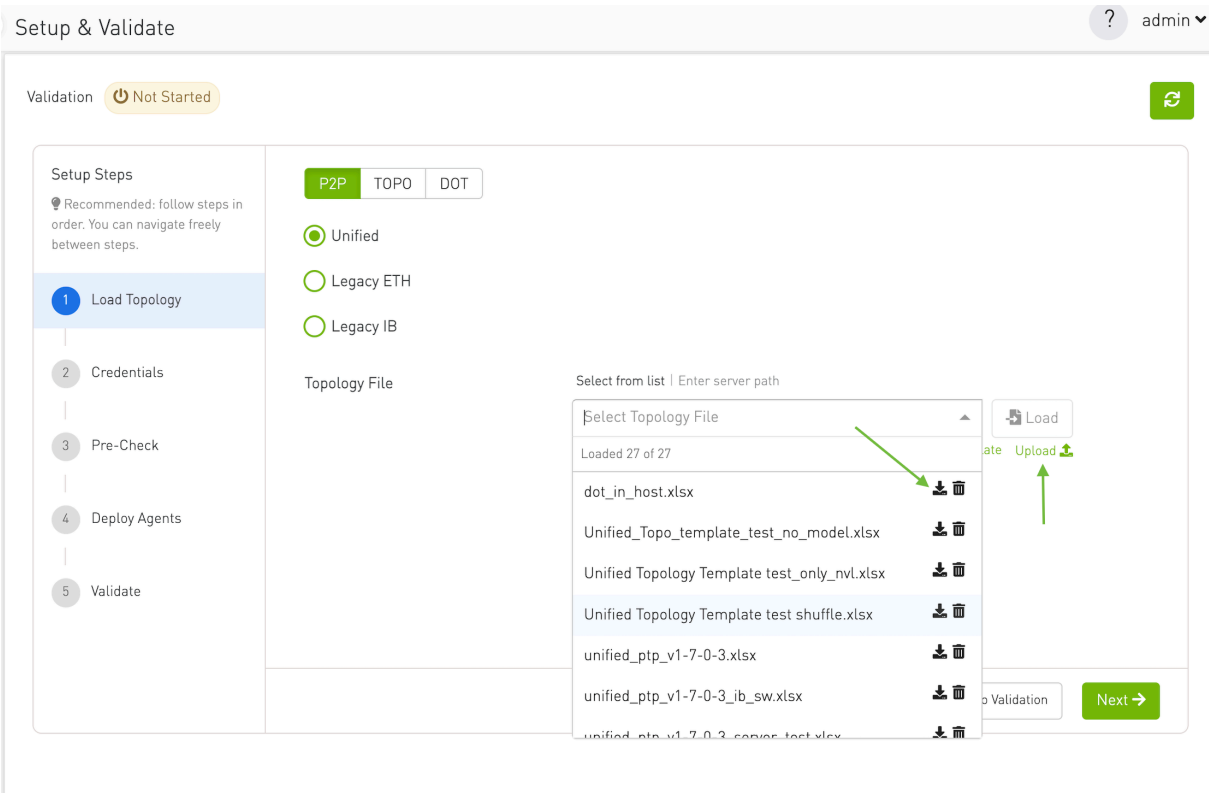
- - Legacy ETH: It accepts only Ethernet fabric. For more information, please visit PTP ETH.



- - Legacy IB: It accepts only InfiniBand fabric. For more information, please visit PTP IB.



- Topology File:
  - Allows users to select or upload a PTP file.
  - Files can be chosen from a dropdown menu listing all .xlsx files in the directory: /opt/bringup_data/data/uploads/topology/ptp/
  - This directory is a shared volume linked to:/cable_bringup_root/data/uploads/topology/ptp/
  - Users can also upload a file directly from their computer using the **Upload** button below the load button.
  - Additionally, users can **download** any file from the dropdown to their local computer.
  - For the **Unified** type, users can download a **template file** that serves as a guide for creating their own unified P2P file.



- DC Layout File (Optional, Appears Only with Legacy formats):
  - Used for selecting or uploading a data center floor layout file.
  - Files can be chosen from a dropdown menu listing all .csv files in the directory: /opt/bringup_data/data/uploads/topology/dc_layout/
  - This directory is a shared volume linked to: /cable_bringup_root/data/uploads/topology/dc_layout/
  - Users can upload a file directly from their computer using the Upload button.
  - Users can enable the DC Layout field by selecting the HCA Mapping checkbox.
- HCA Mapping (Optional, Appears Only with Legacy formats):
  - Allows users to select or upload an HCA mapping file.
  - Files can be chosen from a dropdown menu listing all .csv files in the directory:/opt/bringup_data/data/uploads/topology/hca_mapping/
  - This directory is a shared volume linked to:

    ```
    /cable_bringup_root/data/uploads/topology/hca_mapping/
    ```

  - Users can enable the HCA Mapping field by selecting the HCA Mapping checkbox.
- Sheets (Optional, Appears Only with Legacy formats): The user can specify sheets to be included in the file by enabling the **Sheets** input field (via checkbox) and entering the sheet names as comma-separated values.
- Use DNS to resolve device names (Appears Only with Legacy formats): This option is selected by default. When deselected, an additional input field appears where users can upload an IP file.
  In this mode, the **Load Topology** process runs two commands sequentially — **Load Topology** and **Load IP** — to load both topology and IP data correctly.

- Allow CVT to manage agents on servers (Appears Only with Unified and Legacy ETH formats): This option is deselected by default. If selected, CVT manages the servers included in the topology file; otherwise, it manages only the switches and ignores the servers.

10.7.3.1.2 Topo File

User must provide the following input:

- Topology File:
  - Allows users to select or upload a .topo file.
  - Files can be chosen from a dropdown menu listing all .topo files in the directory:/opt/bringup_data/data/uploads/topology/topo/
  - This directory is a shared volume linked to:/cable_bringup_root/data/uploads/topology/topo/
  - Users can upload a file directly from their computer using the Upload button.
  - Additionally, users can download any file from the dropdown to their local computer.
- Use DNS to resolve device names: This option is selected by default. When deselected, an additional input field appears where users can upload an IP file.
  In this mode, the Load Topology process runs two commands sequentially — Load Topology and Load IP — to load both topology and IP data correctly.

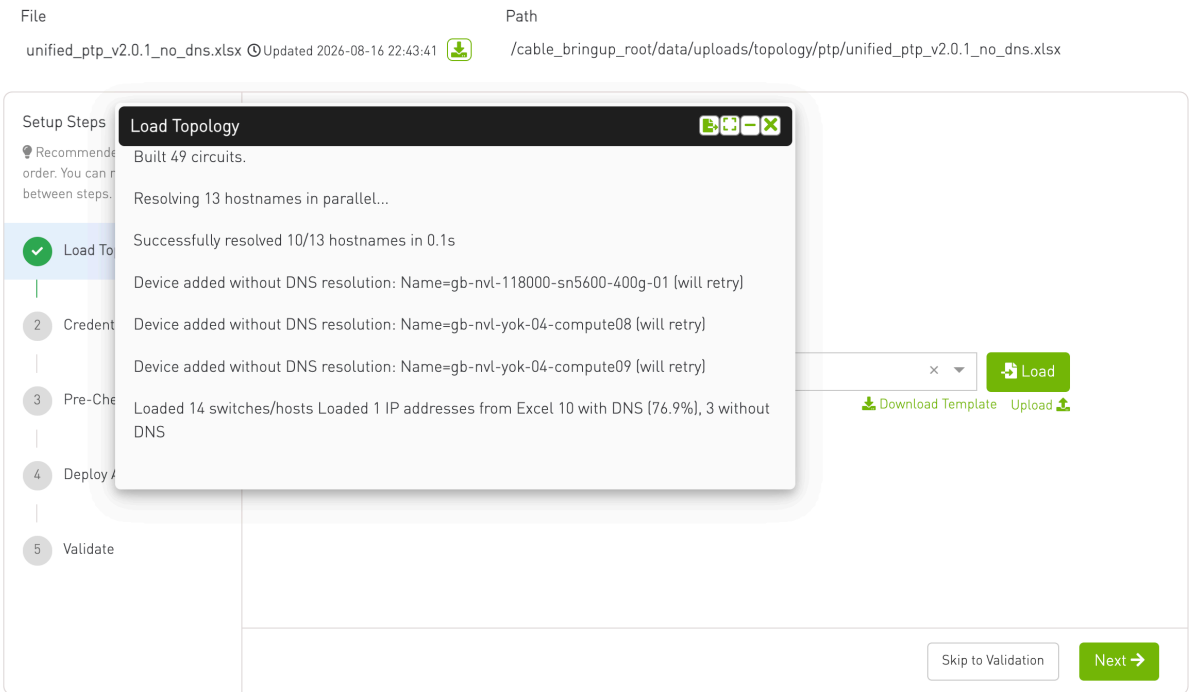
10.7.3.1.3 Dot File

- Topology File:
  - Allows users to select or upload a .dot file.
  - Files can be chosen from a dropdown menu listing all .topo files in the directory: /opt/bringup_data/data/uploads/topology/dot/
  - This directory is a shared volume linked to :/cable_bringup_root/data/uploads/topology/dot/
  - Users can upload a file directly from their computer using the Upload button.
  - Additionally, users can download any file from the dropdown to their local computer.
- Use DNS to resolve device names: This option is selected by default. When deselected, an additional input field appears where users can upload an IP file.
  In this mode, the Load Topology process runs two commands sequentially — Load Topology and Load IP — to load both topology and IP data correctly.
- Allow CVT to manage agents on servers: This option is deselected by default. If selected, CVT manages the servers included in the topology file; otherwise, it manages only the switches and ignores the servers.

10.7.3.2 Topology Summary

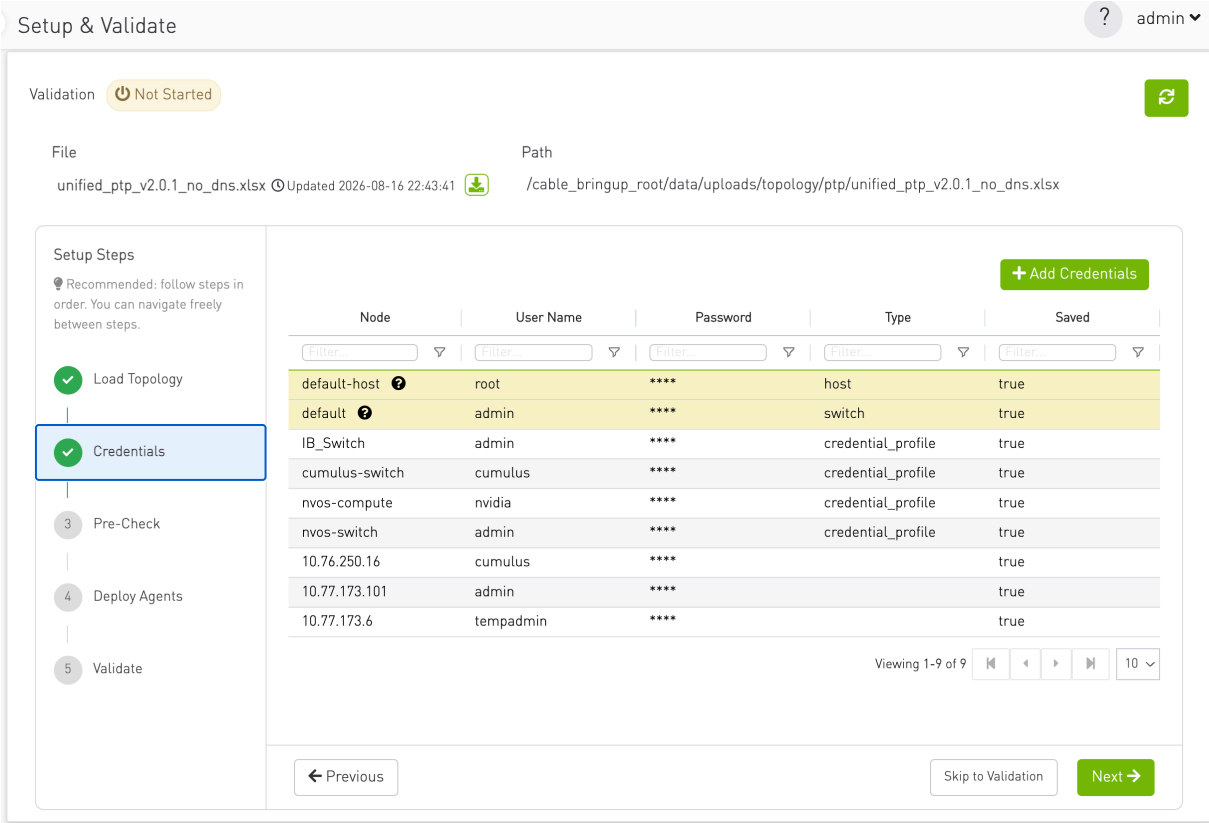
After clicking on Load, the topology file is loaded and this step provides a summary of the loaded topology.
Users can review the details to confirm that the correct file and settings have been applied.

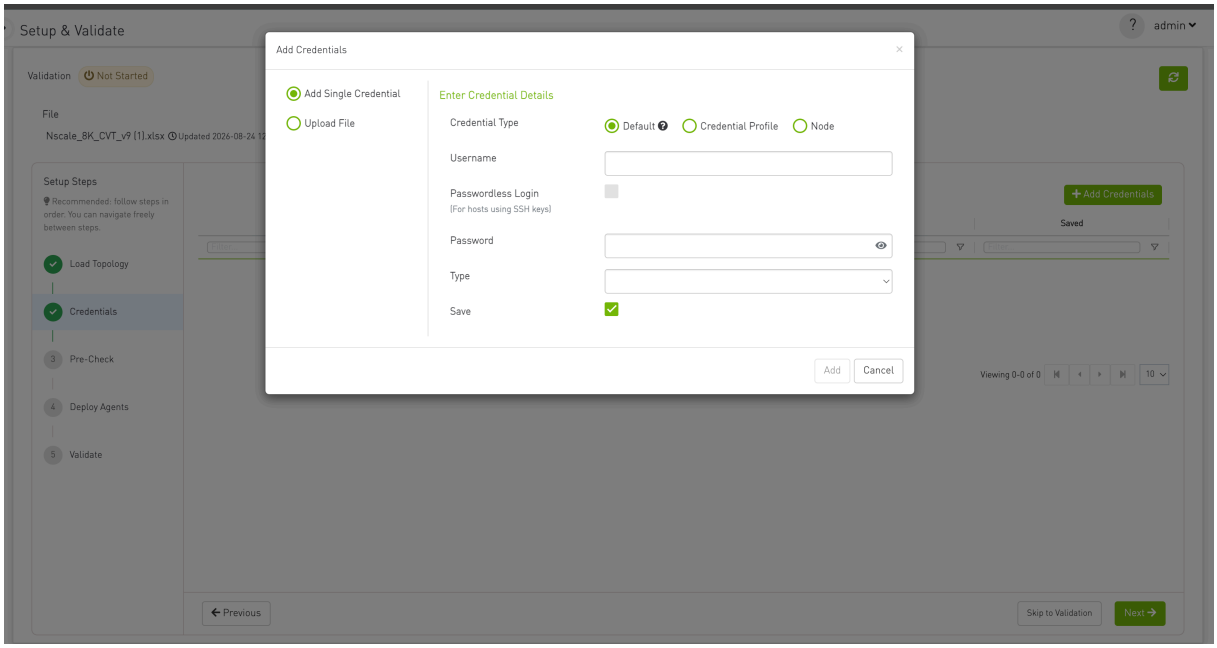
The topology summary can be exported as a text file, allowing users to save, share, or reference the information later.

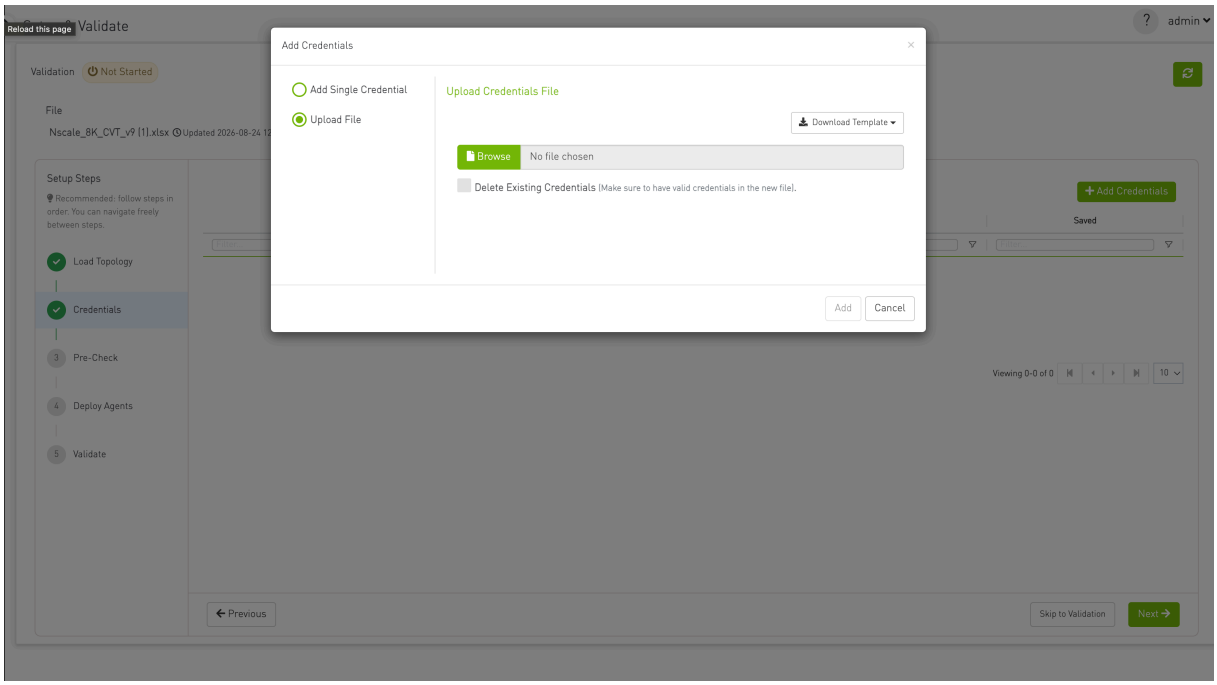


10.7.3.3 Step 2 — Credentials

Manage node credentials required for agent deployment and device access. Uses the shared nodes-credentials table component. Credentials must be configured before pre-check and deploy steps can succeed.


Setup & Validate
admin
Validation
Not Started
File
Path
unified_ptp_v2.0.1_no_dns.xlsx
Updated 2026-08-16 22:43:41
/cable_bringup_root/data/uploads/topology/ptp/unified_ptp_v2.0.1_no_dns.xlsx
Setup Steps
Recommended: follow steps in order. You can navigate freely between steps.
Load Topology
Credentials
3 Pre-Check
4 Deploy Agents
5 Validate
Add Credentials
Node
User Name
Password
Type
Saved
default-host root **** host true
default admin **** switch true
IB_Switch admin **** credential_profile true
cumulus-switch cumulus **** credential_profile true
nvos-compute nvidia **** credential_profile true
nvos-switch admin **** credential_profile true
10.76.250.16 cumulus **** true
10.77.173.101 admin **** true
10.77.173.6 tempadmin **** true
Viewing 1-9 of 9
10
Previous
Skip to Validation
Next


Setup & Validate
admin
Add Credentials
Add Single Credential
Upload File
Enter Credential Details
Credential Type
Default
Credential Profile
Node
Username
Passwordless Login
(For hosts using SSH keys)
Password
Type
Save
Add
Cancel

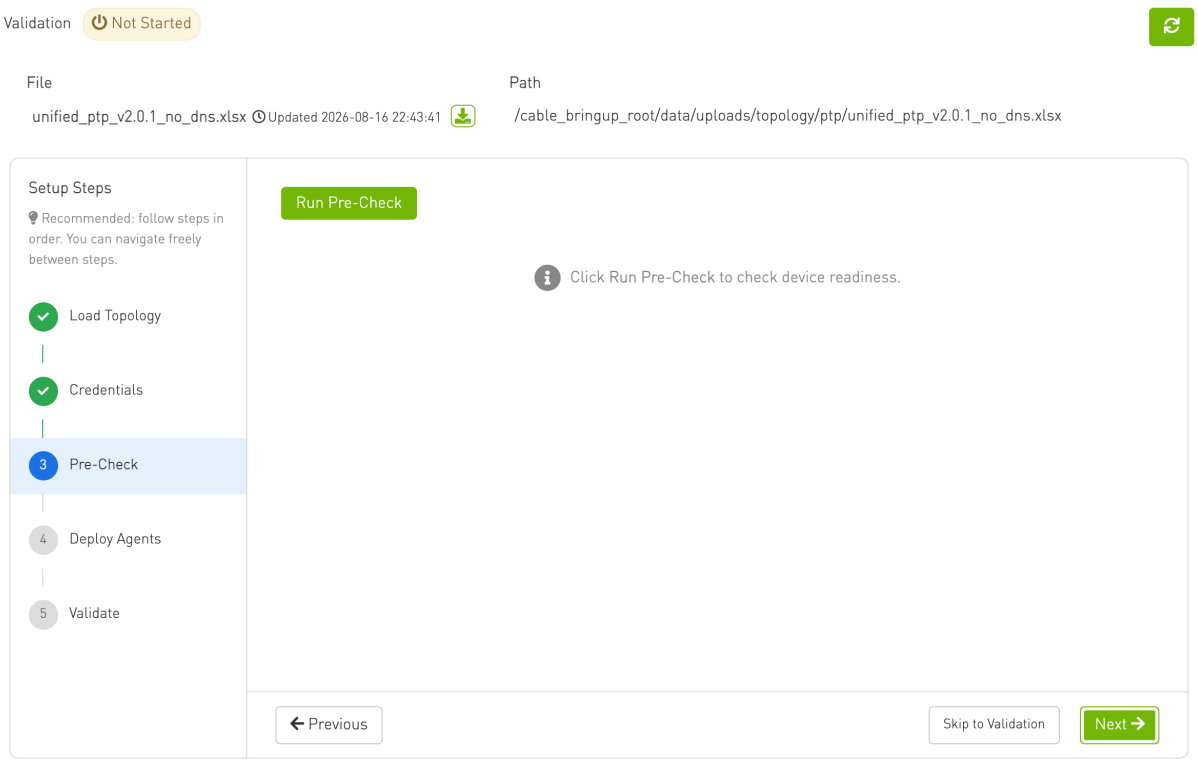


For more information, please visit Device Access Credentials

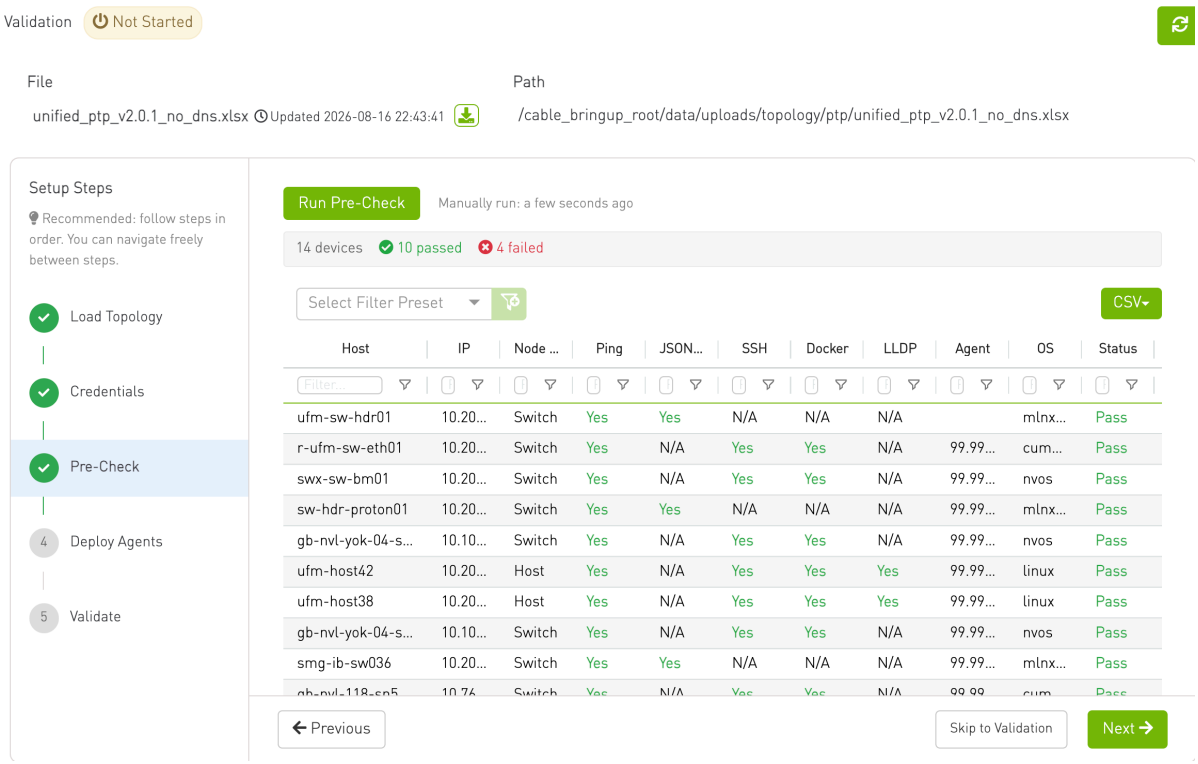
10.7.3.4 Step 3 — Pre-Check

Run readiness checks across all devices.

- **Checks performed:** Ping, JSON API, SSH, Docker, LLDP, and agent status.
- **Run Pre-Check** starts the check; results poll until complete. A last-run timestamp indicates result freshness.
- **Results:** Summary bar (total / passed / failed) and a per-device results grid with pass/fail per check.



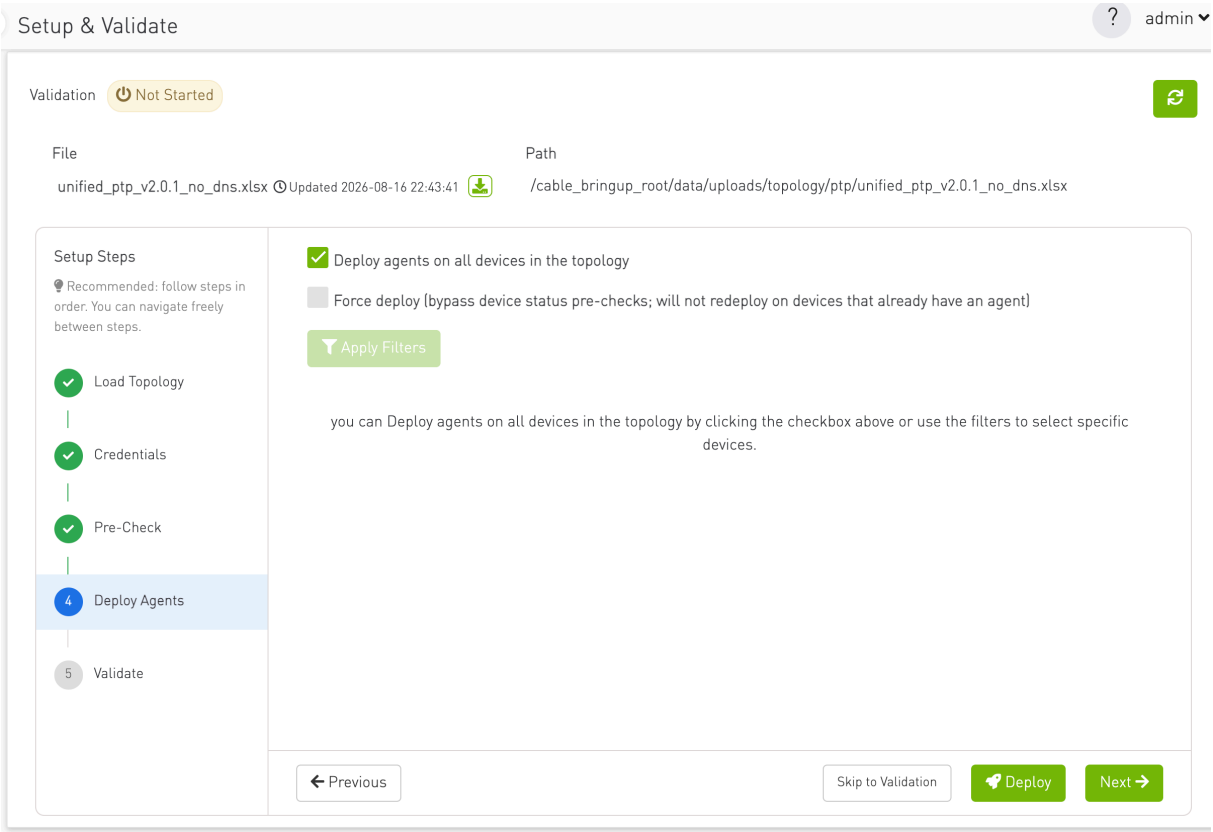
Pre-check results:



10.7.3.5 Step 4 — Deploy Agents

Deploy or remove CVT agents on selected devices.

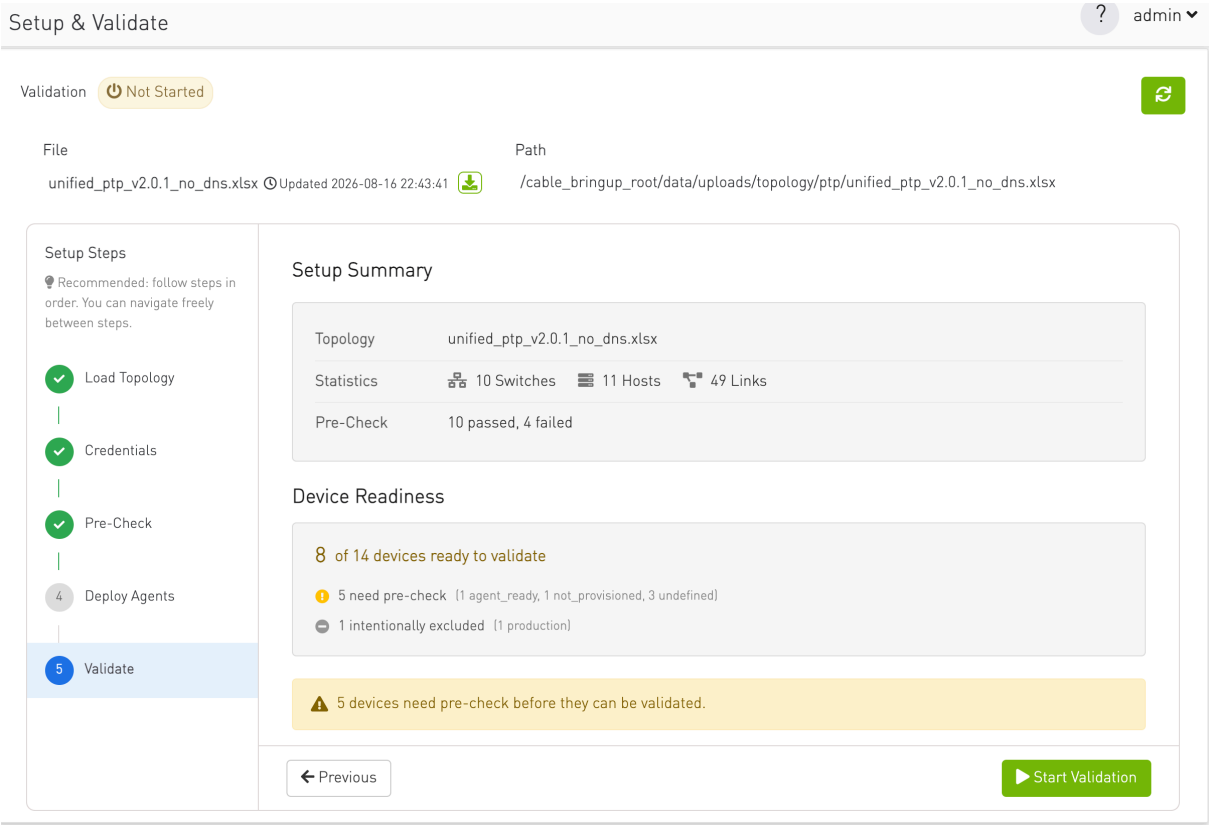
- Shows the deploy-agents component with scope context (mode, workspace, deployment, scoped device count).
- **Deploy** button in the step navigation runs deployment on the current selection.
- Deploy is disabled while validation is running or while a deploy is in progress.
- Agent command output is shown via the standard command polling window.



10.7.3.6 Step 5 — Validate

Review setup readiness and start or stop validation.

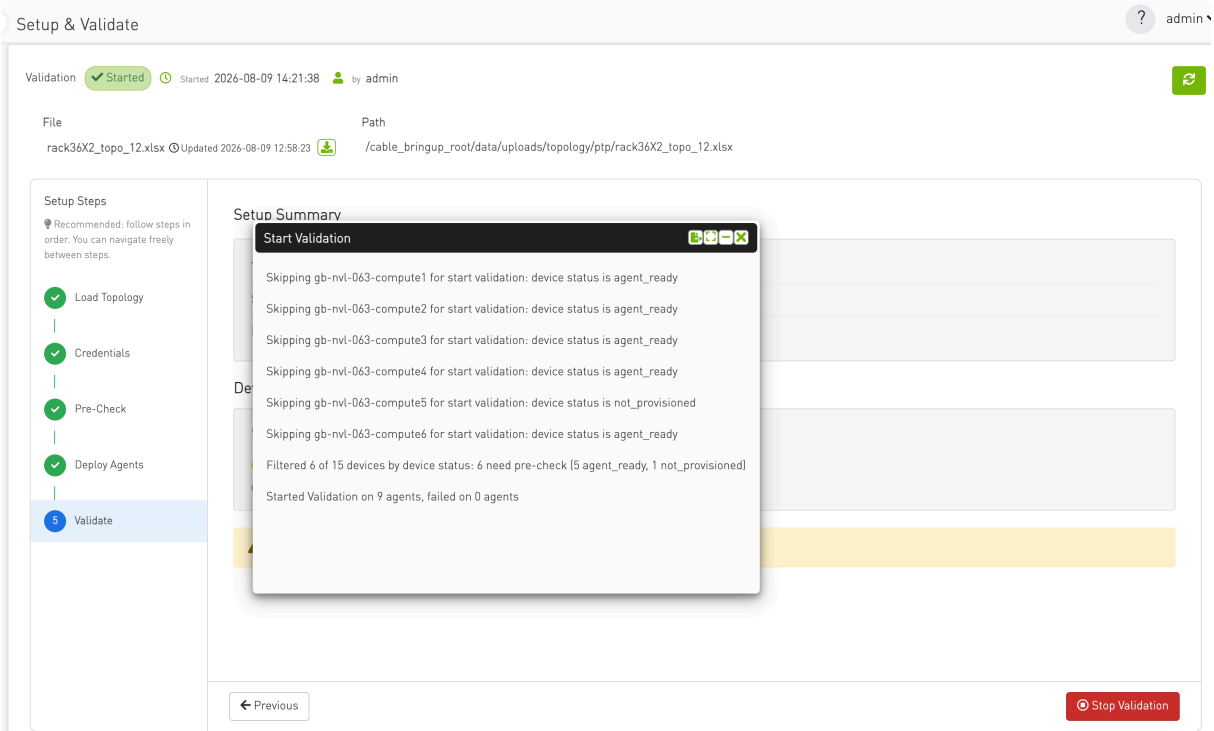
- **Setup Summary:** Topology file name, switch/host/link counts, scope label (if enabled), and pre-check summary.
- **Device Readiness:** Count of devices ready to validate vs. total, with breakdowns for devices needing pre-check or intentionally excluded.
- **Alerts:** Warning or error banners when readiness is incomplete.
- **Start Validation / Stop Validation** buttons in the step navigation (same behavior as Service Controls).



When the user clicks **Start Validation**, a pop-up window appears, providing real-time visibility into the validation progress and results. Users can minimize this window and continue navigating within the application while the validation runs in the background.

The validation output can be exported as a text file for later review, troubleshooting, or documentation purposes.

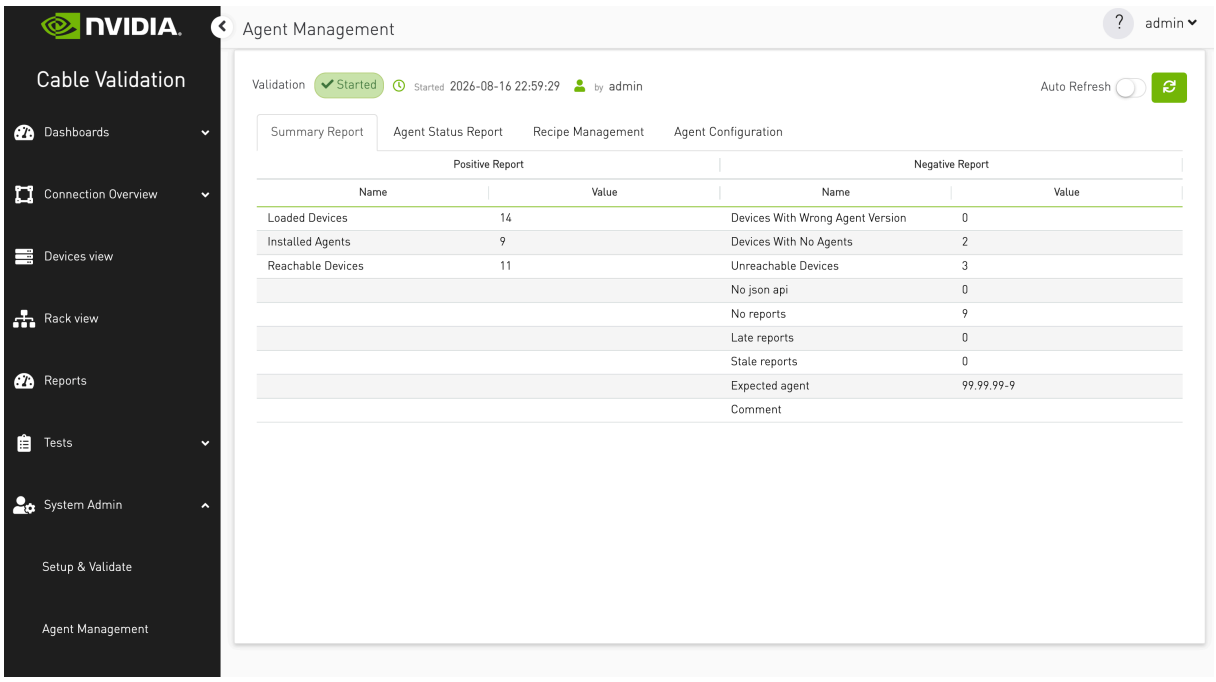
During the validation process, the **Close** button is disabled to prevent accidental closure until the validation results are ready for review.



10.7.4 Agent Management

10.7.4.1 Summary Report

The Summary Report View offers a comprehensive overview of the loaded topology, providing information about the state of devices and their associated agents.

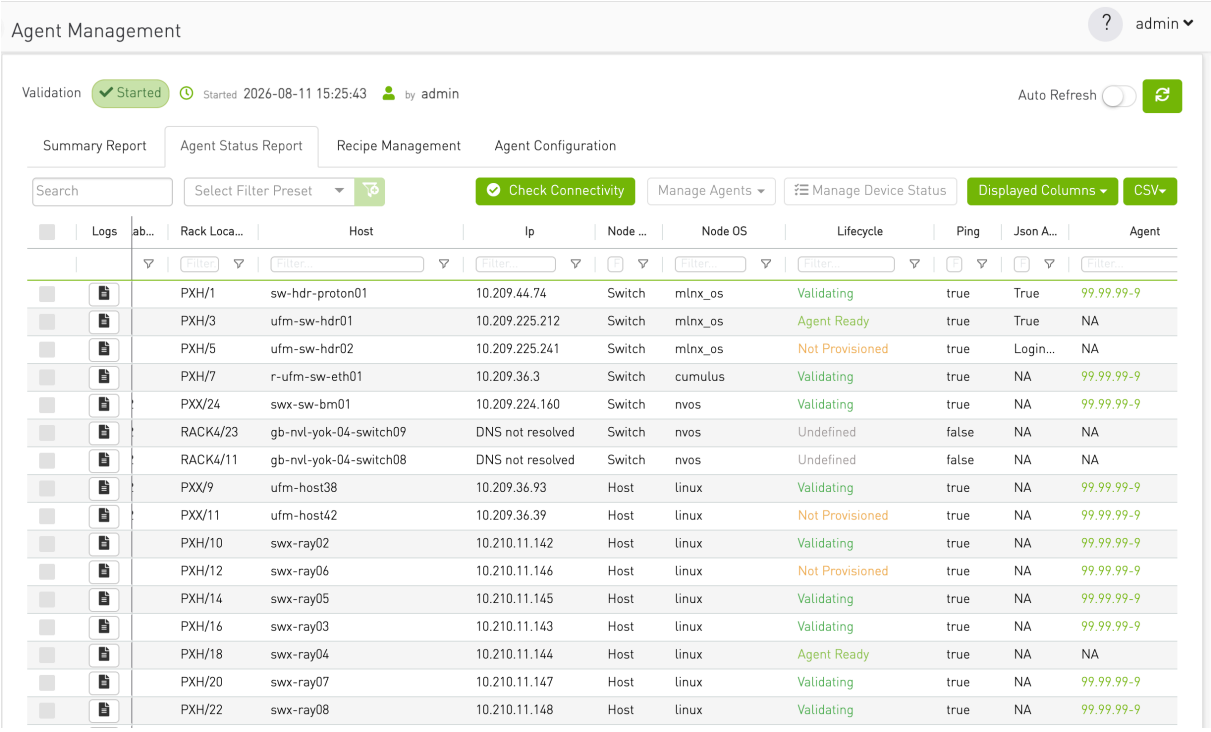


The report includes the following key metrics:

- **Loaded Devices** – The total number of devices loaded in the topology.
- **Installed Agents** – The number of devices with agents successfully installed.
- **Reachable Devices** – Devices that are accessible and responsive.
- **Unreachable Devices** – Devices that could not be reached during validation.
- **Devices with Wrong Agent Version** – Devices running an agent version different from the expected one.
- **Devices with No Agents** – Devices that do not have any agents installed.
- **No JSON API** – Devices that do not expose the required JSON API.
- **No Reports** – Devices that have not submitted any reports.
- **Late Reports** – Devices whose reports were received after the expected time.
- **Stale Reports** – Devices that have not sent updated reports within the last 15 minutes.

10.7.4.2 Agent Status Report

The Agent Status Report provides detailed information about the agents installed on each node within the topology.
The report includes key details such as the node's **location** (rack and unit), **type**, **name**, and **IP address**.



Logs	ab...	Rack Loca...	Host	Ip	Node ...	Node OS	Lifecycle	Ping	Json A...	Agent
		PXH/1	sw-hdr-proton01	10.209.44.74	Switch	mlnx_os	Validating	true	True	99.99.99-9
		PXH/3	ufm-sw-hdr01	10.209.225.212	Switch	mlnx_os	Agent Ready	true	True	NA
		PXH/5	ufm-sw-hdr02	10.209.225.241	Switch	mlnx_os	Not Provisioned	true	Login...	NA
		PXH/7	r-ufm-sw-eth01	10.209.36.3	Switch	cumulus	Validating	true	NA	99.99.99-9
		PXX/24	swx-sw-bm01	10.209.224.160	Switch	nvos	Validating	true	NA	99.99.99-9
		RACK4/23	gb-nvl-yok-04-switch09	DNS not resolved	Switch	nvos	Undefined	false	NA	NA
		RACK4/11	gb-nvl-yok-04-switch08	DNS not resolved	Switch	nvos	Undefined	false	NA	NA
		PXX/9	ufm-host38	10.209.36.93	Host	linux	Validating	true	NA	99.99.99-9
		PXX/11	ufm-host42	10.209.36.39	Host	linux	Not Provisioned	true	NA	99.99.99-9
		PXH/10	swx-ray02	10.210.11.142	Host	linux	Validating	true	NA	99.99.99-9
		PXH/12	swx-ray06	10.210.11.146	Host	linux	Not Provisioned	true	NA	99.99.99-9
		PXH/14	swx-ray05	10.210.11.145	Host	linux	Validating	true	NA	99.99.99-9
		PXH/16	swx-ray03	10.210.11.143	Host	linux	Validating	true	NA	99.99.99-9
		PXH/18	swx-ray04	10.210.11.144	Host	linux	Agent Ready	true	NA	NA
		PXH/20	swx-ray07	10.210.11.147	Host	linux	Validating	true	NA	99.99.99-9
		PXH/22	swx-ray08	10.210.11.148	Host	linux	Validating	true	NA	99.99.99-9

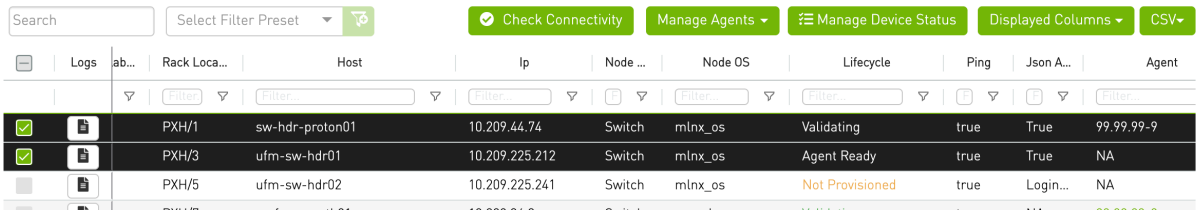
To simplify validation, agent versions are color-coded in the UI:

- **Green** → Version matches the bring-up version.
- **Orange** → Version does not match the bring-up version.
- **Black** → Not available (N/A).

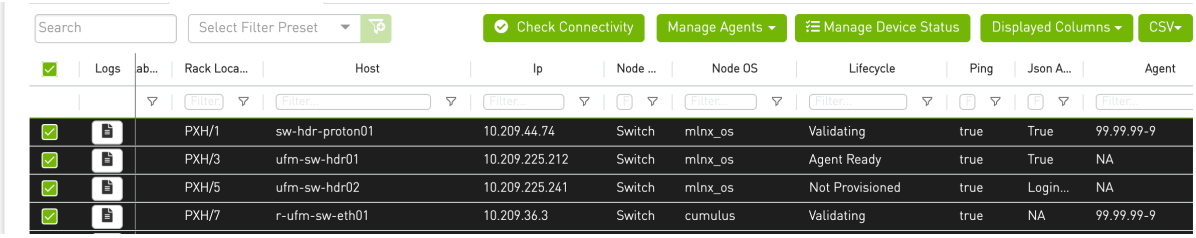
Agent
1.7.0-5
1.7.0-5
1.7.1-2
1.7.1-2
1.7.1-2
99.99.99-9
99.99.99-9
NA

10.7.4.3 Selection-based management

Administrators can select one or more rows and manage agents and device lifecycle status directly from the report. The toolbar actions remain disabled until at least one row is selected.



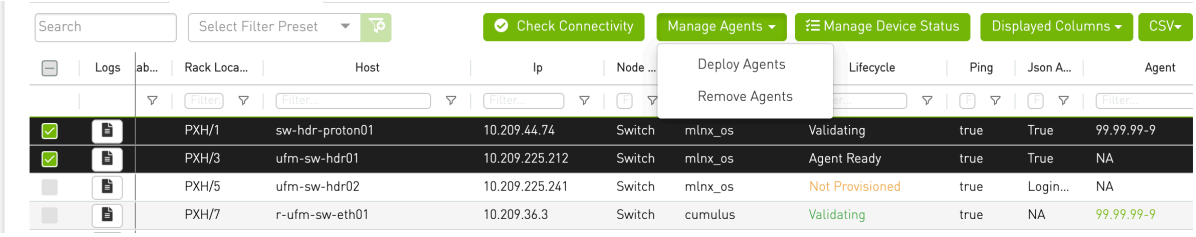
Use the header checkbox to select all rows when an operation should apply to all agents.



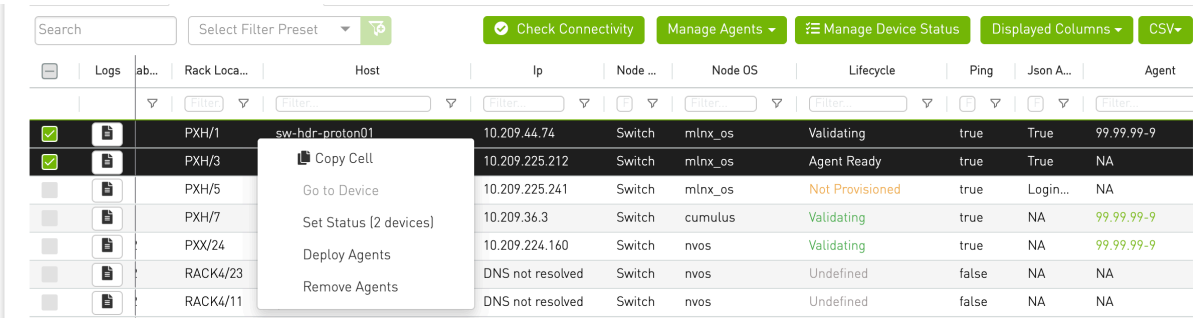
Selections are preserved when the table reloads, so the same devices remain selected after data refreshes.

10.7.4.3.1 Manage Agents

Select the required rows, then open Manage Agents and choose Deploy Agents or Remove Agents.



The same actions are also available from the right-click context menu.

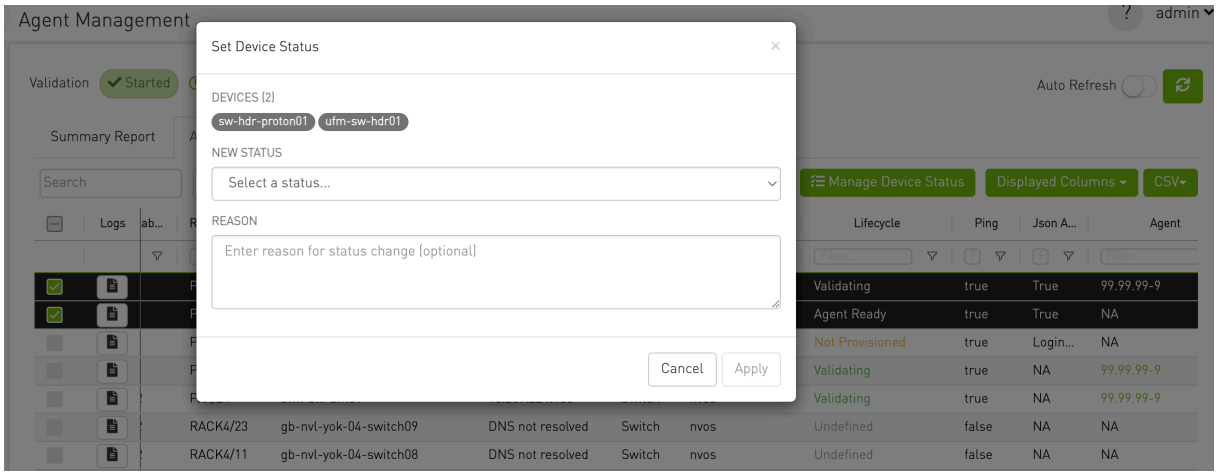


Action availability depends on the selected rows:

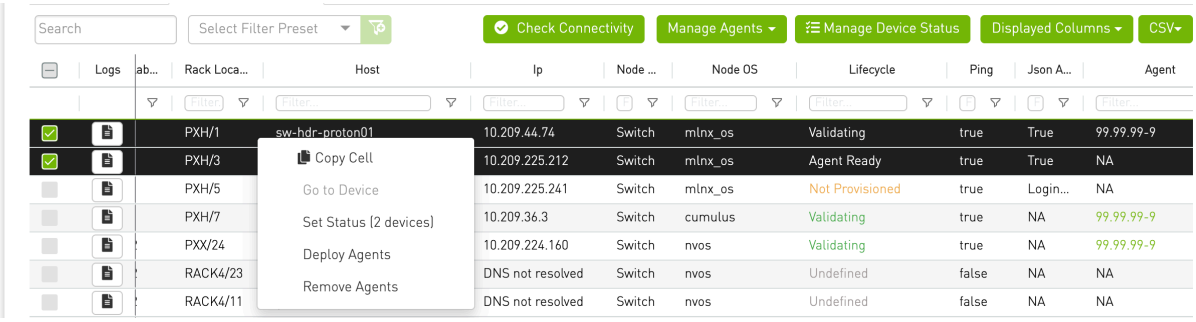
- **Deploy Agents** is enabled when every selected row has an IP address and at least one selected device does not have the bring-up agent version.
- **Remove Agents** is enabled when every selected row has an IP address and at least one selected device has an agent installed.
- For a mixed selection, both actions can be available.
- To deploy or remove all agents, select all rows with the header checkbox and then use **Manage Agents**.

10.7.4.3.2 Manage Device Status

Select one or more rows and click **Manage Device Status** to update their lifecycle status from the shared **Set Device Status** modal. The report reloads after the status is changed.



The right-click context menu also provides **Set Device Status** for one selected device or **Set Status (N devices)** for multiple selected devices. Device-status management is available only when every selected row has a device hostname.



10.7.4.3.3 Context menu

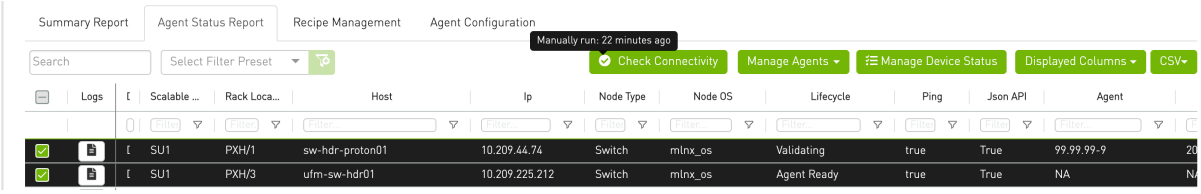
Right-click the selected rows to access the actions that apply to the current selection:

- **Go to Device** – Available only when one device with a hostname is selected.
- **Set Device Status / Set Status (N devices)** – Updates lifecycle status for the selected devices.
- **Deploy Agents** – Deploys agents to the selected devices when applicable.
- **Remove Agents** – Removes agents from the selected devices when applicable.

Unavailable actions are disabled.

10.7.4.3.4 Connectivity last-run indicator

The **Check Connectivity** control displays the last manual run status and provides the ability to run it.



10.7.4.3.5 Real-time deployment monitoring

After deploying or removing agents, a monitoring window is displayed.
This window shows real-time output of the ongoing operation, allowing administrators to track deployment progress and verify success or failure.

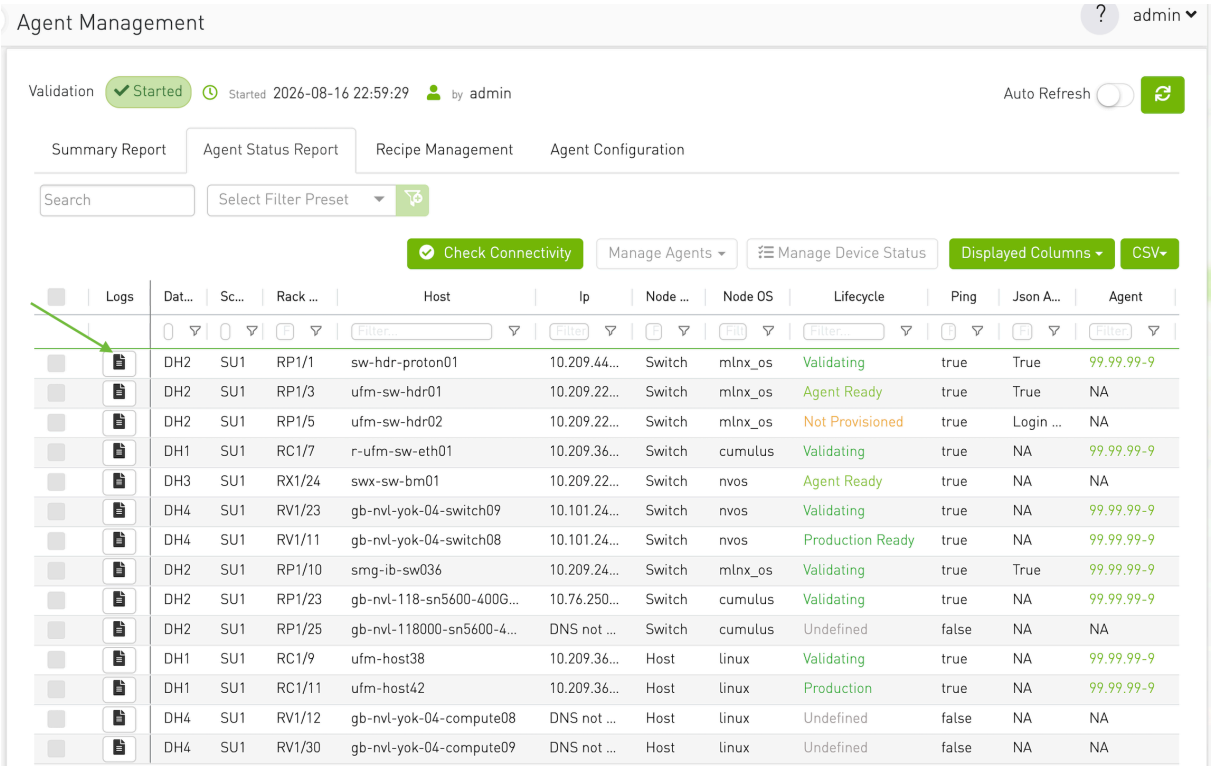
The monitoring window is resizable, enabling users to minimize it and continue navigating through the application while the operation runs in the background.

Users can also export the displayed output to a text file for later review, troubleshooting, or record-keeping.

Agent logs:

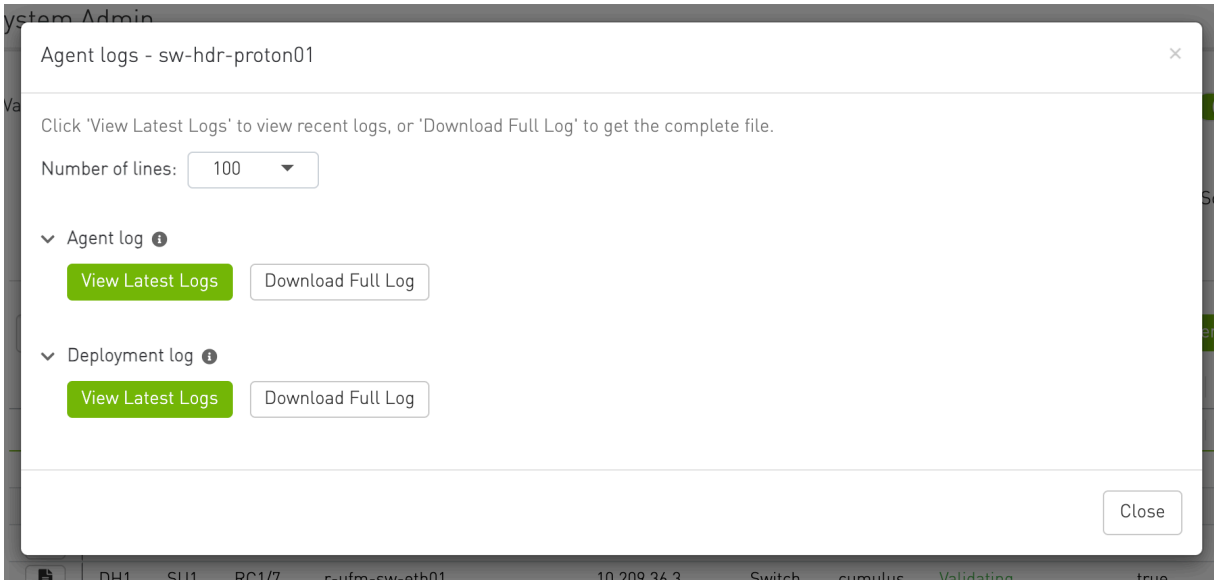
In the Agent status report page, the table contains a row per node managed on CVT. The first column in the table contains the link to the logs (click on corresponding log icon of the node to see its logs).

- Clicking on the log icon on a row opens a modal with options to view logs.

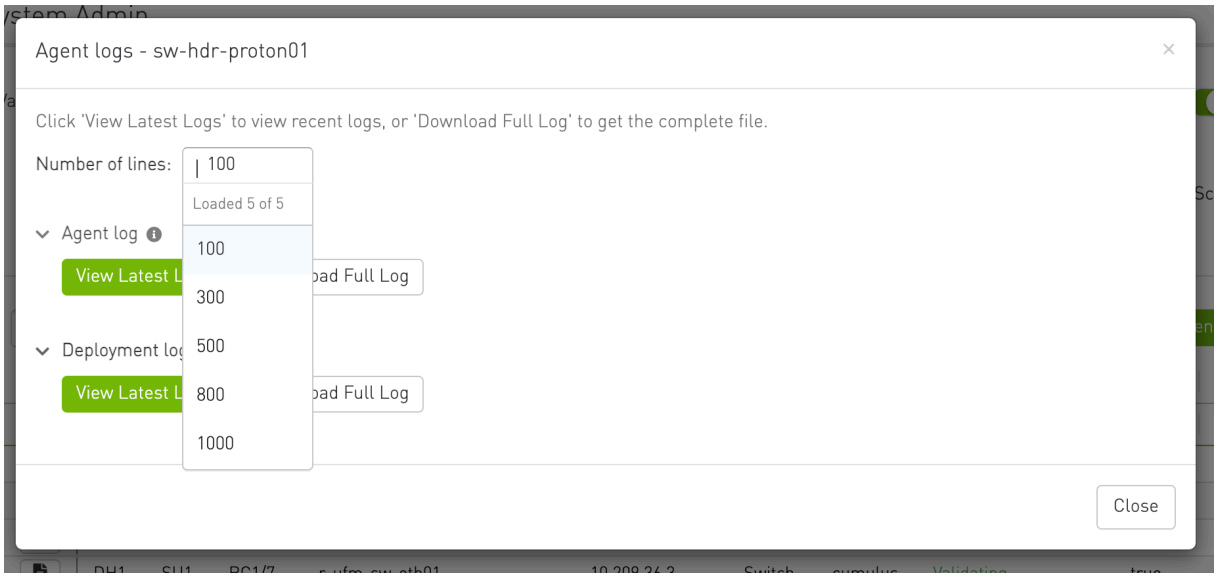


- Number of lines — same choices as collector (100, 300, 500, 800, 1000). Changing the value reloads any section that already has loaded content.
- Two collapsible sections:
    - Agent log — runtime / validation output (Buttons to **View Latest Logs** and **Download full Log**).
    - Deployment log — deploy/remove operations (Buttons to **View Latest Logs** and **Download full Log**).
- View Latest Logs — loads the tail for that section (content does not load until you click this).
- Download Full Log / Export Displayed Logs — per section, same idea as on the collector logs screen.
- Export Displayed Logs — saves the text currently in the panel (the tail), not the full file. This option is available once **View Latest Logs** button is clicked.

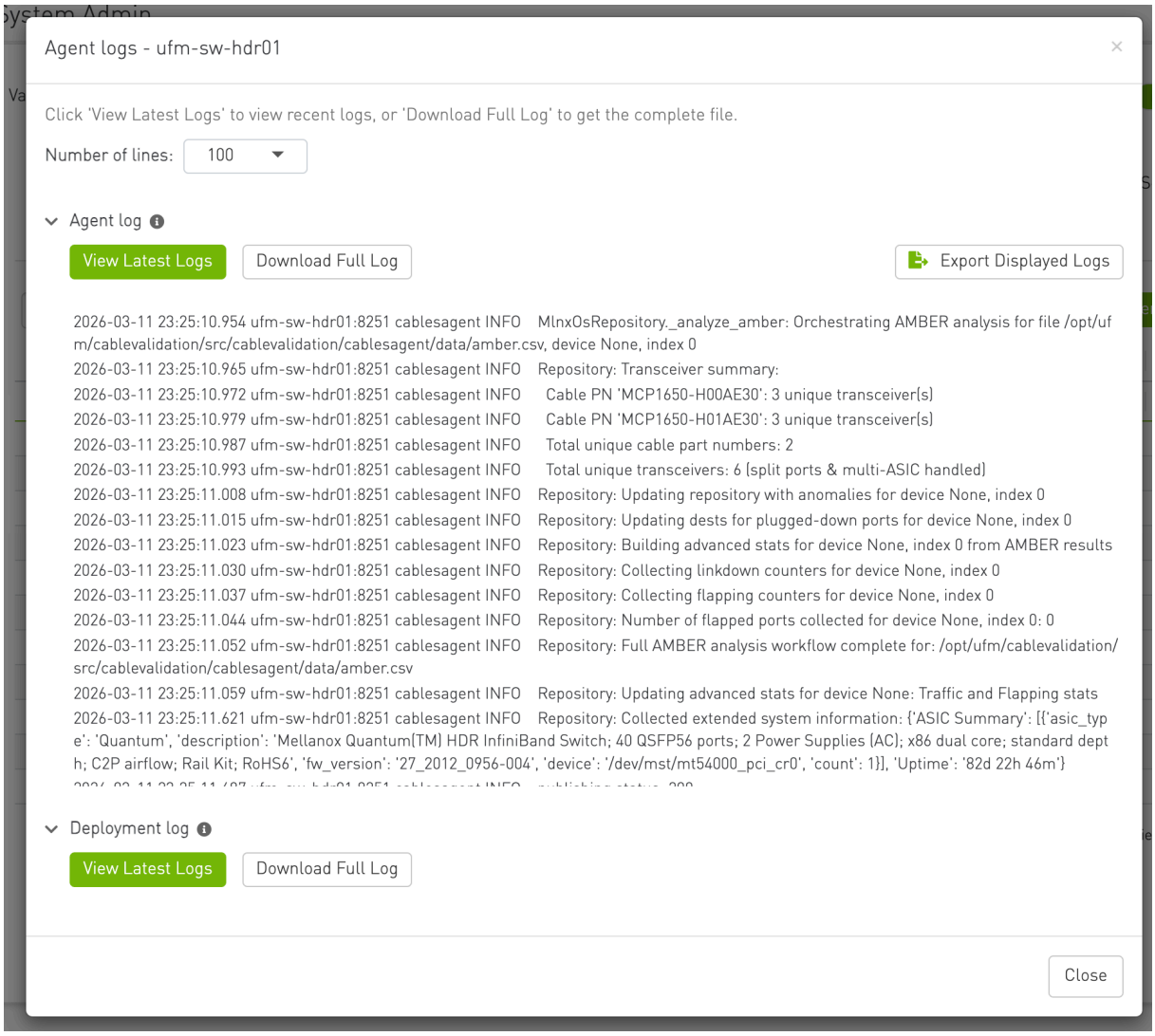
Agent Logs Modal opens:



Select number of latest log lines to display:



Click buttons to view latest logs, or download full log file or export the latest logs loaded on the modal:



10.7.4.4 Recipe Management

Recipe validation lets CVT operators confirm that devices in a fabric are running the expected software and firmware versions defined in YAML recipe files. CVT compares actual versions reported by agents against expected versions in the recipe, and surfaces pass/fail status at the device and rack level.

10.7.4.4.1 How it works

1. Recipe files — YAML files listing expected software and firmware versions (e.g. switch image, ASIC firmware).
2. Mapping file — A YAML mapping assigns recipes to devices using rules (by data hall, scalable unit, rack, hostname pattern, role, etc.) or explicit per-device overrides. Resolution order is: override → rack → SU → DH → pattern → role → default.
3. Validation — When agent data is available, the collector compares each version field in the recipe to what the device reports and marks each field Pass or Fail. A device-level status is derived from those results.

10.7.4.4.2 Where it appears in the UI

Location	Purpose
System Admin → Agent Management → Recipe Management (described in this page)	Upload/manage recipe and mapping YAML files
Devices View → Device Recipe tab	See assigned recipe, mapping rule, and field-level results; assign or override per device
Devices View grid	Recipe Compliance column for fleet-wide scanning
Rack View	Per-rack compliance summary and per-unit recipe badges

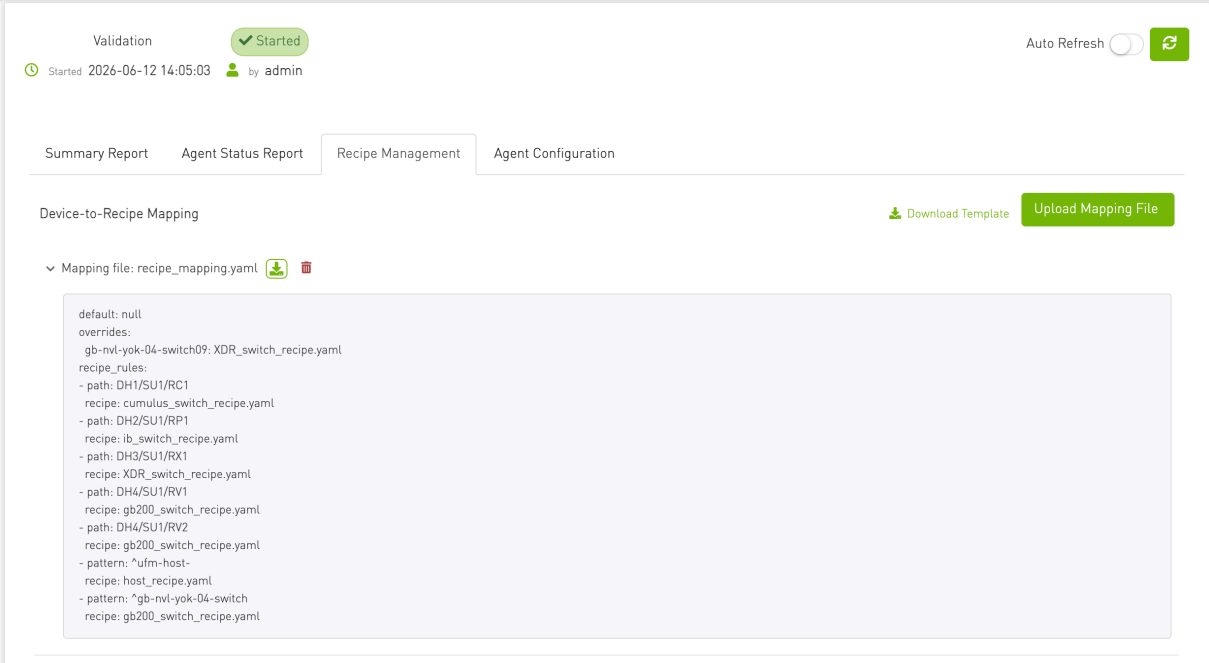
Upload and manage recipe validation files used across the cluster.

- Device-to-Recipe Mapping: Upload a single mapping YAML file. Download template, preview content inline, download, or delete.
- Recipe Files: Upload one or more recipe YAML files. Select a recipe from a dropdown to preview, download, or delete.
- Overwrite confirmation: Prompted when uploading a file that already exists.

Recipe files drive compliance reporting in Devices View and Rack View.

The view has two stacked sections:

10.7.4.5 Device-to-Recipe Mapping Section



Element	Description
Section title	"Device-to-Recipe Mapping"

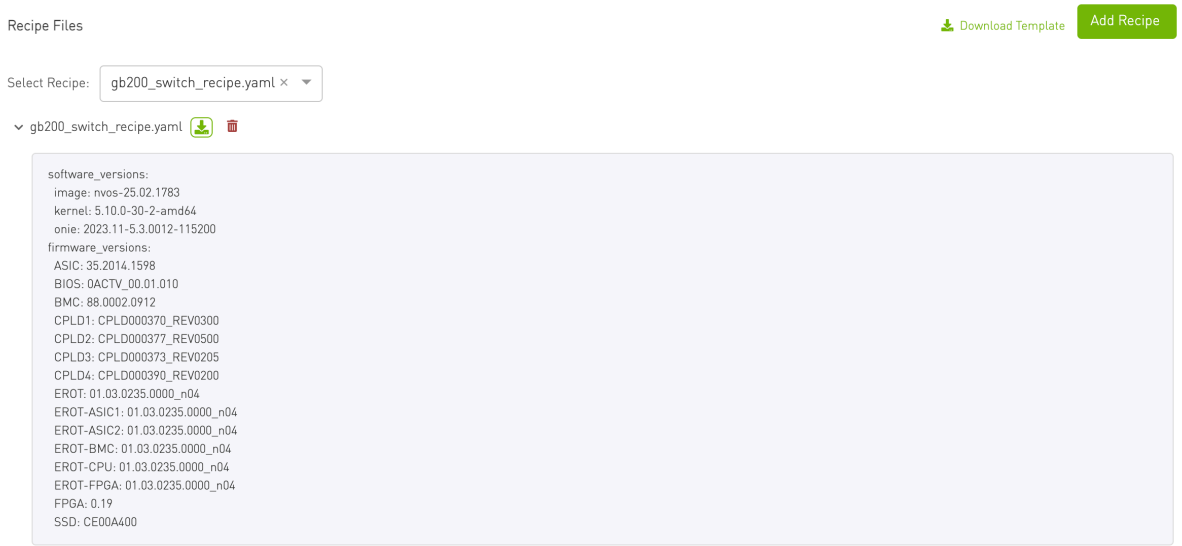
Element	Description
Download Template	Downloads bundled `recipe_mapping_template.yaml` from collector
Upload	`app-x-core-file-uploader` — accepts `.yaml` / `.yml`; posts to mapping folder
Mapping file row	Shows first mapping file in folder (single-file model); collapsible preview
Preview	Raw YAML in `<pre>` block
Download	Downloads current mapping file
Delete	Removes mapping file; triggers mapping reload and targeted revalidation
Empty state	"No mapping file uploaded"
Overwrite confirm	Modal when uploading a file that already exists

Behavior on upload/delete:

- Backend reloads mapping, re-resolves all topology nodes, diffs old vs new per-device assignments, and revalidates only **affected** devices.

Storage: `uploads/recipes/mapping/` (first `.yaml` / `.yml` file found is used).

10.7.4.6 Recipe Files Section



Element	Description
Section title	"Recipe Files"
Download Template	Downloads bundled `recipe_template.yaml`
Upload	Adds recipe YAML to recipes folder
Recipe dropdown	`sms-dropdown-select` — lists all recipe files

Element	Description
Selected recipe row	Collapsible preview with download and delete actions
Preview	Raw YAML expected versions
Empty state	"No recipe files uploaded"
Overwrite confirm	Modal on duplicate filename upload

Behavior on upload/delete:

- Recipe file upload invalidates that file's parse cache and revalidates all devices mapped to it.
- Recipe file delete invalidates cache and revalidates affected devices.

Storage: `uploads/recipes/files/`

10.7.4.7 Recipe and Mapping File Formats

10.7.4.7.1 Recipe File

Defines expected versions. All sections and fields are optional — only listed fields are validated; others appear as Not Validated in the UI.

```yaml
software:
  image: nvos-25.02.1783
  kernel: 5.10.0-30-2-amd64
  onie: 2023.11-5.3.0012-115200
firmware:
  ASIC: 35.2014.1598
  BIOS: 0ACTV_00.01.010
  BMC: 88.0002.0912
```

Section keys are matched flexibly (case-insensitive substring: `software`, `firmware`). Firmware keys must match agent-reported component names.

Bundled template: `src/cablevalidation/collector/recipes/templates/recipe_template.yaml`

10.7.4.7.2 Mapping File

Uses hierarchy path-based rules, per-device overrides, and global default.

```yaml
recipe_rules:
  - path: "DH1/SU1/RP1"
    recipe: spine_recipe.yaml
  - path: "DH1/SU2"
    roles:
      switch: leaf_recipe.yaml
      host: gpu_host_recipe.yaml
  - pattern: "mgmt-.*"
    recipe: mgmt_recipe.yaml
  - roles:
      switch: default_switch_recipe.yaml
      host: default_host_recipe.yaml
overrides:
  spine-01: spine_special.yaml
default: base_recipe.yaml
```

Bundled template: `src/cablevalidation/collector/recipes/templates/recipe_mapping_template.yaml`

10.7.4.8 Device-to-Recipe Mapping Resolution

Resolution order (7 levels, highest priority first):

Priority	Level	Source

Priority	Level	Source
1	Per-device override	`overrides` in mapping YAML or UI assignment
2	Rack path	`recipe_rules` with `path: "DH/SU/Rack"`
3	SU path	`recipe_rules` with `path: "DH/SU"`
4	DH path	`recipe_rules` with `path: "DH"`
5	Pattern	`recipe_rules` with `pattern` (regex on hostname; first match wins)
6	Role-only	`recipe_rules` with `roles` only (no path/pattern)
7	Global default	`default` key

Path rule notes:

- Path rules are sorted by depth (rack > SU > DH) regardless of file order.
- Rules may include `exclude.node_name` regex list — excluded devices fall through to lower-priority rules.
- Rules may use `roles` instead of `recipe` for per-`node_type` splits (`switch`, `host`, etc.).
- Role entries may include per-role `exclude` blocks.

Topology attributes used: `data_hall`, `su_number`, `rack` on each node → path `DH/SU/Rack`; `node.get_type()` for role.

Unmapped device: No recipe file; all version fields show No Recipe when agent data exists.

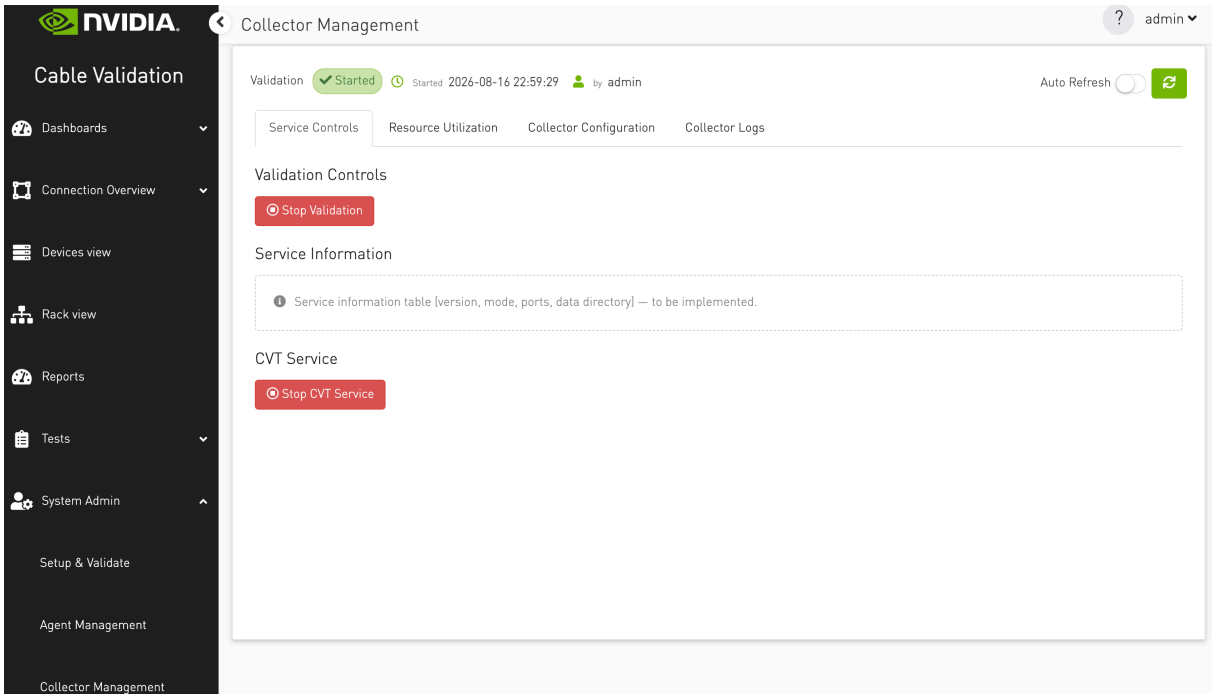
10.7.5 Collector Management

Collector management contains tabbed view for operating the CVT collector service and monitoring its health.

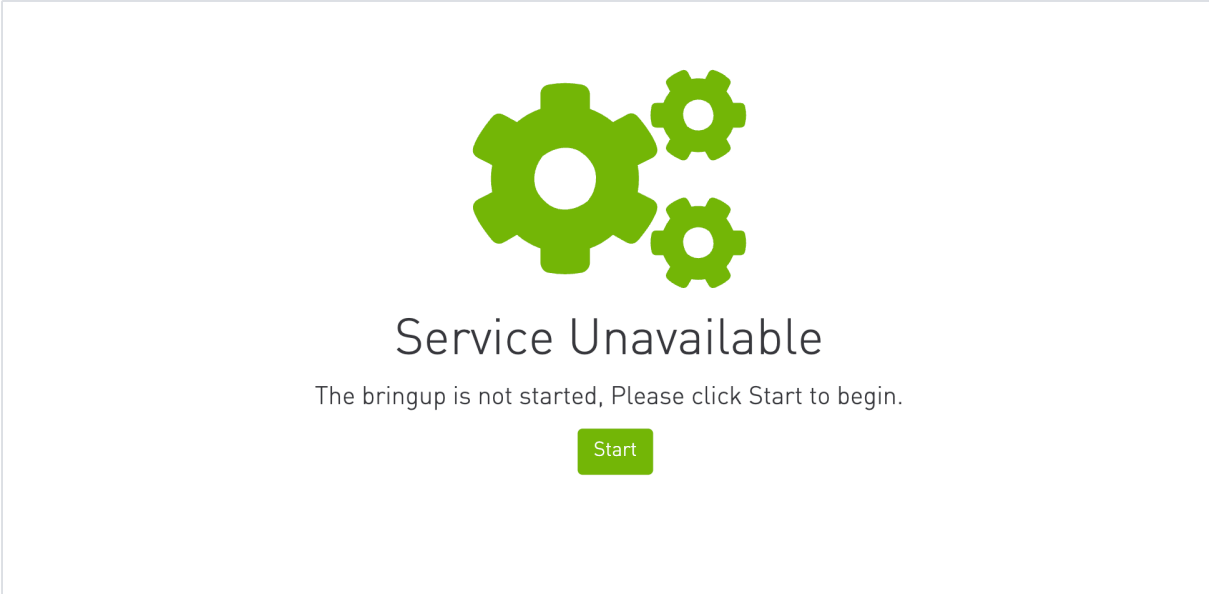
10.7.5.1 Service Controls

Operational controls for validation and the CVT service itself.

- Validation Controls: Start Validation / Stop Validation button, with state-aware labels (Starting..., Stopping...). Command output opens in a polling results window.
- Service Information: Placeholder for a future table showing version, mode, ports, and data directory.
- CVT Service: Stop CVT Service with confirmation modal. On success, redirects to the server error handler view (service is no longer available).



When clicked, the service is gracefully stopped, and the user is redirected to a dedicated page that displays the current service status.

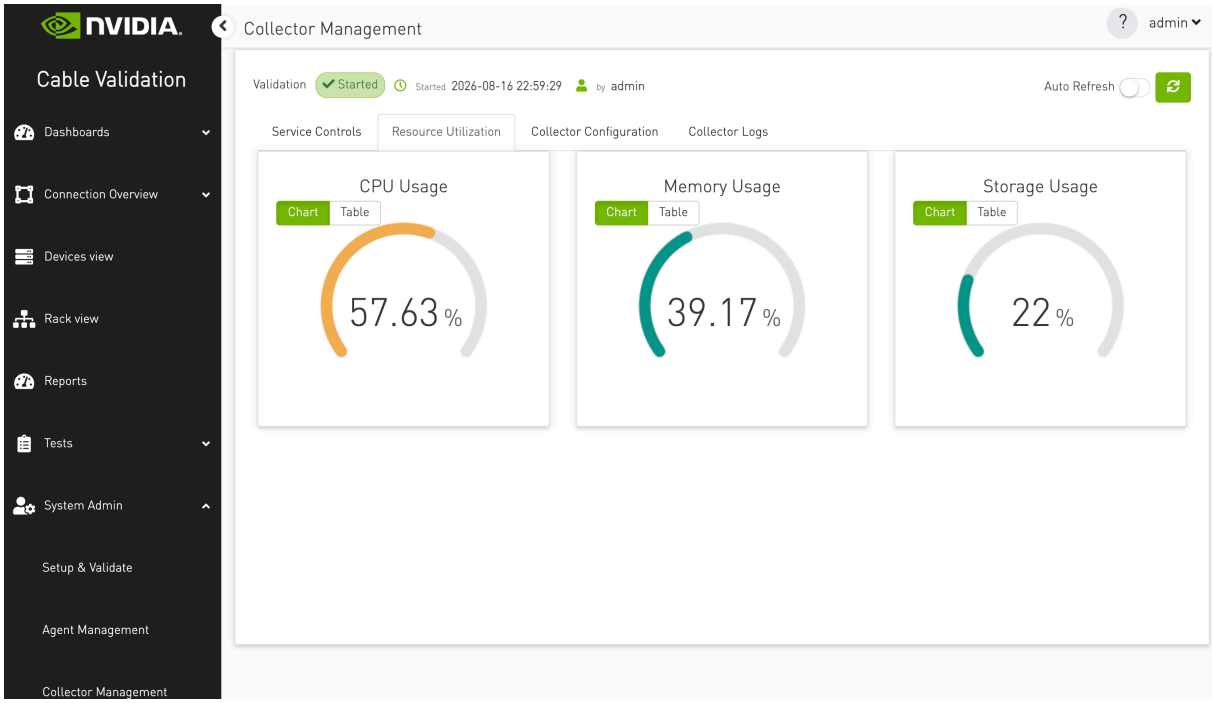


10.7.5.2 Resource Utilization

Resource utilization is used to track CVT system health and understand how it impacts resource consumption. It will be analyzed across three key areas: storage, memory, and CPU usage.

This analysis will help monitor system performance, optimize resource allocation, and ensure efficient operation.

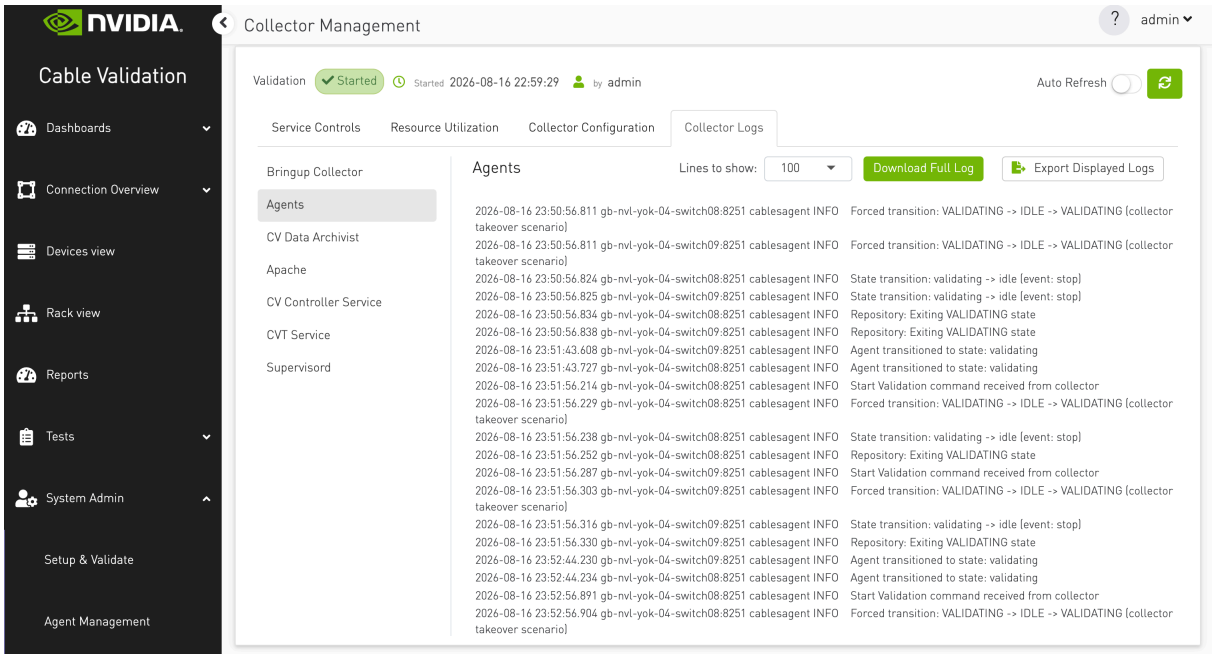
CPU utilization is calculated over a 3-second interval, meaning it will take 3 seconds for the CPU utilization data to be available.



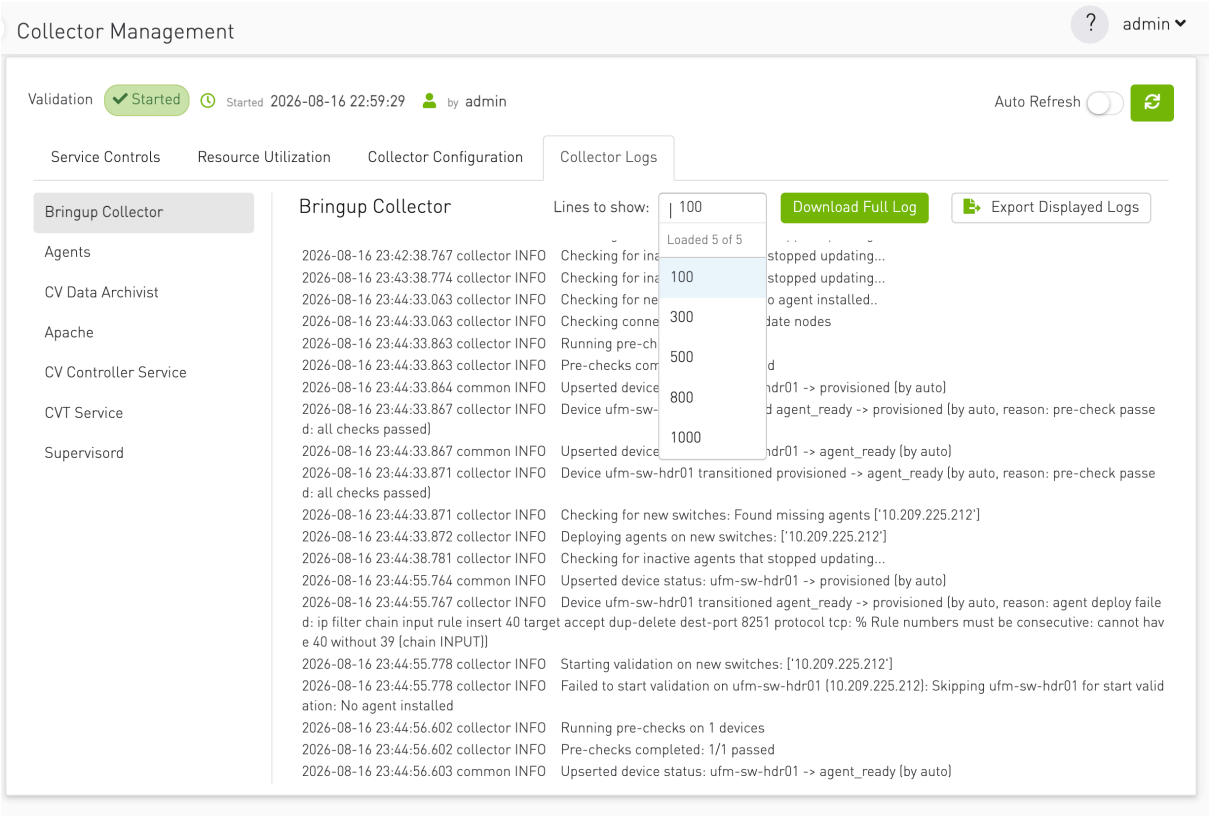
10.7.5.2.1 Collector Logs

Sidebar — log types from **GET /logs/collector** (labels such as *Bringup Collector*, *Agents*, *Apache*, ...).

- Selecting a type **automatically loads** the latest tail.
- **Lines to show** — dropdown: 100, 300, 500, 800, 1000; changing it refetches the tail for the current log type.
- **Download Full Log** — **GET .../download** for the selected type.
- **Export Displayed Logs** — saves the text currently in the panel (the tail), not the full file.



Option to select number of latest logs to display.
Click on buttons to download full log file or export the latest logs loaded on the page:

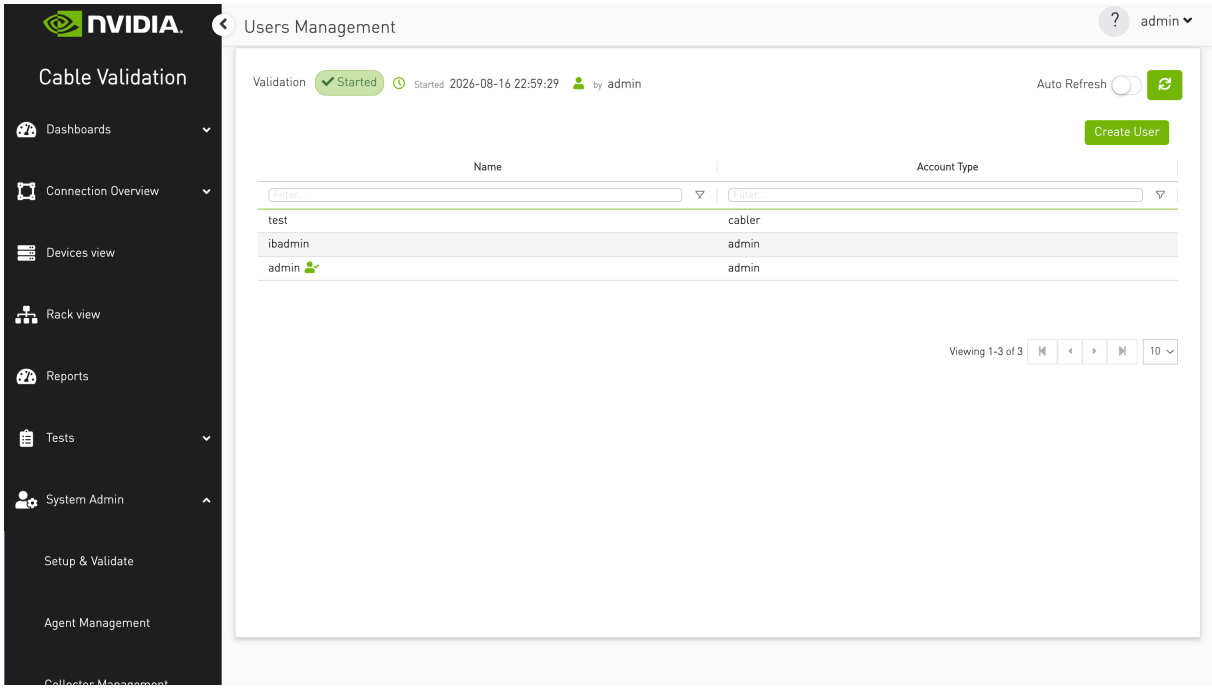


10.7.6 Users Management

Users Management is where administrators create and maintain **local CVT login accounts** for standalone deployments.

Availability: Standalone CVT deployments only. Hidden and route-guarded when CVT runs embedded in UFM.

User table columns: Name (with icon for the currently logged-in user), Account Type.

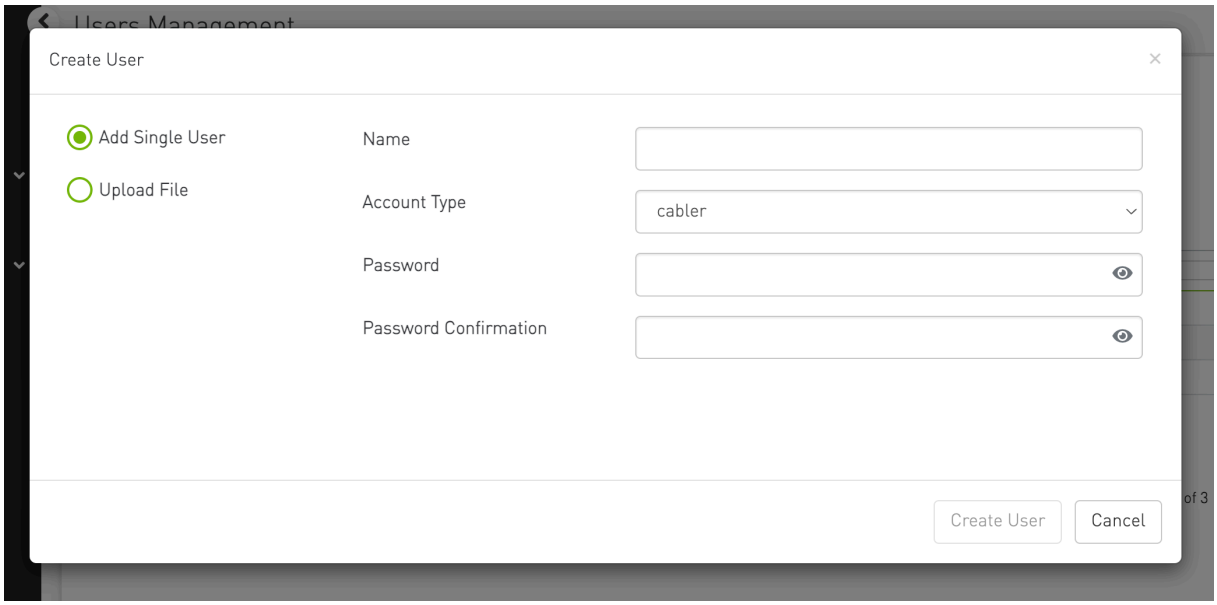


Actions:

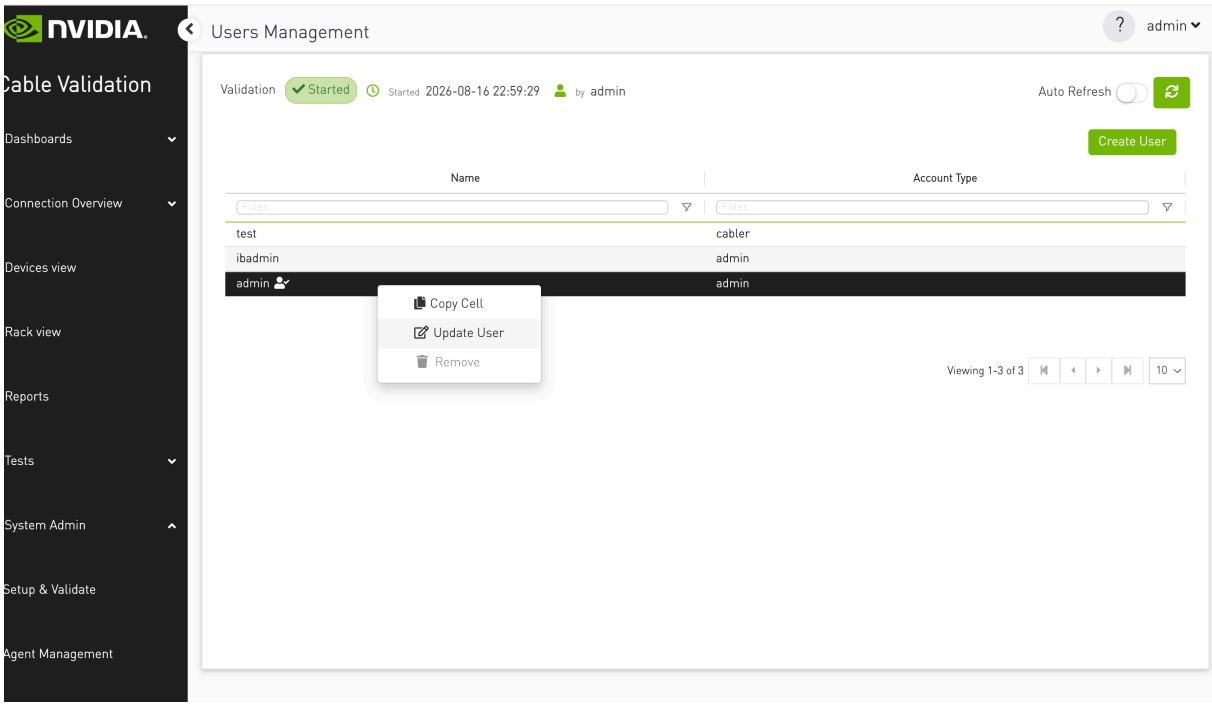
Action	How
Create User	Toolbar button opens a modal with two modes: Add Single User (name, password, account type) or Upload File (bulk import)
Update User	Right-click context menu → Update User
Remove User	Right-click context menu → Remove (disabled for the currently logged-in user)

Account types are loaded from the backend and presented in the create/update forms.

Click on Create button to add new users.

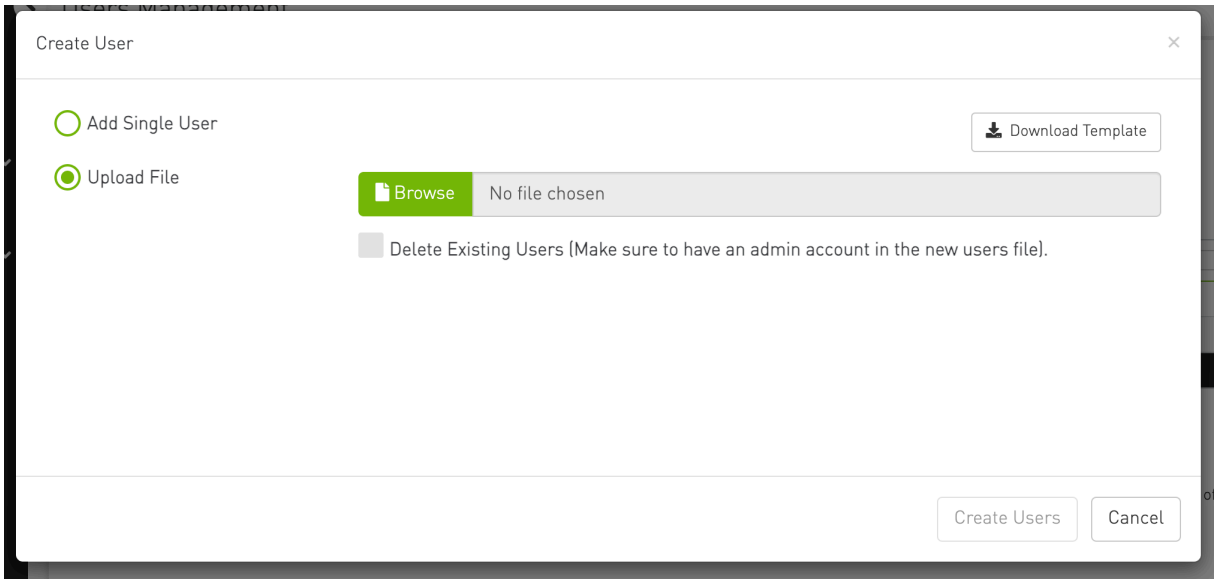


Right click to update or delete the user



10.7.6.1 Upload File

This option allows to upload multiple users in one go. The user details uploaded from a json file (template is available for download).



Bulk upload supports a **Download Template** option and an optional **Delete Existing Users** flag (the new file must include at least one admin account).

10.7.6.2 Account types

Four account types are supported:

Type	Typical use
admin	Full access — System Admin, setup wizard, tests, agent management, and other restricted features
cabler	Standard operator — some admin-only actions are hidden; certain operations (e.g. circuit simulation) are restricted
nvidia	NVIDIA internal user
developer	Developer access

Account type controls what appears in the sidebar and which actions are available across the app. The backend enforces admin-only API operations separately.

10.7.6.3 Safeguards

- Usernames must be unique and cannot use reserved names (e.g. `system`)
- Password length and format are validated on create/update
- The **last admin account** cannot be deleted or demoted to a non-admin type
- You cannot remove your own account while logged in

10.8 Auto Refresh

Most pages in the tool contain auto refresh feature to ensure content remains up-to-date.

Users can perform the following actions related to the auto-refresh functionality:

10.8.1 Enable or Disable Auto-Refresh

Toggle the auto-refresh feature on or off based on your preferences.

10.8.2 Adjust the Refresh Interval

Customize the interval time for auto-refresh to control how frequently the page updates automatically.

10.8.3 Manually Refresh the Page

Trigger a manual refresh of the page at any time to view the latest updates immediately.

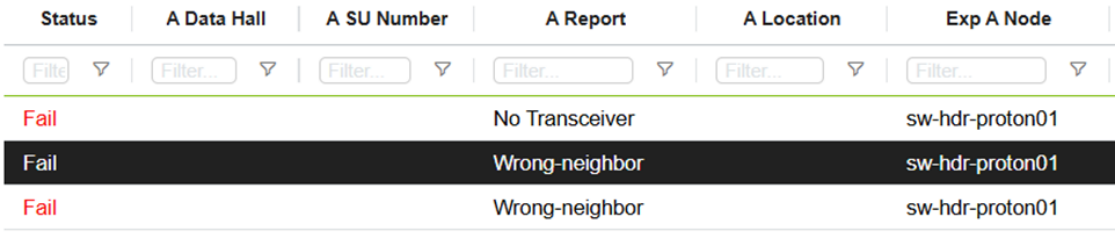
10.9 Table

All tables in the tool contain a general functionality, this chapter will describe them.

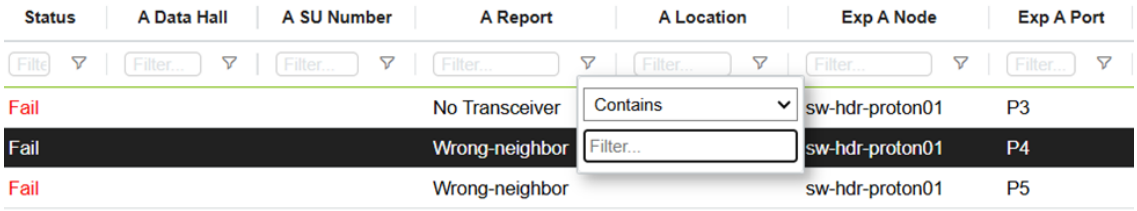
10.9.1 Filter

The tables provide advanced search capabilities, offering users two primary ways to utilize the search mechanism:

1. Search Input Field Below the Table Name:
   A straightforward search field located beneath the table name allows users to quickly filter table content based on their input and selected filter type.



2. Search Menu via the Column Search Icon:
   Clicking the search icon beside the search input opens a dedicated search menu. This menu provides additional options and customization for filtering data.

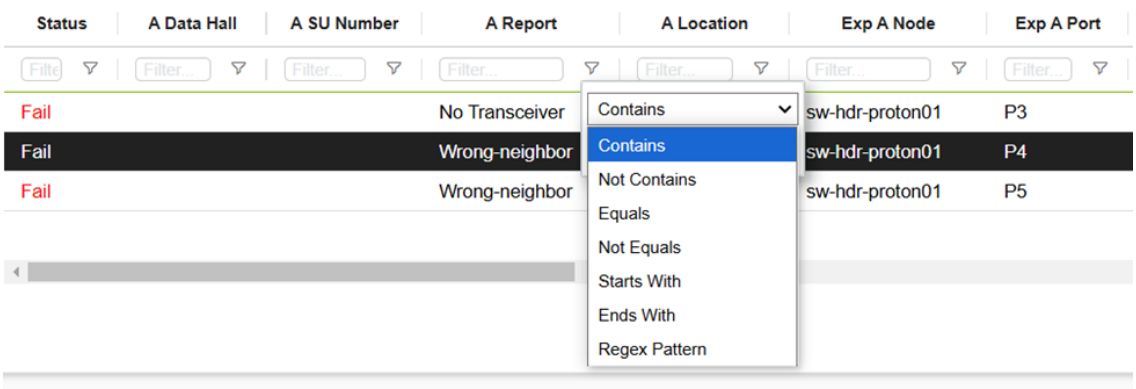


Users can change the search type **only** through the search menu accessible via the column search icon.

10.9.1.1 Supported Filter Types

The table supports the following types of filters to refine search results:

1. **Contains:** Finds records that include the specified text.
2. **Does Not Contain:** Excludes records that include the specified text.
3. **Equals:** Displays records that exactly match the specified value.
4. **Does Not Equal:** Excludes records that exactly match the specified value.
5. **Starts With:** Finds records that begin with the specified text.
6. **Ends With:** Finds records that end with the specified text.
7. **Regular Expression (Regex) Pattern:** Enables advanced searches using custom regular expressions for precise filtering.



10.9.2 Layout Manager

10.9.2.1 Layout Manager

The Layout Manager lets you save and reuse a table configuration. Use layouts to create focused views for troubleshooting or monitoring and to switch between them without manually reconfiguring the table each time.

A layout can save the following table settings:

- Column width and order
- Visible and hidden columns
- Pinned columns
- Column sorting

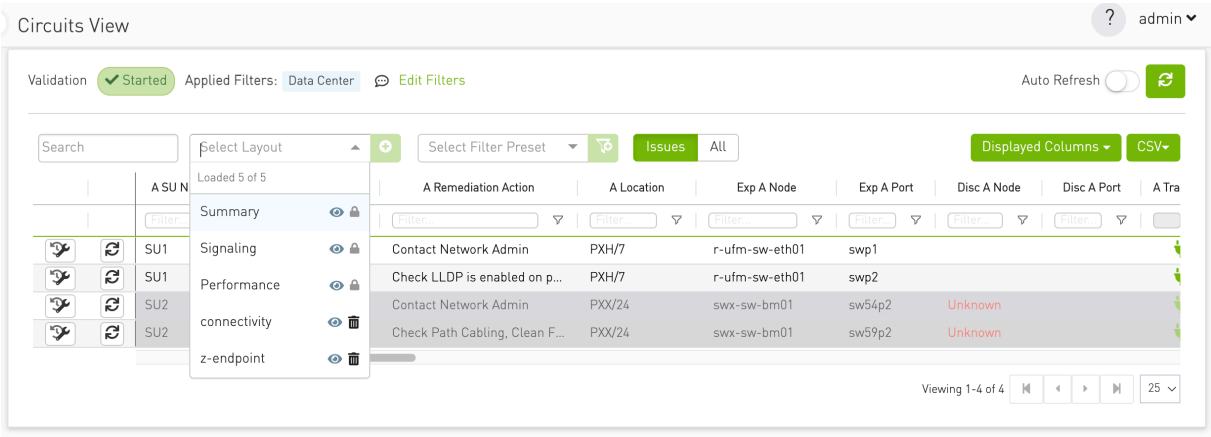
10.9.2.1.1 Supported Tables

The Layout Manager is available in the following views:

- Circuits
- Flapping Circuits
- Golden BER and other test circuit details
- Amber test details

The Layout Manager may not be available in other tables.

10.9.2.1.2 Layout Controls



The table toolbar contains a layout list and a control for creating a layout. Depending on the selected layout and its current state, the following actions are available:

Action	Description
Select a layout	Applies the saved table configuration.
Create	Saves the current table configuration as a new layout.
Save	Updates the selected personal layout with your current changes.
Reset	Discards unsaved changes and restores the last loaded or saved state.
View summary	Displays the settings stored in the layout.
Delete	Removes a personal layout.

10.9.2.1.3 Layout Types

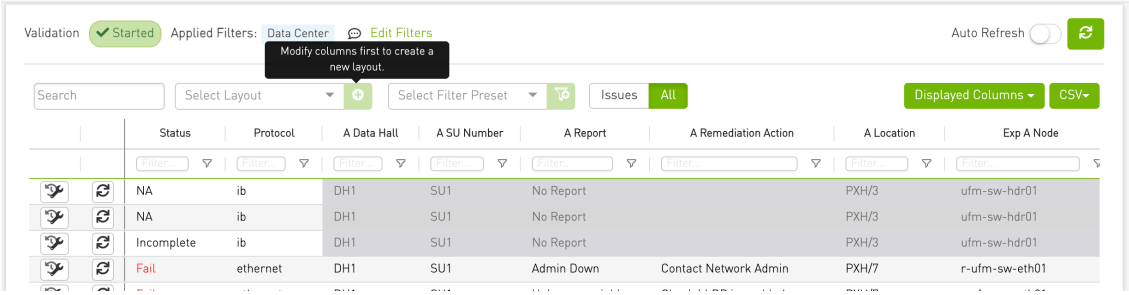
The layout list can contain the following types:

- Personal layouts are created by you and are visible only to your user account. You can update and delete them.
- Predefined layouts, when available, are provided by the system. A lock icon identifies them. You can apply and inspect predefined layouts, but you cannot update or delete them.

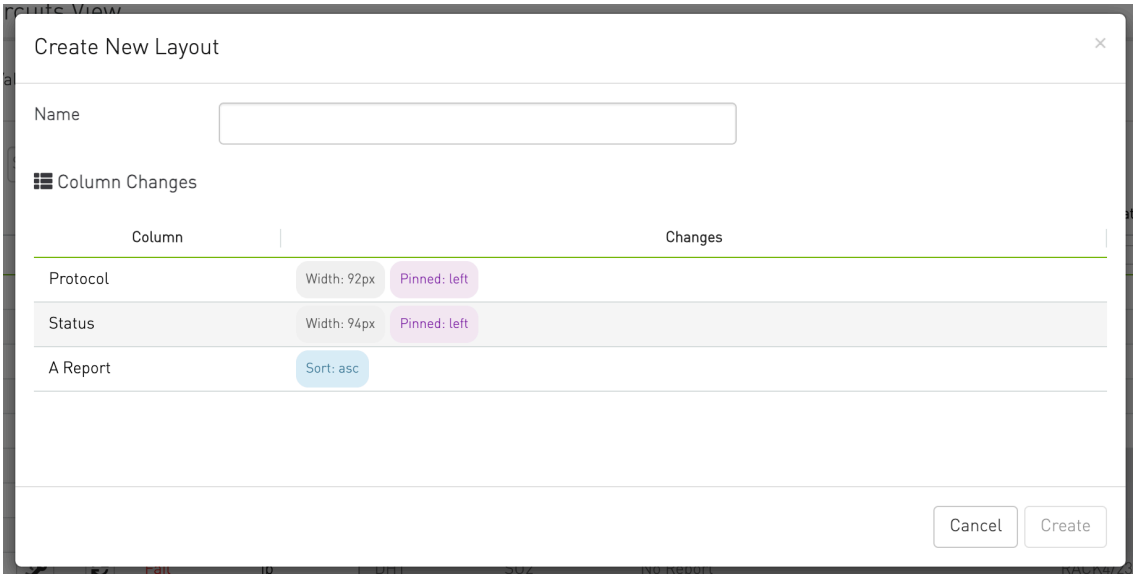
Layouts are kept separately for each table. A layout created in one view does not appear in another view.

10.9.2.1.4 Creating a Layout

1. Configure the table as needed. You can resize, move, show, hide, or pin columns and change sorting.

2. 

   Select the **Create** control in the table toolbar.



3. Enter a unique layout name.
4. Review the layout summary to verify the settings that will be saved.
5. Confirm the action.

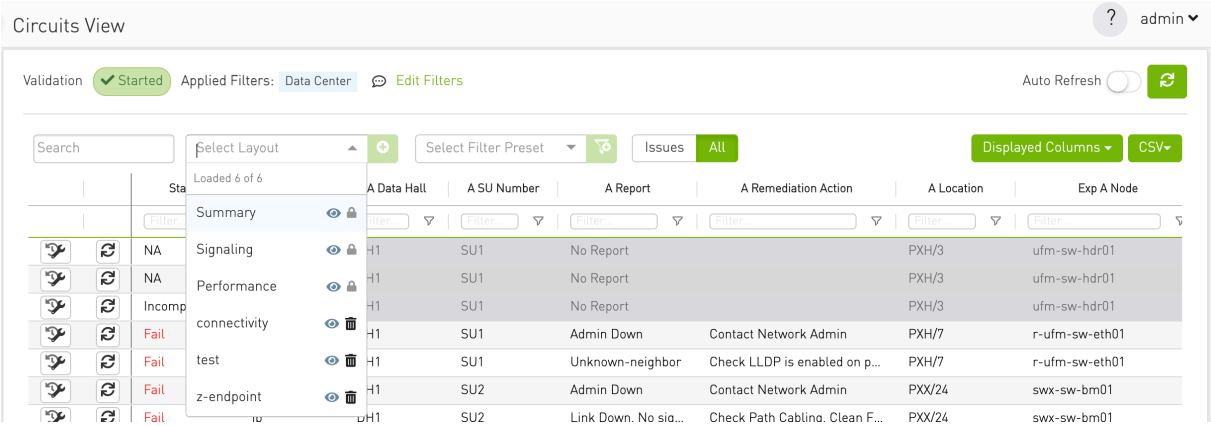
The new layout is added to the layout list and selected automatically.

A layout must contain at least one table change or filter. Layout names must:

- Start with a letter or number.
- Contain only letters, numbers, hyphens (-), underscores (_), or periods (.).
- Be unique among your layouts for the current table.
- Not contain spaces, slashes, backslashes, or two consecutive periods (..).

10.9.2.1.5 Applying or Switching Layouts

Open the layout list and select a layout. The saved column settings are applied immediately.



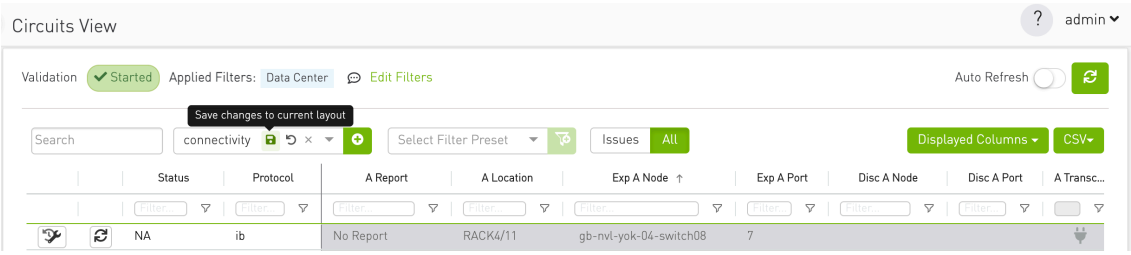
The selected layout is remembered for the current table. When you return to the table or refresh the browser, CVT reapplies it if the layout still exists. If it is no longer available, CVT uses a default predefined layout when one is configured; otherwise, the table opens in its default state.

To return to the table's default configuration, clear the selected layout.

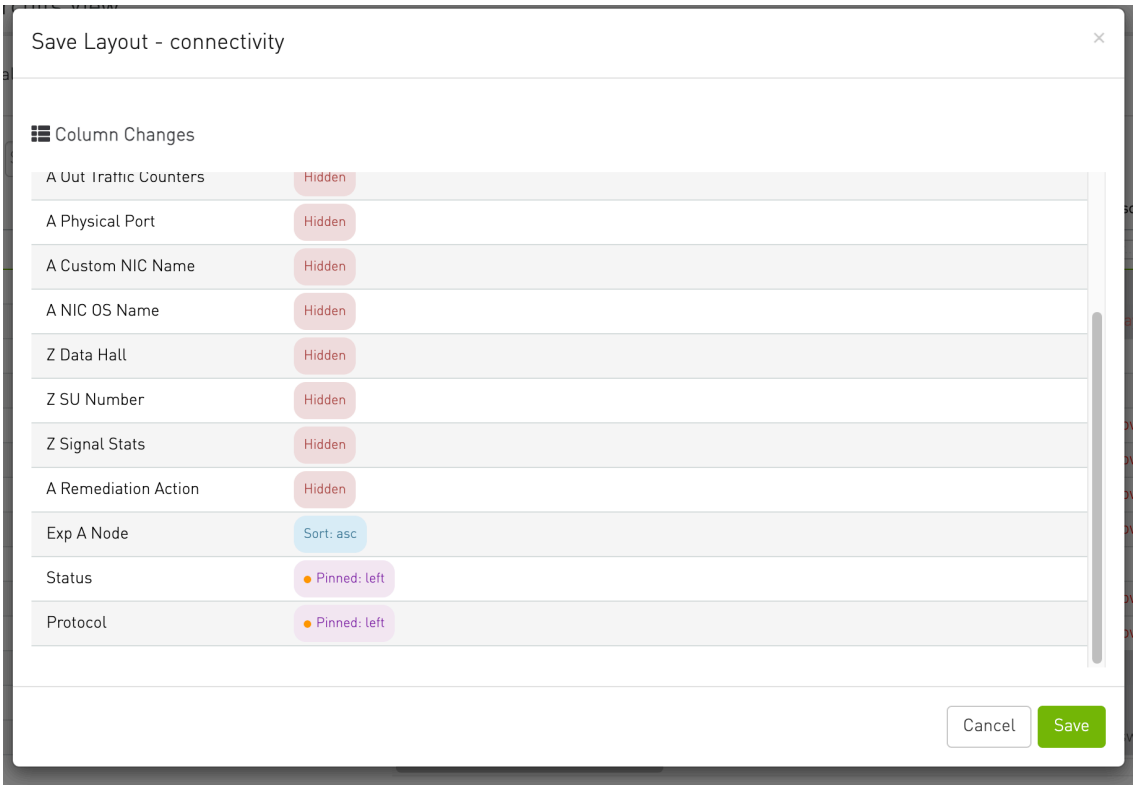
10.9.2.1.6 Updating a Personal Layout

When you change the table after selecting a personal layout, **Save** and **Reset** actions appear.

1. Modify the table configuration.
2. Select **Save**.



3. Confirm the update.



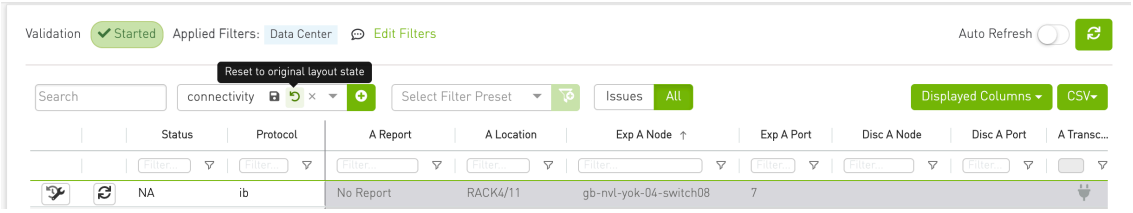
The current table settings replace the previously saved settings for that layout.

Predefined layouts cannot be updated. To preserve changes based on a predefined layout, create a new personal layout.

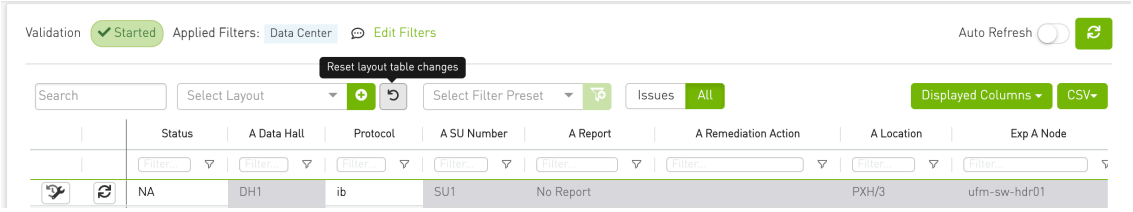
10.9.2.1.7 Resetting Changes

Use **Reset** to discard changes that have not been saved:

- If a layout is selected, the table returns to that layout's last loaded or saved state.



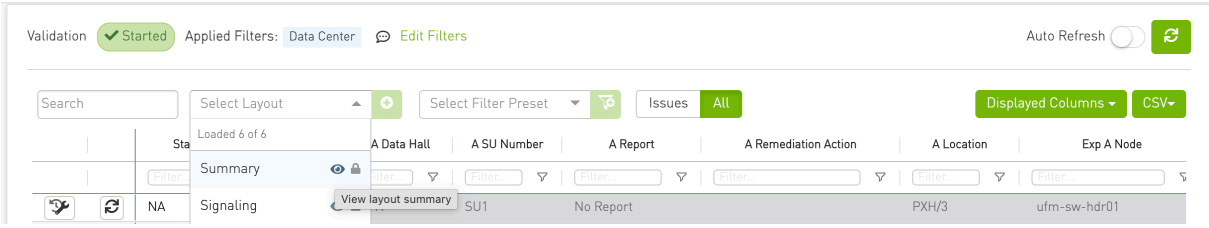
- If no layout is selected, the table returns to its initial default state.



Resetting does not delete the layout.

10.9.2.1.8 Viewing a Layout Summary

Select the **View summary** icon next to a layout to review its saved settings.



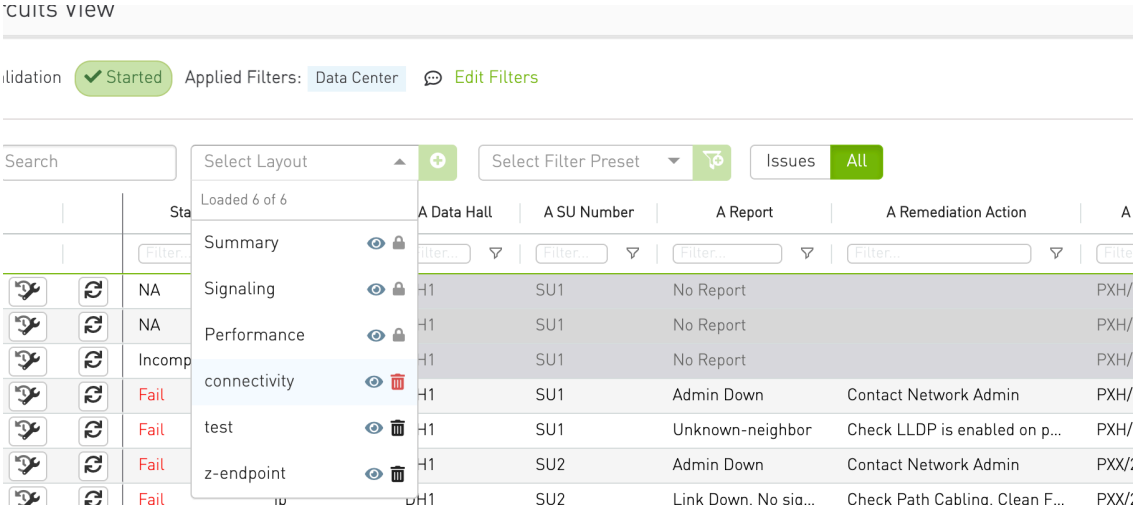
The summary can include:

- Quick-filter text
- Column filters
- Sorting
- Column width and order
- Pinned columns
- Visible and hidden columns

You can view the summary for both personal and predefined layouts.

10.9.2.1.9 Deleting a Personal Layout

1. Open the layout list.
2. Select the **Delete** icon next to the personal layout.



3. Confirm the deletion.

Deleting a layout cannot be undone. If the deleted layout is currently selected, CVT clears the selection and returns the table to its default state.

10.9.2.1.10 Additional Notes

- Personal layouts are private to your user account. Other users cannot view, update, or delete them.
- Different users can use the same layout name.
- Layouts are independent for each supported table.
- CSV export follows the table's current visible column order.
- In the save layout modal the new changes will be marked with an amber dot

10.10 Resource Filter

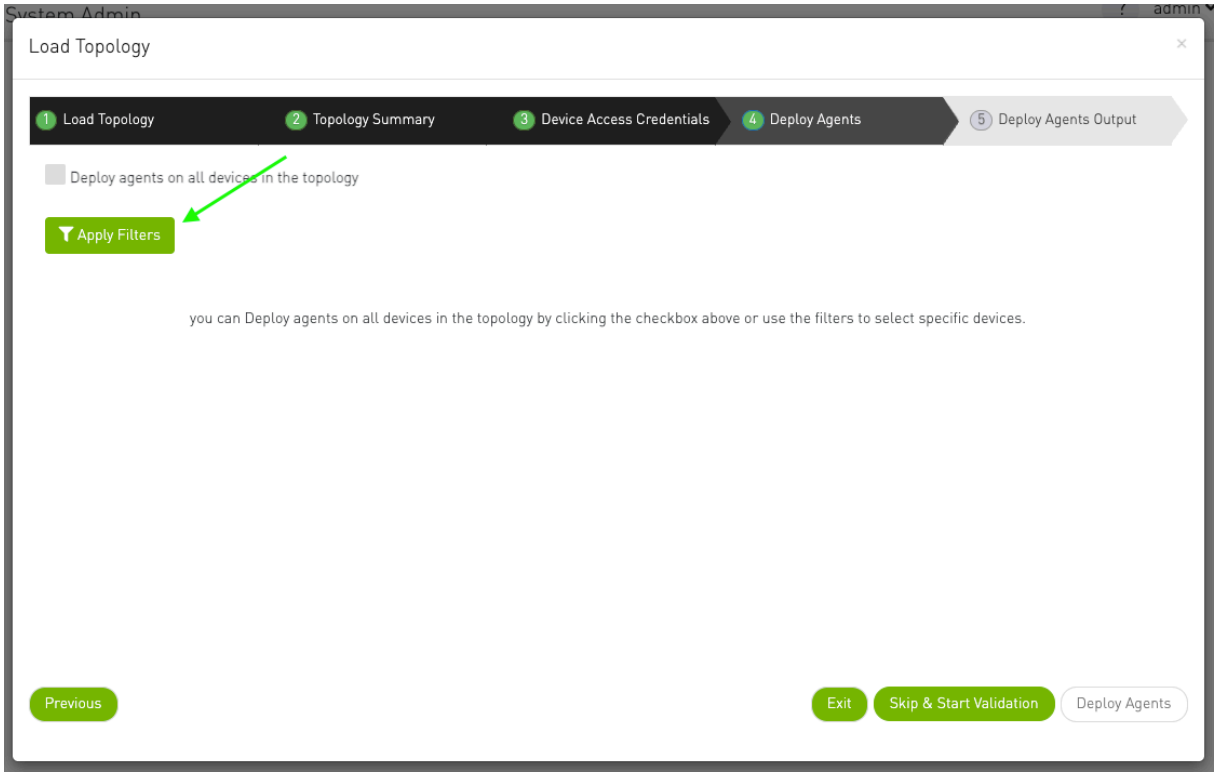
The **Resource Filter** is a shared filtering mechanism within the **Cable Validation Tool (CVT)** that provides a unified and flexible way to refine data across multiple views and workflows. It is designed to enhance usability, consistency, and efficiency in data selection.

This filter is used in several key areas of the application, such as the **Circuits View** and the **Deploy Agents** step of the **Deploy Agent Wizard**.

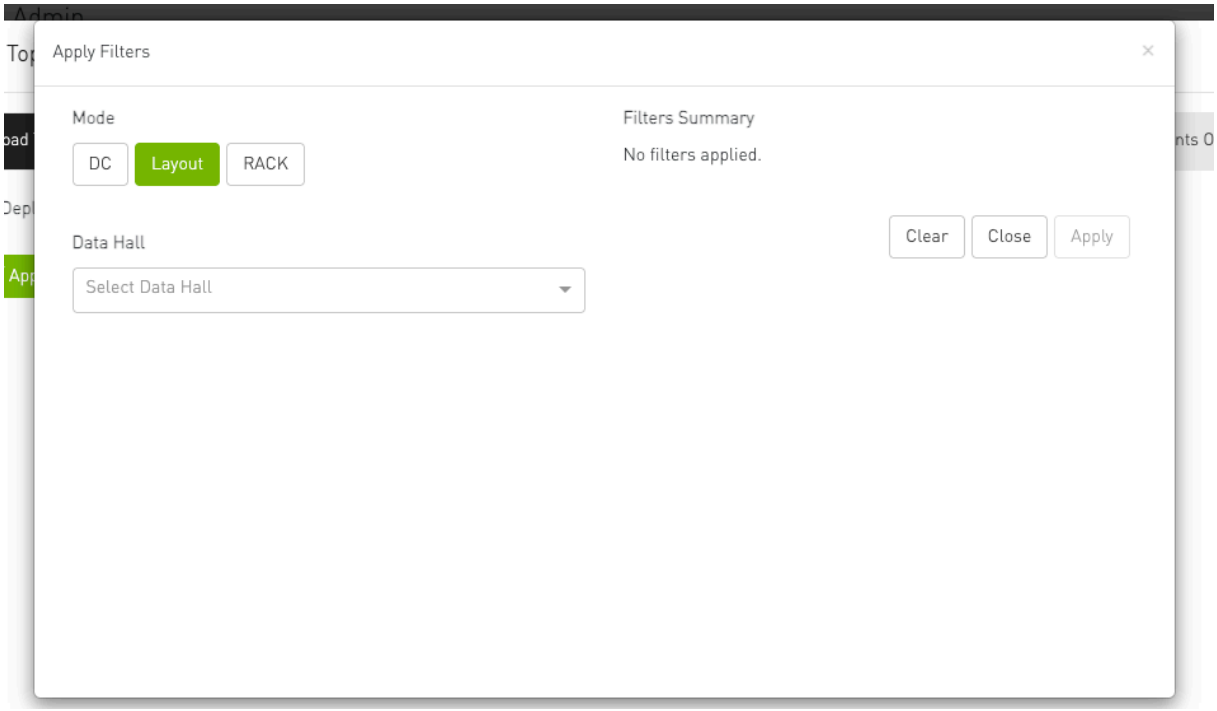
The main purpose of the Resource Filter is to help users easily narrow down large datasets—such as circuits or agents—based on hierarchical or contextual criteria.

The filter is accessible through a **Filter** button that appears on any page or component supporting this functionality.

For example, in the **Deploy Agents** step of the **Deploy Agent Wizard**, the **Filter** button appears as shown below:



When the user clicks the **Filter** button, a modal window opens, allowing them to define their filtering criteria.

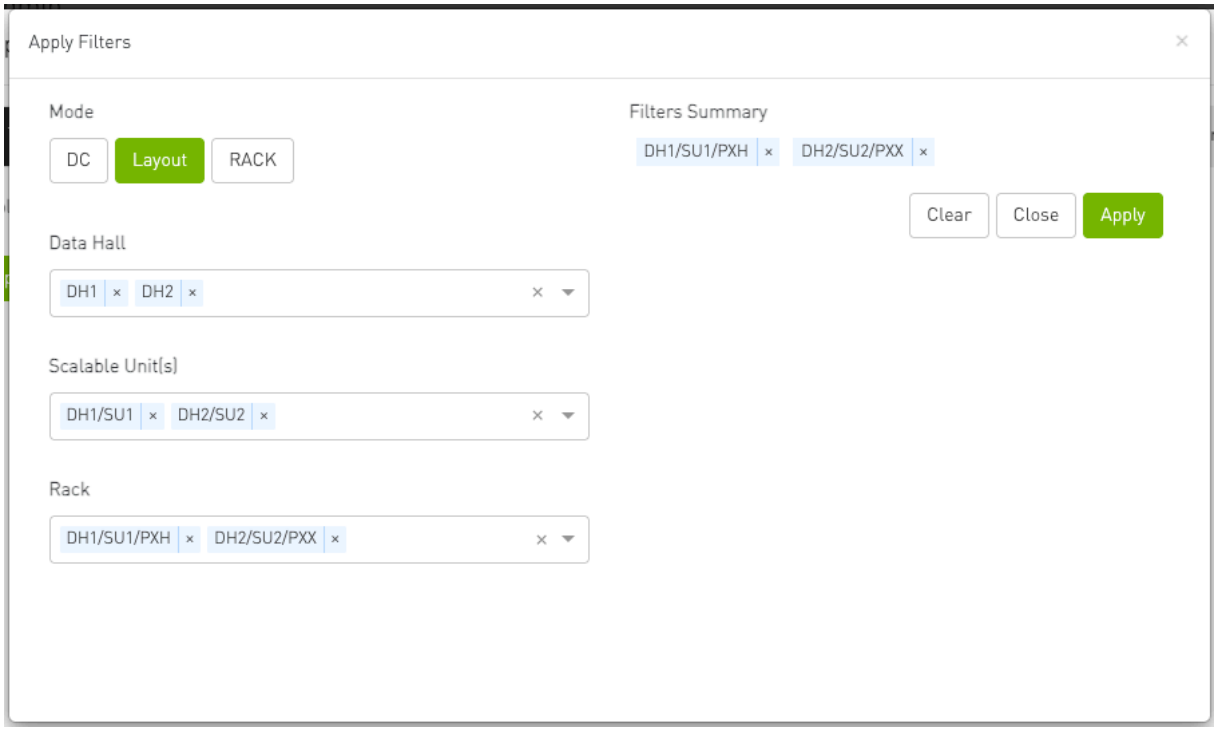


10.10.1 Filter Modes

The General Filter Modal supports three distinct modes, providing flexibility depending on the user's context and workflow:

10.10.1.1 Layout (Hierarchical Mode)

- Enables drill-down selection through topology layers: Data Hall → SU → Rack.
- Dynamically filters devices as the user navigates through each level.
- Provides a clear hierarchical context for every selection step.

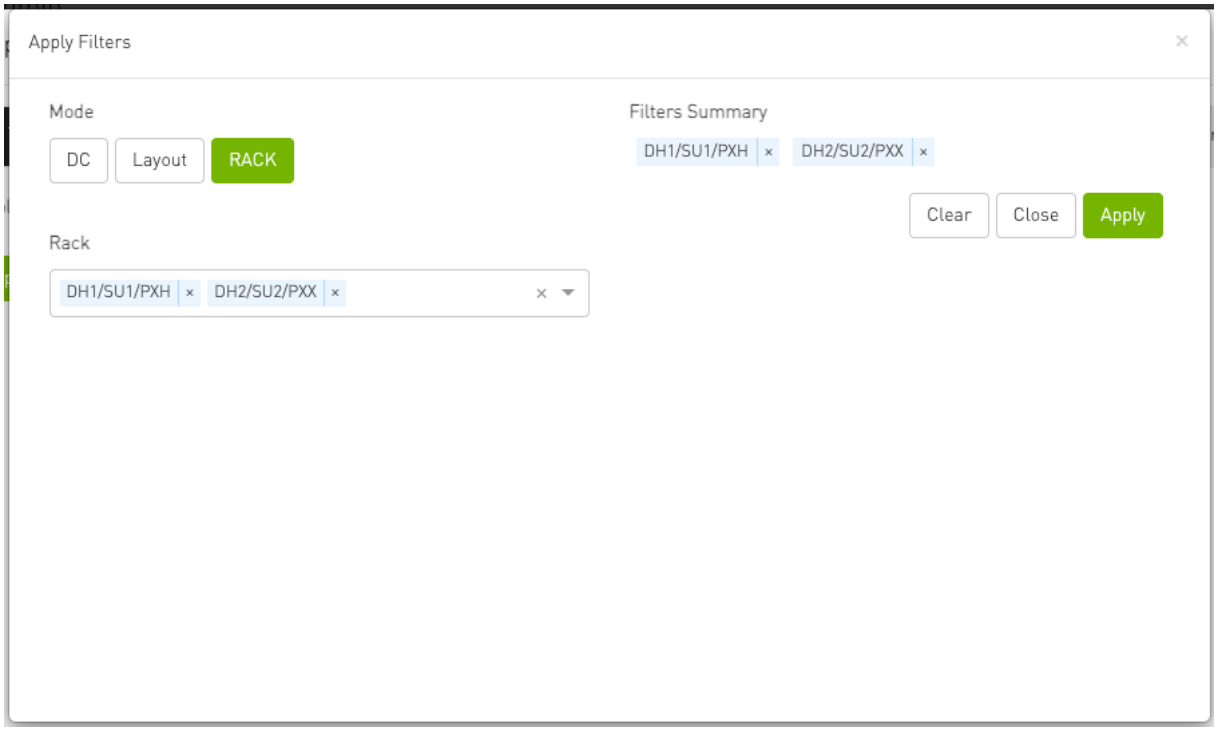


10.10.1.2 Data Center (DC Mode)

- Displays all available resources (e.g., devices) within the selected data center.
- Not preferred with large-scale environments.

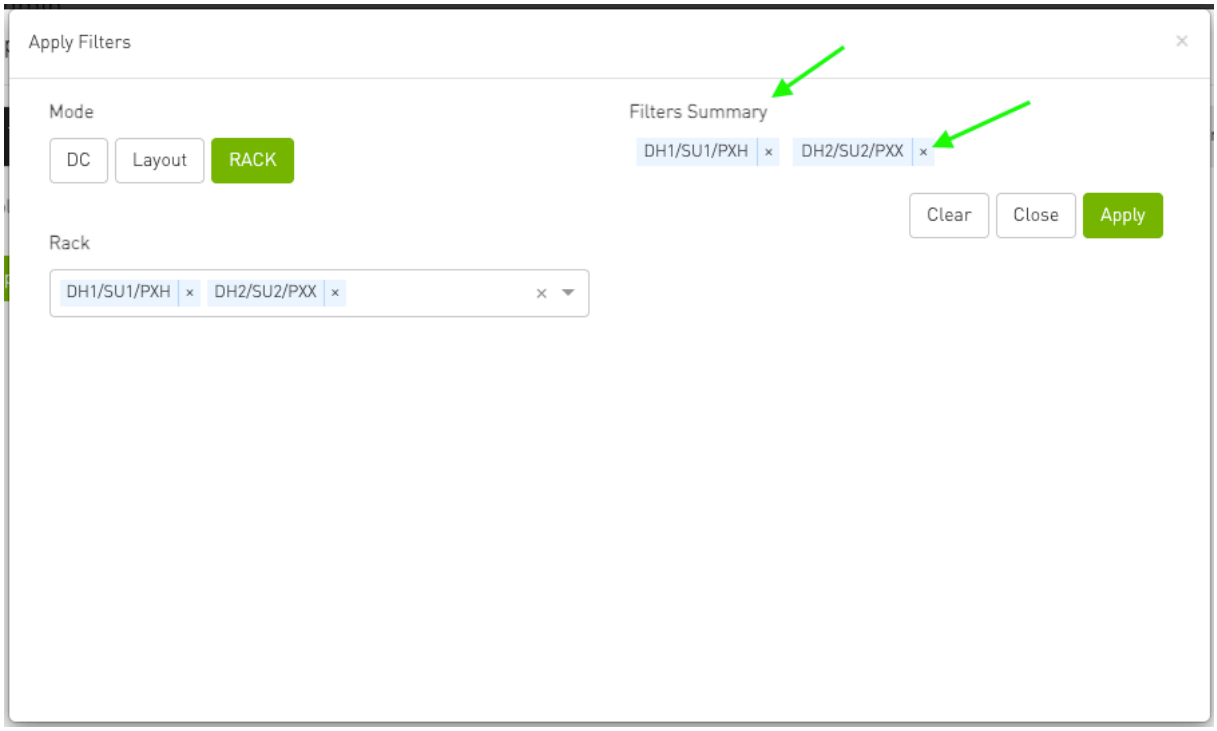
10.10.1.3 1.4.1.3. Rack Mode

- Provides a flat, searchable list of all racks in the topology.
- Useful for systems that do not include Data Halls or SUs, or when the rack name is already known.



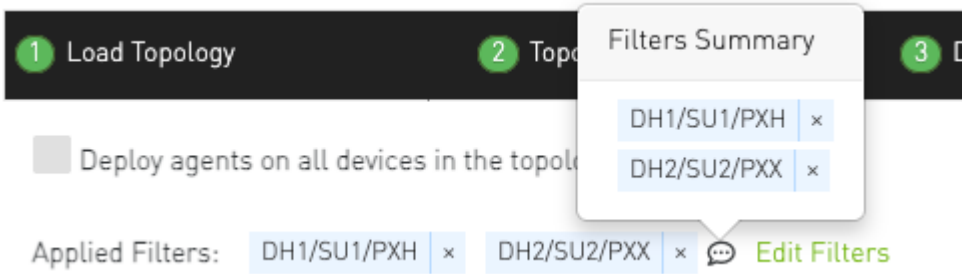
10.10.2 Filter Summary and Results

A **summary panel** is displayed on the left side of the modal, showing all selected filters.
Users can review and remove individual filters directly from this panel before applying the filter.



Once **Apply** is clicked:

- The system filters the resources or devices according to the selected criteria.
- The applied filter details appear beside the **Filter** button for visibility, as shown below:



> The Resource Filter remembers your last applied filters when you navigate between pages that share the same filtering mechanism.
> For example, a filter applied in **Circuit View** will automatically appear in **Flapping Circuit View**.
>
> However, filters that are opened inside a **modal window** (for example, within a wizard or pop-up dialog) **are not remembered**.
> Each time you reopen the modal, you'll need to set your filters again.

11 Rest APIs

11.1 Bringup Server REST API

The collector has a web server listening on two internal port **8251**. This port is not advertized outside the machine. The bringup server is running an *Apache* server which uses the default https port.

11.2 Overriding Collector Web server Port

Users can modify the collector web server port as the following:

- Plugin
  a. Execute /opt/ufm/scripts/manage_ufm_plugins.sh add -p cablevalidation to add the Cable Validation plugin.
  b. Stop the plugin using /opt/ufm/scripts/manage_ufm_plugins.sh stop -p cablevalidation.
  c. Use vim /opt/ufm/files/conf/plugins/cablevalidation/config.cfg to modify the BRINGUP_PORT variable.
  d. Update and save the file.
  e. Start the plugin again with /opt/ufm/scripts/manage_ufm_plugins.sh start -p cablevalidation.
- Standalone
  a. Pass the new port as an environment variable by executing docker run --name cables_bringup -itd --network=host --env BRINGUP_PORT=<new-port> cables_bringup.

With these changes, the new configuration will take effect, and Apache will run with the updated ports.

11.3 Reports APIs

11.3.1 Get Validation Report

- Description – Gets the recent cable validation report
- Request URL – **GET /cablevalidation/report/validation**
- Response Content Type – application/json
- Status Codes
  - 200 – OK
  - 404 – NOT FOUND
- Response Example

```
{
  "report": "ValidationReport",
  "stats": {
    "in_progress": 14,
    "no_issues": 8,
    "not_started": 5
  },
  "issues": [
    {
```

```
      "timestamp": 1722290755.0392804,
      "node_desc": "dell001.cm.cluster",
      "rack": "D01",
      "unit": 2,
      "issues": [
        ["Unreachable-device"]
      ]
    }
  ],
  "metadata": {
    "file_name": "v22_testing_extracable.xlsx",
    "file_hash": "d2a8ce40b613a17245acb3310fa0b810238dc61613e8f683b95139d19567aa5a",
    "load_time": 1722290305.0386324
  }
}
```


11.3.2 Get Validation Status

- Description – Gets the current validation status and timing information.
- Request URL – **GET /cablevalidation/validation/status**
- Response Content Type – application/json
- Status Codes
    - 200 – OK
    - 500 – INTERNAL SERVER ERROR

The **started_at**, **started_by**, **duration_seconds**, and **last_cycle_seconds_ago** fields can be **null** when validation has not started or when the corresponding information is unavailable.

- Response Data Example


```
{
  "status": "started",
  "started_at": 1767225600,
  "started_by": "admin",
  "duration_seconds": 3600,
  "last_cycle_seconds_ago": 12
}
```


11.3.3 Get Topology Metadata

- Description – Gets topology metadata. When topology statistics are available, the response includes switch, host, and link counts.
- Request URL – **GET /cablevalidation/topology/metadata**
- Response Content Type – application/json
- Status Codes
    - 200 – OK
    - 400 – BAD REQUEST
- Response Data Example


```
{
  "file_name": "proton-ptp.xlsx",
  "file_path": "/cable_bringup_root/data/uploads/topology/ptp/proton-ptp.xlsx",
  "dc_layout_file_path": null,
  "file_hash": "c13187caece919c9aa88d2c1e26404fe5e3d0cd56ea801c4c46a7236e42549fb",
  "file_id": "94a8c27f",
  "load_time": 1732376613.3298793,
  "switches_count": 32,
  "hosts_count": 96,
  "links_count": 128
}
```

11.3.4 Get Summary Report

- Description – Gets the issue summary for a cluster. The response includes counts originating from stale agent reports when available.
- Request URL – `GET /cablevalidation/report/summary`
- Request Params
  - `cluster` – Name of the cluster. If set, the response returns the device health for that specific cluster.
    - Type: string
    - Optional: true
    - Default: default
- Response Content Type – application/json
- Status Codes
  - 200 – OK
  - 400 – BAD REQUEST
  - 404 – NOT FOUND
- Response Example

```
[
  {
    "syndrome": "No Transceiver",
    "number_of_occurrences": 2,
    "switches_affected": 2,
    "number_of_occurrences_from_stale": 1,
    "switches_affected_from_stale": 1
  },
  {
    "syndrome": "Link Down, No signal",
    "number_of_occurrences": 3,
    "switches_affected": 2,
    "number_of_occurrences_from_stale": 0,
    "switches_affected_from_stale": 0
  }
]
```

11.3.5 Get Circuit Report

- Description – Gets circuits information.
- Request URL – `GET /cablevalidation/report/circuits`
- Request Params
  - `cluster` – Name of the cluster. If set, the response returns circuits for that specific cluster.
    - Type: string
    - Optional: true
    - Default: default
  - `node` – Node name. If set, the response returns circuits for that node.
    - Type: string
    - Optional: true
  - `port` – Port name. If set, the response returns circuits for that port.
    - Type: string
    - Optional: true
  - `page` – Page type. Supported values: `circuit`, `flap`, and `flap_hist`.
    - Type: string
    - Optional: true

- **healthy** – If true, only healthy circuits are returned.
    - Type: bool
    - Optional: true
- **circuit_id** – Circuit identifier. If set, the response returns that circuit.
    - Type: string
    - Optional: true
- **report** – Syndrome name. If set, the response returns circuits with that syndrome.
    - Type: string
    - Optional: true

- Response Content Type – application/json
- Status Codes
    - 200 – OK
    - 400 – BAD REQUEST
    - 404 – NOT FOUND
- Response Example


```
[
  {
    "circuit_id": "c83df4d68b40",
    "a_endpoint": {
      "node_type": "Switch",
      "data_hall": "DH0",
      "su_number": "SU01",
      "node": "sw-hdr-proton01",
      "port": "P4",
      "rack": null,
      "unit": null,
      "actual_node": null,
      "actual_port": null,
      "port_status": "up",
      "plugged": true,
      "advanced_stats": {},
      "remediation_action": "Check LLDP is enabled on peer; Verify the peer is fully provisioned and reachable",
      "report": "Unknown-neighbor"
    },
    "z_endpoint": {},
    "healthy": false,
    "status": "Fail"
  }
]
```


11.3.6 Get Circuit Statistics

- Description – Gets a summary of circuit health and agent-related circuit conditions.
- Request URL – **GET /cablevalidation/report/circuits/stats**
- Response Content Type – **application/json**
- Status Codes
    - 200 – OK
    - 400 – BAD REQUEST
- Response Fields
    - **total_circuits** – Total number of circuits.
    - **healthy_circuits** – Circuits without a detected issue.
    - **unhealthy_circuits** – Active circuits with detected issues.
    - **not_active_circuits** – Circuits that are not active.
    - **unreachable_circuits** – Circuits affected by unreachable endpoints.
    - **stale_agent_circuits** – Circuits affected by stale agent reports.
    - **not_tracked_circuits** – Circuits that are not currently tracked.
    - **one_side_stale_circuits** – Circuits with a stale report on one endpoint.

    - `both_sides_stale_circuits` – Circuits with stale reports on both endpoints.
    - `no_agent_circuits` – Circuits with at least one endpoint that does not have an agent.
- Response Example

```
{
  "total_circuits": 250,
  "healthy_circuits": 210,
  "unhealthy_circuits": 25,
  "not_active_circuits": 15,
  "unreachable_circuits": 8,
  "stale_agent_circuits": 6,
  "not_tracked_circuits": 4,
  "one_side_stale_circuits": 3,
  "both_sides_stale_circuits": 2,
  "no_agent_circuits": 5
}
```

11.3.7 Get Device Health Report

- Description – Gets the device health summary report.
- Request URL – `GET /cablevalidation/report/health`
- Request Params
    - `cluster` – Name of the cluster. If set, the response returns device health for that specific cluster.
        - Type: string
        - Optional: true
        - Default: default
    - `context` – `dc`, `dh`, `su`, `rack`, or `node`.
    - `items` – Report scope, typically a list of data halls, scalable units/data halls, racks, or nodes depending on the selected context. Do not include this parameter when `context` is `dc`.
- Response Content Type – application/json
- Status Codes
    - 200 – OK
    - 400 – BAD REQUEST
- Response Example

```
[
  {
    "health_summary": {
      "Power Supply": {
        "failed": 1,
        "active": 1
      },
      "Power Supply Fans": {
        "failed": 1,
        "active": 1
      },
      "Fans": {
        "failed": 0,
        "active": 12
      },
      "Ports": {
        "failed": 0,
        "active": 2
      }
    },
    "data_hall": null,
    "su_number": null,
    "rack": "PXX"
  }
]
```

11.4 Resources APIs

11.4.1 Get Clusters

- Description – Gets all clusters.
- Request URL – GET /cablevalidation/clusters
- Response Content Type – application/json
- Status Codes
    - 200 – OK
- Response Example

```
["default"]
```

11.4.2 Get Data Halls

- Description – Get a list of all data halls
- Request URL – GET /cablevalidation/resources/data_halls
- Response Content Type – application/json
- Status Codes
    - 200 – OK
- Response Example

```
["DH1", "DH2"]
```

11.4.3 Get Scalable Units

- Description – Get a list of scalable units.
- Request URL – GET /cablevalidation/resources/scalable_units
- Request Params
- data_hall: If set, the API will return the scalable units for the specified data hall; otherwise, it will return all scalable units in the system.
    - Type - string
    - Optional - yes
- Response Content Type – application/json
- Status Codes
    - 200 – OK
    - 404 - NOT FOUND
- Response Example

```
["SU1", "SU2"]
```

11.4.4 Get All Racks

- Description – Gets a list of rack names.
- Request URL – **GET /cablevalidation/resources/racks**
- Request Params
  - cluster: Name of the cluster. If set, the response will return the racks in that specific cluster.
    - Type - string
    - Optional - true
    - Default - default
- Response Content Type – application/json
- Status Codes
  - 200 – OK
  - 404 – NOT FOUND
- Response Example

```
["PXX", "PXH"]
```

11.4.5 Get Rack

- Description – Gets a single rack by name.
- Request URL – **GET /cablevalidation/resources/racks/<rack_name>**
- Request Params
  - cluster: Name of the cluster. If set, the response will return the rack in that specific cluster.
    - Type - string
    - Optional - true
    - Default - default
- Response Content Type – application/json
- Status Codes
  - 200 – OK
  - 404 – NOT FOUND
- Response Example


```
{
  "name": "AAA",
  "port_type": "ib",
  "dh": "DH1",
  "su": "SU1",
  "units": [{
      "unit": "35",
      "nodedesc": "swx-ray09 mlx5_0",
      "nodetype": "Switch",
      "system_info": {
        "Manufacturer": "Nvidia",
        "Product Name": "Q3400_RA",
        "Version": "V0-F*Tb-L*GcNaEi-P*PaPa-O*Tb",
        "Serial Number": "MT2421X00988"
      },
      "device_health": {
        "Power Supply":
          "Failed": 11,
          "Active": 1
        },
        "Power Supply Fans": {
          "Failed": 1,
          "Active": 1
        },
        "Fans": {
```

```
          "Failed": 0,
          "Active": 8
        }
      },
      "ports": [
        {
          "port": "sw26p1",
          "port_name": "",
          "syndrome": "Wrong-neighbor",
          "peer_port": "sw10p1",
          "peer_node": "x-spine-1"
        }]
    }],
    "rear_view":{
      "cable_cartridges":{
         "1":{
            "label":"CC#1",
            "serial":"1782724110090",
            "part_number":"750-0584-000",
            "status":"ok",
            "syndrome":null,
            "action":null,
            "issues":[

            ],
            "manufacturing_date":"08/20/24 - 05:25:00",
            "devices":{
               "gb-nvl-063-switch01":{
                  "status":"ok",
                  "syndrome":"Active",
                  "action":null,
                  "unit":"19",
                  "expected_slot":5,
                  "nodetype":"Switch",
                  "slot_id":"5",
                  "tray_id":"0",
                  "serial":"1782724110090",
                  "part_number":"750-0584-000",
                  "manufacturing_date":"08/20/24 - 05:25:00"
               }
               }
            }
         },
         "2":{
            "label":"CC#2",
            "serial":"1782824110005",
            "part_number":"750-0584-000",
            "status":"ok",
            "syndrome":null,
            "action":null,
            "issues":[

            ],
            "manufacturing_date":"08/21/24 - 03:36:00",
            "devices":{

               "gb-nvl-063-switch01":{
                  "status":"ok",
                  "syndrome":"Active",
                  "action":null,
                  "unit":"19",
                  "expected_slot":5,
                  "nodetype":"Switch",
                  "slot_id":"5",
                  "tray_id":"0",
                  "serial":"1782824110005",
                  "part_number":"750-0584-000",
                  "manufacturing_date":"08/21/24 - 03:36:00"
               }
               }
            }
         },
         "3":{
            "label":"CC#3",
            "serial":"1782724110099",
            "part_number":"750-0584-000",
            "status":"ok",
            "syndrome":null,
            "action":null,
            "issues":[

            ],
            "manufacturing_date":"08/19/24 - 02:13:00",
            "devices":{
               "gb-nvl-063-switch01":{
                  "status":"ok",
                  "syndrome":"Active",
                  "action":null,
                  "unit":"19",
                  "expected_slot":5,
                  "nodetype":"Switch",
                  "slot_id":"5",
                  "tray_id":"0",
                  "serial":"1782724110099",
                  "part_number":"750-0584-000",
                  "manufacturing_date":"08/19/24 - 02:13:00"
               }
               }
            }
         },
         "4":{
            "label":"CC#4",
            "serial":"1782824110040",
            "part_number":"750-0584-000",
            "status":"ok",
```

```
            "syndrome":null,
            "action":null,
            "issues":[

            ],
            "manufacturing_date":"08/20/24 - 15:31:00",
            "devices":{
               "gb-nvl-063-switch01":{
                  "status":"ok",
                  "syndrome":"Active",
                  "action":null,
                  "unit":"19",
                  "expected_slot":5,
                  "nodetype":"Switch",
                  "slot_id":"5",
                  "tray_id":"0",
                  "serial":"1782824110040",
                  "part_number":"750-0584-000",
                  "manufacturing_date":"08/20/24 - 15:31:00"
               }
            }
         }
      },
      "summary":{
         "status":"ok",
         "failed_cc_count":0,
         "issues_count":0
      }
   }
}
```


11.4.6 Get Nodes

- Description – Gets a single rack by name.
- Request URL – **GET /cablevalidation/resources/nodes**
- Request Params
  - cluster: Name of the cluster. If set, the response will return the nodes in that specific cluster.
    - Type - string
    - Optional - true
    - Default - default
- Response Content Type – application/json
- Status Codes
  - 200 – OK
  - 404 – NOT FOUND
- Response Example


```
[
    {
        "node_desc": "MQM8700 sw-hdr-proton01",
        "node_name": "sw-hdr-proton01",
        "type": "Switch",
        "rack": "PXX",
        "unit": 27,
        "cluster": "default",
        "su_number": null,
        "data_hall": null,
        "health_summary": {
            "Power Supply": {
                "failed": 1,
                "active": 1
            },
            "Power Supply Fans": {
                "failed": 1,
                "active": 1
            },
            "Fans": {
                "failed": 0,
                "active": 12
            },
            "Ports": {
                "failed": 0,
                "active": 2
            }
        },
        "is_managed": true,
        "system_info": {
            "Manufacturer": "Mellanox Technologies Ltd.",
            "Product Name": "MQM8700",
            "Version": "AF",
            "Serial Number": "MT2019T11025"
```

```
        },
        "asset_tag": null,
        "ip": "10.209.44.74",
        "last_update_time": 1754222504
    }]
```


11.4.7 Get Resource Utilization

- Description – Gets resource utilization.
- Request URL – **GET /cablevalidation/resources/resource_utilization**
- Response Content Type – application/json
- Status Codes
    - 200 – OK
- Response Example


```
{
    "storage": {
        "space_total": "1.8T",
        "space_used": "156G",
        "space_available": "1.5T",
        "space_percent_used": "10%",
        "type": "ext4"
    },
    "memory": {
        "sys_mem_total": "125Gi",
        "sys_mem_used": "10Gi",
        "sys_mem_free": "71Gi",
        "sys_mem_used_percent": "8.24%",
        "swap_mem_total": "15Gi",
        "swap_mem_used": "0B",
        "swap_mem_free": "15Gi",
        "swap_mem_used_percent": "0.0%"
    },
    "cpu_util": {
        "timestamp": 1743960805.0907748,
        "cpu_util_all_cores": "1.43%",
        "cpu_util_current_core": "0.33%"
    }
}
```


11.5 Filters APIs

Filter APIs work on a per-user basis, allowing each user to create their own filters.

11.5.1 Get Page Filters

- Description – Gets all filters in the given page.
- Request URL – **GET /cablevalidation/report/filters/<page>**
- Response Content Type – application/json
- Status Codes
    - 200 – OK
    - 404 – NOT FOUND
- Response Example


```
["filter1","filter2"]
```


11.5.2 Get Filter

- Description – Gets the filter by name in the specified page.
- Request URL – **GET /cablevalidation/report/filters/<page>/<filter_name>**

- Response Content Type – application/json
- Status Codes
    - 200 – OK
    - 404 – NOT FOUND
    - 500 – INTERNAL SERVER ERROR
- Response Example

```
{
  "filter_name": "filter1",
  "columns": {
    "a_endpoint_cable_pn": {
    "type": "Contains",
    "filter": "22"
    }
  }
}
```

11.5.3 Add Filter

- Description – Adds filter to the given page.
- Request URL – P **OST /cablevalidation/report/filters/<page>**
- Response Content Type – application/json
- Status Codes
    - 201 – Created
    - 400 – BAD REQUEST
    - 404 – NOT FOUND
- Request Data Example

```
{
  "filter_name":"test2",
  "columns":{
  "a_endpoint_cable_pn":{"type":"Contains","filter":"22"}
  }
}
```

- Response - 201: Created

11.5.4 Delete Filter

- Description – Deleted the given filter in the specified page.
- Request URL – **DELET /cablevalidation/report/filters/<page>/<filter_name>**
- Response Content Type – application/json
- Status Codes
    - 204 – NO CONTENT
    - 404 – NOT FOUND
- Response – No Content

11.6 Commands APIs

Commands API has been deprecated.

please see Deprecated REST API Migration Guide.

11.7 Users Management APIs

11.7.1 Get All Users

- Description – Gets all users.
- Request URL – **GET /cablevalidation/users**
- Response Content Type – application/json
- Status Codes
    - 200 – OK
    - 500 – INTERNAL SERVER ERROR
- Response Data Example

```
[
    {
        "name": "admin",
        "account_type": "admin"
    }
]
```

11.7.2 Get User

- Description – Gets the user by the given name.
- Request URL – **GET /cablevalidation/users/<user_name>**
- Response Content Type – application/json
- Status Codes
    - 200 – OK
    - 404 – NOT FOUND
- Response Data Example

```
{
"name": "admin",
"account_type": "admin"
}
```

11.7.3 Create User

- Description – Creates a user.
- Request URL – **POST /cablevalidation/users**
- Response Content Type – application/json
- Request Data Example

```
{"account_type":"cabler","password":"user1","name":"user1"}
```

- Status Codes
    - 201 – CREATED
    - 500 – INTERNAL SERVER ERROR
    - 400 – BAD REQUEST

- Response - 201: Created

11.7.4 Update User

- Description – Updates account type or password or both.
- Request URL – **PATCH /cablevalidation/users/<user_name>**
- Response Content Type – application/json
- Request Data Example

```
{"account_type":"nvidia","password":"test"}
```

- Status Codes
  - 204 – NO CONTENT
  - 500 – INTERNAL SERVER ERROR
  - 400 – BAD REQUEST
- Response - NA

11.7.5 Delete User

- Description – Deletes user by the given name.
- Request URL – **DELETE /cablevalidation/users/<user_name>**
- Response Content Type – application/json
- Status Codes
  - 204 – NO CONTENT
  - 500 – INTERNAL SERVER ERROR
- Response - NA

11.7.6 Get Account Types

- Description – Gets user account types.
- Request URL – **GET /cablevalidation/users/account_types**
- Response Content Type – application/json
- Status Codes
  - 200 – OK
  - 500 – INTERNAL SERVER ERROR
- Response Data Example

```
["nvidia", "admin", "cabler", "developer"]
```

11.8 Files Management APIs

11.8.1 Get Files

- Description – Gets the files inside a specified folder, if accessible via web.

- Request URL – **GET /cablevalidation/files_manager/<folder_name>/files?full_path=<true|false>**
- Params
  - **full_path** : if set to true the files will return with the absolute path
    - Type - bool
    - Optional - yes
- Response Content Type – application/json
- Status Codes
  - 200 – OK
  - 400 – BAD REQUEST
  - 404 – NOT FOUND
- Response Data Example

```
[
    "cable_bringup_root/data/uploads/topology/ptp/proton-ptp.xlsx",
    "cable_bringup_root/data/uploads/topology/ptp/proton-ptp-2.xlsx"
]
```

11.8.2 Upload File

- Description – Uploads the provided file to the specified folder, only if the folder is accessible via web.
- Request URL – **POST /cablevalidation/files_manager/{folder_name}/files**
- Response Content Type – application/json
- Request Data Example

```
  ------WebKitFormBoundarybxAm2tY7u5Lrhs4q
Content-Disposition: form-data; name="file"; filename="dc_layout.csv"
Content-Type: text/csv
 ------WebKitFormBoundarybxAm2tY7u5Lrhs4q--
```

- Status Codes
- 201 – CREATED
- 404 – NOT FOUND
- 400 – BAD REQUEST
- Response - 201: Created

11.8.3 Delete File

- Description – Deletes the specified file, only if it is accessible via web.
- Request URL – **DELETE /cablevalidation/files_manager/<folder_name>/files/<file_name>**
- Response Content Type – application/json
- Status Codes
  - 204 – NO CONTENT
  - 404 – NOT FOUND
  - 400 – BAD REQUEST
- Response - NA

11.9 Exclude Devices APIs

11.9.1 Get Excluded Devices

- Description – Gets the excluded devices.
- Request URL – **GET /cablevalidation/devices/excluded**
- Response Content Type – application/json
- Status Codes
    - 200 – OK

11.9.2 Exclude Devices

- Description – Excludes devices
- Request URL – **POST /cablevalidation/devices/exclude**
- Request Data Example

```
["sw-hdr-proton01", "sw-hdr-proton02"]
```

- Response Content Type – application/json
- Status Codes
    - 200 – OK
    - 400 – BAD REQUEST
- Response - OK

11.9.3 Un-exclude Devices

- Description – Unexcludes devices.
- Request URL – **DELETE /cablevalidation/devices/unexclude**
- Request Data Example

```
["sw-hdr-proton01", "sw-hdr-proton02"]
```

- Response Content Type – application/json
- Status Codes
    - 200 – OK
    - 400 – BAD REQUEST
- Response – OK

11.10 Version APIs

11.10.1 Get Version

- Description – Gets the bringup server and cables agent versions.
- Request URL – **GET /cablevalidation/version**

- Response Content Type – application/json
- Status Codes
    - 200 – OK
- Response Data Example

```
{"bringup_version": "1.3.5-28", "agent_version": "1.3.5-28"}
```

11.11 Agent Status APIs

11.11.1 Get Agents Summary

- Description – Gets the agent's summary.
- Request URL – **GET /cablevalidation/nodes/agents/summary**
- Request Params
    - cluster: Name of the cluster. If set, the response will return the agent details for that specific cluster.
        - Type - string
        - Optional - true
        - Default - default
- Response Content Type – application/json
- Status Codes
    - 200 – OK
- Response Data Example

```
{
    "total_device_count": 1,
    "correct_agent_version_installed": 1,
    "no_ping": 0,
    "no_json_api": 0,
    "no_agent": 0,
    "wrong_agent_version": 0,
    "no_reports": 0,
    "late_reports": 0,
    "stale_reports": 0,
    "expected_agent": "1.6.0-3",
    "comment": ""
}
```

11.11.2 Get Agents Details

- Description – Gets the agent's summary.
- Request URL – **GET /cablevalidation/nodes/agents/details**
- Request Params
    - cluster: Name of the cluster. If set, the response will return the agent details for that specific cluster.
        - Type - string
        - Optional - True
        - Default - default
    - context: one of the following: dc, dh, su, rack, or node.
        - Type - string
        - Optional - False

- items - The report scope, which typically includes a list of data-halls, scalable-unit/data-hall, racks, or nodes, depending on the selected context. *Do not include this item if the context is "dc".*
    - Type - comma-separated string
    - Optional - False (not included for DC context).
    - Examples:
        - **`context=su&items=DH1/SU1,DH1/SU2`**
        - **`context=dh&items=DH1,DH2`**
        - **`context=rack&items=PXH,PXX`**
- Response Content Type – application/json
- Status Codes
    - 200 – OK
    - 404 - Not Found
- Response Data Example


```
[
    {
        "hostname": "sw-hdr-proton01.mtr.labs.mlnx",
        "ip": "10.209.44.74",
        "ping": true,
        "json_api": "True",
        "agent": "1.6.0-3",
        "node_type": "Switch",
        "comment": "",
        "last_updated": 1754220100,
        "time_since_update": 3.36,
        "dh": null,
        "su": null,
        "rack": "PXX",
        "unit": 27
    }
]
```


11.12 Golden BER Test APIs

11.12.1 Get Tests

- Description – Gets golden BER tests.
- Request URL – **`GET /cablevalidation/tests/golden_ber`**
- Response Content Type – application/json
- Status Codes
    - 200 – OK
- Response Data Example


```
 [
  {
    "timestamp": 1674151365.5055714,
    "name": "Test1",
    "scope": "",
    "status": "Passed",
    "user": "admin",
    "duration_time": 3,
    "test_time": 125 ,
    "result": null,
    "number_of_failed_circuits": 0,
    "best_circuit_ber": {
      "effective_ber": 0.0,
      "raw_ber": 0.0
    },
    "worst_circuit_ber": {
      "effective_ber": 0.0,
      "raw_ber": 1e-10
    }
  }
]
```

11.12.2 Get Test Circuits

- Description – Gets golden BER test circuits.
- Request URL – **GET /cablevalidation/tests/golden_ber/<test_name>/circuits**
- Response Content Type – application/json
- Status Codes
  - 200 – OK
  - 404 – NOT FOUND
- Response Data Example


```
[
 {
   "a_endpoint": {
     "node": "sw-hdr-proton01",
     "node_type": "Switch",
     "port": "P1",
     "rack": null,
     "unit": null,
     "advanced_stats": {
       "metadata": {
         "read_time": 1727982822.4693458,
         "file_name": "/opt/ufm/cablevalidation/src/cablevalidation/cablesagent/data/amber.csv",
         "file_timestamp": 1727982822.4331076
       },
       "time_since_last_clear": 0.9,
       "power_stats": {
         "rx_power_lane_0": 1.0,
          .......
         "rx_power_lane_7": 0.82,
         "rx_power_high_th": 4.0,
         "rx_power_low_th": -5.0,
         "tx_power_lane_0": 1.27,
           …
         "tx_power_lane_7": 1.33,
         "tx_power_high_th": 4.0,
         "tx_power_low_th": -3.47
       },
       "temp_stats": {
         "module_temperature": 49.0,
         "temperature_low_th": -10.0,
         "temperature_high_th": 80.0
       },
       "ber_stats": {
         "effective_ber": 1.5e-254,
         "raw_ber": 2e-08,
         "ber_status": "Poor",
         "ber_performance": "Improved"
       },
       "carrier_transition_counter": 2,
       "ib_interface_stats": null,
       "eth_interface_stats": {
         "in_bytes": null,
         "in_drops": null,
         "in_errors": null,
         "out_bytes": null,
         "out_drops": null,
         "out_errors": null
       },
       "flapping_counters": {
         "flap_30_sec": 0,
         "flap_1_min": 1,
         "flap_5_min": 1,
         "flap_1_hr": 1,
         "flap_12_hrs": 1,
         "flap_24_hrs": 1
       }
     }
   },
   "z_endpoint": {
     .....
   }
 ]
```


11.12.3 Create Test

- Description – Creates a golden BER test.
- Request URL – **POST /cablevalidation/tests/golden_ber**
- Request Data Example

- context - dc, dh, su, rack or node
- items - The test scope, which typically includes a list of DH<n>, SU<n>/DH<n>, racks, or nodes, depending on the selected context. *Do not include this item if the context is "dc"*

```
{"name":"test","duration":5,"context":"rack","items":["AAA"]}
```

- Response Content Type – application/json
- Status Codes
  - 201 – CREATED
  - 409 – CONFLICT
- Response - 201: Created

11.12.4 Stop Test

- Description – Stops the given test.
- Request URL – **DELETE /cablevalidation/tests/golden_ber/{test_name}**
- Response Content Type – application/json
- Status Codes
  - 204 – NO CONTENT
  - 404 – NOT FOUND
- Response – NA

11.12.5 Download The Amber File For A Specific Node

- Description – Download the amber file generated by a specific Golden BER test for a specific node.
- Request URL – **GET /cv_tests/golden_ber/<test_name>/amber_<node_name>.csv.gz**
- Response Content Type – text/html
- Status Codes
  - 200 – OK
  - 404 – NOT FOUND
- Response – File

11.12.6 Download The Amber Files For The Failed Circuits

- Description – Download amber files for the failed circuits generated by the golden ber test.
- Request URL – **GET /cv_tests/golden_ber/<test_name>/<test_name>_failed_circuits_amber.tar.gz**
- Response Content Type – text/html
- Status Codes
  - 200 – OK
  - 404 – NOT FOUND
- Response – File

11.13 Amber Collection Test APIs

11.13.1 Get Tests

- Description – Gets amber tests.
- Request URL – **GET /cablevalidation/tests/amber**
- Response Content Type – application/json
- Status Codes
    - 200 – OK
- Response Data Example

```
[
  {
    "timestamp": 1674151365.5055714,
    "scope": "DH0/SU1",
    "status": "Running",
    "name": "amber1"
  },
  {
    "timestamp": 1674151365.5055714,
    "scope": "DH0/SU1",
    "status": "Finished",
    "name": "amber2"
  }
]
```

11.13.2 Get Test Details

- Description – Gets amber test details.
- Request URL – **GET /cablevalidation/tests/amber/<test_name>**
- Response Content Type – application/json
- Status Codes
    - 200 – OK
    - 404 – NOT FOUND
- Response Data Example

```
[
  {
    "hostname": "sw-hdr-proton01",
    "ip": "10.209.44.74",
    "node_type": "Switch",
    "status": "Finished",
    "dh": "DH1",
    "su": "SU2",
    "rack": "AAA",
    "unit": "1"
  }
]
```

11.13.3 Create Test

- Description – Creates amber test.
- Request URL – **POST /cablevalidation/tests/amber**
- Request Data Example
    - context - dc, dh, su, rack or node

- items - The test scope, which typically includes a list of DH<n>, SU<n>/DH<n>, racks, or nodes, depending on the selected context. *Do not include this item if the context is "dc"*

```
{"name":"test", "context":"rack", "items":["AAA"]}
```

- Response Content Type – application/json
- Status Codes
  - 201 – CREATED
  - 409 – CONFLICT
- Response - 201: Created

11.13.4 Download Amber tar File

- Description – Downloads amber tar file.
- Request URL – **GET /cv_tests/amber/<test_name>/<test_name>.tar.gz**
- Response Content Type – text/html
- Status Codes
  - 200 – OK
  - 404 – NOT FOUND
- Response - NA

11.14 Advanced Flapping Test APIs

11.14.1 Get Tests

- Description – Gets advanced flapping tests.
- Request URL – **GET /cablevalidation/tests/flapping**
- Response Content Type – application/json
- Status Codes
  - 200 – OK
- Response Data Example

```
[
    {
      "timestamp": 1674151365.5055714,
      "scope": "DataCenter",
      "status": "Finished",
      "name": "flapping1",
      "total_flapping": 5,
      "test_time": 11,
      "duration_time": 11,
      "user": "admin"
    },
    {
      "timestamp": 1674151365.5055714,
      "scope": "DataCenter",
      "status": "Finished",
      "name": "flapping2",
      "total_flapping": 5,
      "test_time": 11,
      "duration_time": 11,
      "user": "admin"
  }
]
```

11.14.2 Get Test Circuits

- Description – Gets advanced flapping test circuits.
- Request URL – **GET /cablevalidation/tests/flapping/<test_name>/circuits**
- Response Content Type – application/json
- Status Codes
    - 200 – OK
    - 404 – NOT FOUND
- Response Data Example


```
[
  {
    "a_endpoint": {
      "data_hall": "",
      "su_number": "",
      "node": "sw-hdr-proton01",
      "node_type": "Switch",
      "port": "P1",
      "rack": null,
      "unit": null,
      "advanced_stats": {
        "metadata": {
          "read_time": 1727982822.4693458,
          "file_name": "/opt/ufm/cablevalidation/src/cablevalidation/cablesagent/data/amber.csv",
          "file_timestamp": 1727982822.4331076
        },
        "time_since_last_clear": 0.9,
        "power_stats": {
          "rx_power_lane_0": 1.0,
           ...
          "tx_power_lane_7": 1.33,
          "tx_power_high_th": 4.0,
          "tx_power_low_th": -3.47
        },
        "temp_stats": {
          "module_temperature": 49.0,
          "temperature_low_th": -10.0,
          "temperature_high_th": 80.0
        },
        "ber_stats": {
          "effective_ber": 1.5e-254,
          "raw_ber": 2e-08,
          "ber_status": "Good",
          "ber_performance": "Constant"
        },
        "carrier_transition_counter": 2,
        "total_flapping": 1,
        "flapping_counters": {
          "flap_30_sec": 0,
          "flap_1_min": 1,
          "flap_5_min": 1,
          "flap_1_hr": 1,
          "flap_12_hrs": 1,
          "flap_24_hrs": 1,
          "status": "Flapped"
        }
      }
    },
    "z_endpoint": { ... }}]
```


11.14.3 Create Test

- Description – Creates advanced flapping test.
- Request URL – **POST /cablevalidation/tests/flapping**
- Response Content Type – application/json
- Request Data Example
    - context - dc, dh, su, rack or node

- items - The test scope, which typically includes a list of DH<n>, SU<n>/DH<n>, racks, or nodes, depending on the selected context. *Do not include this item if the context is "dc"*

```
{"context":"rack","items":["AAA"],"name":"test","duration":125}
```

- Status Codes
  - 201 – CREATED
  - 409 – CONFLICT
- Response - 201: Created

11.14.4 Stop Advanced Flapping Test

- Description – Stop the given test.
- Request URL – **DELETE /cablevalidation/tests/flapping/<test_name>**
- Response Content Type – application/json
- Status Codes
- 204 – NO CONTENT
- 404 – NOT FOUND
- Response – NA

11.15 Cable Validation Controller APIs

11.15.1 Start Bringup

- Description – Starts the bringup.
- Request URL – **POST /cv_controller/start**
- Response Content Type – application/json
- Status Codes
  - 200 – OK
  - 404 – NOT FOUND
- Response - 200: OK

11.15.2 Stop Bringup

- Description – Stops bringup.
- Request URL – **POST /cv_controller/stop**
- Response Content Type – application/json
- Status Codes
  - 200 – OK
  - 404 – NOT FOUND
- Response - 200: OK

11.16 Credential APIs

11.16.1 Get Devices Credentials

- Description – Gets devices' credentials.
- Request URL – **GET /cablevalidation/creds**
- Response Content Type – application/json
- Status Codes
    - 200 – OK
- Response Data:

```
[
    {
        "node": "default",
        "user": "admin",
        "pwd": "****",
        "type": "switch",
        "save": true
    },
    {
        "node": "sw-hdr-proton01.mtr.labs.mlnx",
        "user": "admin",
        "pwd": "****",
        "type": "",
        "save": false
    }
]
```

11.16.2 Upload Credentials File

11.16.3 Description – Uploads and loads credentials from a credential file (JSON or INI format).

Request URL – POST /cablevalidation/creds/upload

Request Content Type – multipart/form-data

Form Fields:

- file (required) – The credential file (format auto-detected from filename)
- replace (optional) – Whether to replace existing credentials: 'true' or 'false' (default: false)

Response Content Type – application/json

Status Codes:

- 200 – OK
- 400 – Bad Request (invalid file content, unsupported format, or failed to apply credentials)
-

```
{
    "total": 10,
    "success": 9,
    "failed": 1,
    "errors": ["Failed to apply credential_profile..."]
```

```
}
```

11.16.4 Set Node Credentials

Description – Sets credentials for a specific node.

Request URL – POST /cablevalidation/creds/nodes/{node_id}

Parameters:

- node_id – Node identifier (hostname or IP address)

Request Content Type – application/jsonRequest

Request Body:

-
```
{
    "user": "admin",
    "pwd": "password",
    "save": true
}
```

Response Content Type – application/json

Status Codes:

- 200 – OK
- 400 – Bad Request (missing required fields or invalid JSON)

Response Body :

-
```
{
    "message": "Credentials set for node node_id"
}
```

11.16.5 Delete Node Credentials

Description – Removes credentials for a specific node.

Request URL – DELETE /cablevalidation/creds/nodes/{node_id}

Parameters:

- node_id – Node identifier (hostname or IP address)

Response Content Type – application/json

Status Codes:

- 200 – OK

- ```
  {
      "message": "Credentials removed for node sw-hdr-proton01.mtr.labs.mlnx"
  }
  ```

11.16.6 Set Default Credentials

Description – Sets default credentials for a node type (switch or host).

Request URL – POST /cablevalidation/creds/defaults/{node_type}

Parameters:

- node_type – Node type: switch or host

Request Content Type – application/json

Status Codes:

- 200 – OK
- 400 – Bad Request (missing required fields, invalid node_type, or invalid JSON)

Response Data:

- ```
  {
          "message": "Default credentials set for switch"
  }
  ```

11.16.7 Delete Default Credentials

Description – Removes default credentials for a node type.

Request URL – DELETE /cablevalidation/creds/defaults/{node_type}

Parameters:

- node_type – Node type: switch or host

Response Content Type – application/json

Status Codes:

- 200 – OK
- 400 – Bad Request (invalid node_type)

- ```
  {
      "message": "Default credentials removed for switch"
  }
  ```

11.16.8 Set Profile Credentials

Description – Sets credentials for a credential profile.

Request URL – POST /cablevalidation/creds/profiles/{profile_name}
  ```

Parameters:

- profile_name – Name of the credential profile

Request Content Type – application/json

Request Body:

- ```
  {
      "user": "admin",
      "pwd": "password",
      "save": true
  }
  ```

Response Content Type – application/json

Status Codes:

- 200 – OK
- 400 – Bad Request (missing required fields or invalid JSON)
- 404 – Not Found (profile not found

Response Data:

- ```
  {
      "message": "Credentials set for profile profile1"
  }
  ```

11.16.9 Delete Profile Credentials

Description – Removes credentials for a credential profile.

Request URL – DELETE /cablevalidation/creds/profiles/{profile_name}

Parameters:

- profile_name – Name of the credential profile

Response Content Type – application/json

Status Codes:

- 200 – OK

Response Data:

- ```
  {
      "message": "Credentials removed for profile profile1"

  }
  ```

11.17 Help API

- Description – Print supported Rest APIs.
- Request URL – GET /cablevalidation/help
  ```

- Response Content Type – text/html
- Status Codes
- 200 – OK
- Response Data:

```
Bringup server REST API

Command                          Method    Description
/images/{image_name}     GET          For agent use: return the agent docker image file.
/report/validation             GET         return the recent cable validation report.
/clusters                            GET          Get all clusters
/report/summary              GET          Get all clusters summary or a single cluster summary by name
/report/circuits                GET          Get circuits information
```

11.18 Amber Simulation APIs

11.18.1 Get Amber Simulation

- Description – Gets amber simulation.
- Request URL – **GET /cablevalidation/simulation/amber**
- Request Params:

Parameter	Description	Type	Optional/Required
node_name	Node name. If set, gets a simulation info for that node	string	Optional

- Response Content Type – application/json
- Status Codes
  - 200 – OK
- Response Data Example

```
[{"node_name": "sw-hdr-proton01", "device": null}]
```

11.18.2 Load Amber Simulation

- Description – Load amber simulation.
- Request URL – **POST /cablevalidation/simulation/amber/load**
- Request Data

Parameter	Description	Type	Optional/Required
node_name	Node name	string	Required
device	Device's name, if the node contains multiple devices	string	Optional
file	The binary file	Binary	Required

- Response Content Type – application/json

- Status Codes
    - 201 – CREATED
    - 400 - Bad Request
    - 409 - CONFLICT
- Response Data Example

```
{
    "status": "success",
    "node_name": "sw-hdr-proton01",
    "saved_file": "/tmp/amber_simulation_sw-hdr-proton01_device1_1754227317.csv",
    "size_bytes": 185578
}
```

11.18.3 Stop Amber Simulation

- Description – Load amber simulation.
- Request URL – **DELETE /cablevalidation/simulation/amber/stop**
- Request Parameters:

Parameter	Description	Type	Optional/Required
node_name	Node name	string	Required
device	Device's name, if the node contains multiple devices	string	Optional

- Response Content Type – no content
- Status Codes
    - 204 – NO CONTENT
    - 400 - BAD REQUEST
    - 404 - NOT FOUND

11.19 Prometheus Endpoint

11.19.1 Cable Validation Tool - Prometheus Metrics Endpoint

11.19.1.1 Overview

The Cable Validation Tool (CVT) now provides a Prometheus-compatible metrics endpoint that exposes real-time cable health and performance data for monitoring and alerting. This endpoint enables integration with modern monitoring stacks like Prometheus, Grafana, and other observability tools.

11.19.1.2 Features

11.19.1.2.1 🎯 Key Capabilities

- Real-time Metrics: Live cable validation data from network switches and hosts

- **Multi-format Support**: Prometheus, JSON, and CSV output formats
- **Rich Labeling**: Complete network topology context with peer relationships
- **High Performance**: Multi-level caching optimized for frequent scraping
- **Production Ready**: Handles 100K+ ports with memory-adaptive optimizations

11.19.1.2.2 📊 Metrics Categories

- **Power Metrics**: RX/TX optical power per lane (up to 8 lanes per port)
- **BER Metrics**: Effective and Raw Bit Error Rates
- **Temperature Metrics**: Module temperature and thresholds
- **Counter Metrics**: Transceiver reinsert/swap events
- **Validation Metrics**: Port validation status with issue descriptions
- **Threshold Metrics**: Power and temperature alarm thresholds
- **Timestamp Metrics**: Data collection and report timestamps

11.19.1.3 API Endpoints

11.19.1.3.1 Base URL

`https://<cvt-server>/cablevalidation/metrics`

11.19.1.3.2 Available Endpoints

Endpoint	Format	Content-Type	Description
`/cablevalidation/metrics`	Prometheus	`text/plain`	Standard Prometheus exposition format
`/cablevalidation/metrics/json`	JSON	`application/json`	Structured JSON for programmatic access
`/cablevalidation/metrics/csv`	CSV	`text/plain`	Comma-separated values for spreadsheet import

11.19.1.3.3 Authentication

- **No authentication required** for metrics endpoints
- **HTTPS enforced** for security
- **Bypasses session handling** for automated scraping

11.19.1.4 Sample Output

11.19.1.4.1 Prometheus Format

```
# HELP effective_ber Effective Bit Error Rate
# TYPE effective_ber gauge
# HELP validation_status Port validation status with issue
descriptions in labels (value = issue count)
# TYPE validation_status gauge
# HELP port_info Port information with status and validation details
```

```
in labels
# TYPE port_info gauge

# Healthy port with performance metrics
effective_ber{node="ufm-host38",port="enp3s0f0np0",peer_node="r-ufm-
sw-
eth01",peer_port="swp2",node_type="Host",su_number="SU1",data_hall="D
H1"} 1.5e-254 1759345924622
module_temperature{node="ufm-host38",port="enp3s0f0np0",peer_node="r-
ufm-sw-
eth01",peer_port="swp2",node_type="Host",su_number="SU1",data_hall="D
H1"} 65.2 1759345924622
validation_status{node="ufm-host38",port="enp3s0f0np0",peer_node="r-
ufm-sw-
eth01",peer_port="swp2",node_type="Host",su_number="SU1",data_hall="D
H1"} 0 1759345924622

# Unplugged port with validation issue (power/temp metrics excluded
due to NA values)
validation_status{node="ufm-host38",port="enp3s0f1np1",peer_node="r-
ufm-sw-
eth01",peer_port="swp3",node_type="Host",su_number="SU1",data_hall="D
H1",MediaUnplugged="Insert; Reseat or Replace Cable/Transceiver"} 1
1759345923752
effective_ber{node="ufm-host38",port="enp3s0f1np1",peer_node="r-ufm-
sw-
eth01",peer_port="swp3",node_type="Host",su_number="SU1",data_hall="D
H1"} 0.0 1759345923752
time_since_last_clear{node="ufm-
host38",port="enp3s0f1np1",peer_node="r-ufm-sw-
eth01",peer_port="swp3",node_type="Host",su_number="SU1",data_hall="D
H1"} 320035.6 1759345923752

# Port info with detailed status
port_info{node="ufm-host38",port="enp3s0f1np1",peer_node="r-ufm-sw-
eth01",peer_port="swp3",node_type="Host",su_number="SU1",data_hall="D
H1",phy_manager_state="Disable",module_oper_status="unplugged",cable_
sn="N/A",cable_pn="N/A",protocol="Ethernet",module_fw_version="N/A"}
1 1759345923752
```

11.19.1.4.2 JSON Format

```
{
  "ufm-host38:enp3s0f0np0": {
    "timestamp": 1757524769.645,
    "port_info": {
      "node_name": "ufm-host38",
```

```
      "port_name": "enp3s0f0np0",
      "peer_node_name": "r-ufm-sw-eth01",
      "peer_port_name": "swp2",
      "node_type": "Host",
      "su_number": "SU1",
      "data_hall": "DH1"
    },
    "port_labels": {
      "cable_sn": "ABC123",
      "cable_pn": "DEF456",
      "protocol": "400G"
    },
    "port_stats": {
      "effective_ber": 1.5e-254,
      "module_temperature": 65.2,
      "rx_power_lane_0": -2.5
    },
    "validation_data": {
      "issues_count": 1,
      "last_report_time": 1757524769.645,
      "issues": {
        "WrongNeighbor": "Check cable connection to switch2"
      }
    }
  }
}
```


11.19.1.5 Metrics Reference

11.19.1.5.1 Power Metrics

Metric	Type	Description	Unit
`rx_power_lane_N`	gauge	RX optical power for lane N (0-7, not all lanes may be present)	dBm
`tx_power_lane_N`	gauge	TX optical power for lane N (0-7, not all lanes may be present)	dBm
`rx_power_high_th`	gauge	RX power high threshold	dBm
`rx_power_low_th`	gauge	RX power low threshold	dBm

11.19.1.5.2 BER Metrics

Metric	Type	Description
`effective_ber`	gauge	Effective Bit Error Rate
`raw_ber`	gauge	Raw Bit Error Rate

11.19.1.5.3 Temperature Metrics

Metric	Type	Description	Unit
`module_temperature`	gauge	Current module temperature	Celsius
`temperature_high_th`	gauge	Temperature high threshold	Celsius
`temperature_low_th`	gauge	Temperature low threshold	Celsius

11.19.1.5.4 Status Metrics

Metric	Type	Description	Values
`port_status`	gauge	Port plugged status	1=Up, 0=Down
`port_oper_status`	gauge	Port operational status	1=Up, 0=Down

11.19.1.5.5 Counter Metrics

Metric	Type	Description
`transceiver_reinsert_cnt`	counter	Number of transceiver reinsert events
`transceiver_swap_cnt`	counter	Number of transceiver swap events
`time_since_last_clear`	gauge	Time since last counter clear (seconds)

11.19.1.5.6 Validation Metrics

Metric	Type	Description	Special Features
`validation_status`	gauge	Port validation status with issue descriptions	**Value = issue count, descriptions in labels**
`last_report_time`	gauge	Timestamp of last validation report	Unix timestamp

11.19.1.5.6.1 Validation Status Labels

The `validation_status` metric includes dynamic labels for each type of validation issue:

- `WrongNeighbor` : "Check cable connection to correct switch"
- `MediaUnplugged` : "Insert; Reseat or Replace Cable/Transceiver"
- `AnomalousPort` : "Temperature exceeds threshold"
- `FlappingLink` : "Reseat transceiver; Check Fiber"
- `UnknownNeighbor` : "Verify neighbor device connectivity"
- `WrongPort` : "Check port mapping in topology"

- **ExtraCable** : "Remove unexpected cable connection"
- **UnreachableDevice** : "Check device connectivity and power"
- **LinkDown_NoSignal** : "Check physical connection"
- **ErrDisable_Flap** : "Port disabled due to flapping"
- **AdminDown** : "Port administratively disabled"
- **ErrDisable_Rx** : "RX error disable condition"
- **NegotiationFail** : "Check autonegotiation settings"
- **NicNameMismatch** : "Verify NIC provisioning"
- **ModulePnMismatch** : "Replace with compatible module"

Note: Commas in descriptions are automatically converted to semicolons to maintain Prometheus label format compatibility.

11.19.1.5.6.2 Examples:

```
# Port with validation issues (unplugged cable)
validation_status{node="ufm-host38",port="enp3s0f1np1",peer_node="r-
ufm-sw-
eth01",peer_port="swp3",node_type="Host",su_number="SU1",data_hall="D
H1",MediaUnplugged="Insert; Reseat or Replace Cable/Transceiver"} 1

# Port without issues (healthy connection)
validation_status{node="ufm-host38",port="enp3s0f0np0",peer_node="r-
ufm-sw-
eth01",peer_port="swp2",node_type="Host",su_number="SU1",data_hall="D
H1"} 0

# Port with multiple validation issues
validation_status{node="switch1",port="1/1",WrongNeighbor="Check
cable connection to switch2",AnomalousPort="Temperature exceeds
threshold"} 2
```

11.19.1.6 Labels

11.19.1.6.1 Topology Labels (All Metrics)

- **node** : Switch or host name
- **port** : Port identifier
- **peer_node** : Connected peer node name
- **peer_port** : Connected peer port identifier
- **node_type** : Node type (Switch, Host, etc.)
- **su_number** : Scalable Unit identifier
- **data_hall** : Data hall location

11.19.1.6.2 Cable Labels (Status Metrics Only)

- `cable_sn` : Cable serial number
- `cable_pn` : Cable part number
- `protocol` : Cable protocol (400G, InfiniBand, etc.)
- `port_status` : Port status (Up, Down, etc.)
- `plugged` : Module plugged status

11.19.1.7 Performance Characteristics

11.19.1.7.1 Update Frequency

- Agent Data: Updated every 10 minutes (configurable)
- Metrics Cache: Invalidated on data changes
- Prometheus Scraping: Recommended 15-30 second intervals

11.19.1.7.2 Performance Metrics

Deployment Size	Response Time	Memory Usage	Caching Strategy
< 10K ports	< 50ms	~20MB	Full caching enabled
10K-50K ports	< 200ms	~100MB	Collection cache only
50K+ ports	< 500ms	~200MB	No caching, real-time generation

11.19.1.7.3 Optimization Features

- Multi-level Caching: Port, collection, and label caching
- Memory Adaptive: Automatically adjusts for large deployments
- Smart Change Detection: Only updates when cable/module data changes
- Zero Value Handling: Includes all values for complete visibility

11.19.1.8 Troubleshooting

11.19.1.8.1 Common Issues

11.19.1.8.1.1 1. No Metrics Data

Symptoms: Empty response or no metrics Causes:

- CVT service not running
- No topology loaded
- No advanced stats collection

Solutions:

```
# Check service status
```

```
# Check topology loading

# Check agent connectivity
```

11.19.1.8.1.2 2. Missing Port Data

Symptoms: Some ports not appearing in metrics Causes:

- Port not in loaded topology
- Agent not deployed on switch
- Advanced stats not collected

Solutions:

- Verify topology includes all expected ports
- Deploy agents on missing switches
- Check agent connectivity and data collection

11.19.1.8.1.3 3. Stale Timestamps

Symptoms: Old timestamps in metrics Causes:

- Agent not sending updates
- Network connectivity issues

Solutions:

- Check agent logs for errors
- Verify network connectivity to switches
- Restart agents if necessary

11.19.1.8.1.4 4. Missing Validation Data

Symptoms: validation_status metrics missing or always 0 Causes:

- Validation reports not being generated
- Agent data filtering (switch not in topology)
- Report processing errors

Solutions:

- Verify validation is started on agents
- Check switch IP exists in topology
- Review agent and collector logs for errors

11.19.1.8.1.5 5. Inconsistent Issue Counts

Symptoms: validation_status count doesn't match expected issues Causes:

- Issues filtered by port
- Report data synchronization issues
- Processing errors

Solutions:

- Check that report data includes port-specific issues
- Verify advanced stats and reports arrive together
- Review validation report structure

11.19.1.8.2 Performance Tuning

11.19.1.8.2.1 Environment Variables

TBD: not supported yet.

```
# Adjust caching thresholds
PROMETHEUS_MAX_CACHED_PORTS=10000

# Disable detailed metrics for very large deployments
PROMETHEUS_ENABLE_DETAILED_METRICS=false

# Adjust cache TTL
PROMETHEUS_CACHE_TTL=60
```

11.19.1.8.2.2 Memory Optimization

For deployments > 50K ports:

- Collection-level caching automatically disabled
- Port-level caching automatically disabled
- Real-time generation used (acceptable 200-500ms response time)

11.19.1.9 Security Considerations

11.19.1.9.1 Access Control

- HTTPS Required: All access must be over HTTPS
- No Authentication: Designed for automated monitoring tools
- Network Restrictions: Consider IP-based access control

11.19.1.9.2 Sensitive Data

- Network Topology: Metrics expose network structure
- Cable Information: Serial numbers and part numbers included
- Performance Data: Could reveal network capacity information

11.19.1.9.3 Recommended Security

```
<Location /cablevalidation/metrics>
    Use BringupProxy
    SSLRequireSSL

    # Restrict to monitoring networks
    <RequireAll>
```

```
        Require ip 10.0.0.0/8      # Internal networks
        Require ip 172.16.0.0/12   # Container networks
        Require ip 192.168.0.0/16  # Private networks
    </RequireAll>
</Location>
```

11.19.1.10 Integration Examples

11.19.1.10.1 Python Client Example

```
import requests
import json

# Get metrics in different formats
def get_cvt_metrics(server: str, port: int,
response_format='prometheus'):
    endpoints = {
        'prometheus': '/cablevalidation/metrics',
        'json': '/cablevalidation/metrics/json',
        'csv': '/cablevalidation/metrics/csv'
    }
    url = f"https://{server}:{port}{endpoints[format]}"
    response = requests.get(url, verify=False)

    if format == 'json':
        return response.json()
    return response.text

# Usage
metrics = get_cvt_metrics('cvt-server.example.com', 'json')
for port_key, port_data in metrics.items():
    if port_data['port_stats']['effective_ber'] > 1e-12:
        print(f"High BER on {port_key}: {port_data['port_stats']
['effective_ber']}")
```

11.19.1.10.2 Validation Monitoring Examples

```
# Find all ports with validation issues
validation_status > 0

# Count issues by syndrome type
sum by (node) (validation_status{WrongNeighbor!=""})
sum by (node) (validation_status{MediaUnplugged!=""})
sum by (node) (validation_status{LinkDown_NoSignal!=""})

# Find specific issue types
validation_status{MediaUnplugged!=""} > 0  # Unplugged cables
```

```
validation_status{AdminDown!=""} > 0       # Administratively
disabled ports
validation_status{ModulePnMismatch!=""} > 0 # Hardware compatibility
issues

# Ports with multiple issue types
validation_status{WrongNeighbor!="",AnomalousPort!=""}

# Correlation with performance metrics
(validation_status > 0) and (effective_ber > 1e-12)

# Port status correlation
validation_status{MediaUnplugged!=""} and on()
port_info{module_oper_status="unplugged"}
```

11.19.1.11 Architecture

11.19.1.11.1 Data Flow

```
Network Agents → CVT Collector → Advanced Stats + Report Data → Prometheus
Collector → Metrics Endpoint
    ↓                ↓                          ↓                          ↓
                          ↓
(10 min)   (Real-time)   (Synchronized)      (Multi-level Cache)  (GET request)
```

11.19.1.11.1.1 Enhanced Data Processing

1. Agent Data Validation: Switch IP validated against topology before processing
2. Synchronized Processing: Advanced stats and validation reports processed together
3. Optimized Issue Processing: Report data pre-processed to group issues by port (O(n+m) complexity)
4. Independent Validation Cache: PortValidationStatus class with hash-based change detection
5. Robust Syndrome Handling: Automatic fallback for unknown syndromes with developer warnings
6. Smart Data Quality: NA values properly excluded, counters preserve semantics

11.19.1.11.2 Caching Strategy

1. Port-level Cache: Individual port metrics cached until data changes
2. Collection-level Cache: Aggregated output cached for fast retrieval
3. Label Cache: Stable topology/cable labels cached separately
4. Validation Cache: Independent cache for validation status with hash-based change detection
5. Metadata Cache: Static TYPE/HELP comments cached permanently

11.19.1.11.3 Performance Optimizations

- Push-based Updates: Metrics updated when advanced stats arrive
- Smart Change Detection: Only cable/module changes invalidate caches
- Memory Adaptive: Caching disabled automatically for large deployments
- String Manipulation: Efficient JSON aggregation using string operations
- Validation Processing: O(n+m) complexity with report preprocessing
- Hash-based Cache: Validation cache only invalidated when issue content changes
- Agent Data Filtering: Invalid switch IPs filtered early to prevent unnecessary processing

11.19.1.12 Monitoring Best Practices

11.19.1.12.1 Prometheus Configuration

- Scrape Interval: 15-30 seconds (matches CVT data update frequency)
- Timeout: 10 seconds (allows for cache generation)
- Retention: Configure based on historical analysis needs

11.19.1.12.2 Alerting Guidelines

- BER Thresholds: Alert when effective_ber > 1e-12
- Temperature Limits: Alert when module_temperature approaches temperature_high_th
- Validation Issues: Alert when validation_status > 0
- Critical Issues: Alert on specific syndromes (MediaUnplugged, UnreachableDevice, ModulePnMismatch)
- Infrastructure Issues: Alert on LinkDown_NoSignal, ErrDisable conditions
- Counter Anomalies: Alert on rapid increases in transceiver_reinsert_cnt

11.19.1.12.2.1 Sample Alerting Rules

```
# Validation issues alert
- alert: PortValidationIssues
  expr: validation_status > 0
  labels:
    severity: warning
  annotations:
    summary: "Port {{ $labels.node }}:{{ $labels.port }} has
validation issues"
    description: "{{ $value }} validation issues detected"

# Critical validation issues
- alert: CriticalPortIssues
  expr: validation_status{MediaUnplugged!=""} > 0 or
validation_status{UnreachableDevice!=""} > 0
  labels:
    severity: critical
  annotations:
```

```
    summary: "Critical issues on {{ $labels.node }}:
{{ $labels.port }}"
    description: "{{ if $labels.MediaUnplugged }}Cable unplugged:
{{ $labels.MediaUnplugged }}{{ end }}{{ if
$labels.UnreachableDevice }}Device unreachable:
{{ $labels.UnreachableDevice }}{{ end }}"

# Infrastructure issues
- alert: InfrastructureIssues
  expr: validation_status{LinkDown_NoSignal!=""} > 0 or
validation_status{ModulePnMismatch!=""} > 0
  labels:
    severity: warning
  annotations:
    summary: "Infrastructure issue on {{ $labels.node }}:
{{ $labels.port }}"

# Administrative issues
- alert: AdminIssues
  expr: validation_status{AdminDown!=""} > 0 or
validation_status{NicNameMismatch!=""} > 0
  labels:
    severity: info
  annotations:
    summary: "Administrative issue on {{ $labels.node }}:
{{ $labels.port }}"
```

11.19.1.13 Version Information

11.19.1.13.1 Release Notes

- Version: 1.1.0
- Release Date: October 2025
- Compatibility: CVT 1.7.0 and later
- Dependencies: Requires advanced stats collection enabled

11.19.1.13.1.1 New in Version 1.1.0

- ✅ Validation Metrics Integration: Port validation status with actionable issue descriptions
- ✅ Synchronized Data Processing: Advanced stats and validation reports processed together
- ✅ Performance Optimizations: O(n+m) validation processing, hash-based change detection
- ✅ Enhanced Security: Agent data validation prevents processing from unknown switches
- ✅ Improved Data Quality: None-based initialization for gauges, proper counter semantics
- ✅ Better Caching: Independent validation cache with content-based invalidation
- ✅ Comprehensive Syndrome Coverage: 15+ validation issue types with fallback handling

- ✅ **Real-world Validation**: Successfully tested with production data and unplugged ports

11.19.1.13.2 API Stability

- **Metric Names**: Stable (no breaking changes planned)
- **Label Names**: Stable (additions possible, no removals)
- **Output Format**: Prometheus standard compliance maintained
- **Endpoint URLs**: Stable API contract

11.19.1.14 Data Quality Improvements

11.19.1.14.1 Enhanced Counter Semantics

- **Gauges default to None**: Missing sensor data excluded instead of showing false zeros
- **Counters preserve values**: No unexpected resets when data temporarily unavailable
- **Proper NA handling**: Invalid data marked as NA and excluded from metrics
- **Temperature accuracy**: Fixed zero temperature issue by using actual amber timestamps

11.19.1.14.2 Validation Integration Benefits

- **Synchronized processing**: Performance metrics and validation issues always in sync
- **Rich context**: Issue descriptions provide actionable corrective actions
- **Efficient processing**: O(n+m) complexity prevents performance degradation
- **Smart caching**: Validation cache independent of performance metrics cache

11.19.1.15 Support

11.19.1.15.1 Troubleshooting

1. **Check CVT Service**: Ensure Cable Validation service is running
2. **Verify Topology**: Confirm network topology is loaded
3. **Agent Status**: Check that agents are deployed and collecting data
4. **Network Connectivity**: Verify switch/host accessibility

11.19.1.15.2 Performance Monitoring

```
# Check metrics endpoint response time
time curl -k https://cvt-server/cablevalidation/metrics > /dev/null
```

11.19.1.15.3 Contact Information

- **Development Team**: Cable Validation Engineering
- **Documentation**: [Internal Wiki Link]
- **Support**: [Support Channel/Email]

This endpoint provides comprehensive cable validation metrics for modern monitoring and observability workflows, enabling proactive network health management and automated alerting.

11.20 Logs APIs

11.20.1 Introduction

This document describes how log viewing works in the UI: which backend APIs are used, what options you can pass (including lines / tail size), and where to open each view.
All log endpoints are served by the collector HTTP API and require authentication (same session as the rest of the app).

11.20.2 API Requests

Paths below are relative to the collector API root : https://<CVT_IP>/cablevalidation/

11.20.2.1 Collector Logs API:

Method	Path	Purpose/Output
`GET`	`/logs/collector`	JSON array of allowed `log_type` keys.
`GET`	`/logs/collector/{log_type}/tail`	Last **N** lines as **plain text** (`text/plain`).
`GET`	`/logs/collector/{log_type}/download`	Full collector log file is downloaded from browser.

11.20.2.1.1 Query Parameter

Parameter	Applies to	Description
`lines`	`.../tail` only	Number of lines from end of file. Default **100**, max **1000**. For lines more than 1000 user will have to download the whole file.

11.20.2.1.2 log_type values

`log_type`	Log name (under <cable_bringup_root>/log/)
bringup_collector	`bringup_collector.log`
agents	`bringup_agents.log`
apache2_access	`apache2/access.log`
cv_controller_service	`cv_controller_service.log`
cvt_service	`cvt-service.log`
supervisord	`supervisor/supervisord.log`

11.20.2.1.3 Example

```
GET /logs/collector/
GET /logs/collector/bringup_collector/tail?lines=500
GET /logs/collector/cv_controller_service/download
```

11.20.2.2 Agent Logs API

There are two log types per agent, both under `<cable_bringup_root>/log/agents/<hostname>/` on the collector: Agent log — ongoing runtime and cable-validation output (`<hostname>.log`); Deployment log — output from deploy / remove agent operations on that node (`deployment.log`).

Method	Path	Purpose/Output
`GET`	`/logs/agents`	JSON list of agent names that have log directories. Each node will have a directory by its hostname.
`GET`	`/logs/agents/{agent_name}/tail`	Last **N** lines of the **agent** log (`text/plain`).
`GET`	`/logs/agents/{agent_name}/download`	Full agent log file is downloaded from browser.
`GET`	`/logs/agents/{agent_name}/deployment/tail`	Last **N** lines of **deployment.log** (`text/plain`).
`GET`	`/logs/agents/{agent_name}/deployment/download`	Full agent deployment log file is downloaded from browser.

11.20.2.2.1 Query Parameter

Parameter	Applies to	Description
`lines`	`.../tail` only (agent and deployment)	Number of lines from end of file. Default **100**, max **1000**. For lines more than 1000 user will have to download the whole file.

11.20.2.2.2 Example

```
GET /logs/agents/sw-hdr-proton01/tail?lines=300
GET /logs/agents/sw-hdr-proton01/deployment/tail?lines=100
GET /logs/agents/sw-hdr-proton01/download
```

11.21 Deprecated REST API Migration Guide

11.21.1 1. Deprecated REST API Migration Guide

11.21.1.1 1.1. Overview

Several legacy REST API endpoints have been removed from the Cable Validation Tool (CVT) and replaced with dedicated, purpose-built endpoints. This guide helps users migrate from the deprecated APIs to their replacements.

The changes fall into two categories:

1. CommandProc endpoints (`/commands/*`) — a generic command-execution router that has been replaced by dedicated REST endpoints for each feature area.
2. Device exclude/unexclude endpoints (`/devices/exclude`, `/devices/unexclude`, `/devices/excluded`) — replaced by the unified Device Status Management API.

11.21.1.2 1.2. CommandProc Endpoints

The legacy `/commands/*` API allowed callers to execute CLI commands over HTTP through a single generic router. This has been retired in favor of dedicated REST endpoints that follow standard HTTP conventions (proper methods, structured request/response bodies, and per-operation status polling).

11.21.1.2.1 1.2.1. 1 Generic Routes (Removed)

These meta-endpoints provided command discovery, execution status, and help. They have no direct replacements because each feature area now exposes its own endpoints.

Deprecated Endpoint	Method	Description	Replacement
`/commands`	`GET`	List all supported commands	No replacement. Available actions are expressed by the individual REST endpoints listed below.
`/commands/status`	`GET`	Global command execution status (idle / executing)	No replacement.
`/commands/{command_name}`	`POST`	Execute a command by name	See per-command mapping below.
`/commands/{command_name}/output`	`GET`	Poll the output of a running command	Each feature now has its own result-polling endpoint (e.g. `GET /validation/result`, `GET /agents/manage/result`, `GET /topology/action/result`).
`/commands/{command_name}/help`	`GET`	Get help text for a command	No replacement.

11.21.1.2.2 1.2.2. 2 Topology Loading

All topology loading commands now use the `POST /topology/load/{type}` pattern. These are asynchronous — the server returns `202 Accepted` immediately and progress is tracked by polling `GET /topology/action/result`.

Deprecated Command	Replacement Endpoint	Method	Request Body
`load_topo`	`/topology/load/topo`	`POST`	`{"files": [...], "dns": true}`
`load_ip`	`/topology/load/ip`	`POST`	`{"files": [...]}`
`load_ptp`	`/topology/load/ptp`	`POST`	`{"file": "...", "format": "...", "sheets": [...], "dns": true, "dc_layout": "...", "hca_mapping": "..."}`
`load`	No replacement	—	The `load` command loaded from a saved file prefix. JSON topology can be loaded via `POST /topology/load/json` instead.

Polling: `GET /topology/action/result` (also `GET /topology/load/result` for backward compatibility).

11.21.1.2.3 1.2.3. 3 Credential Management

Credential commands are replaced by RESTful endpoints under `/creds/*`. All are synchronous (immediate response).

Deprecated Command	Replacement Endpoint	Method	Request Body
`set_default_creds`	`/creds/defaults/{node_type}`	`POST`	`{"user": "...", "pwd": "...", "save": true}` where `node_type` is `switch` or `host`
`set_node_creds`	`/creds/nodes/{node_id}`	`POST`	`{"user": "...", "pwd": "...", "save": true}`
`set_credential_profile`	`/creds/profiles/{profile_name}`	`POST`	`{"user": "...", "pwd": "...", "save": true}`
`remove_node_creds`	`/creds/nodes/{node_id}`	`DELETE`	—
`remove_credential_profile`	`/creds/profiles/{profile_name}`	`DELETE`	—

Additional credential endpoints:

- `GET /creds` — list all credentials
- `GET /creds?credential_profile=true` — list all credential profiles
- `POST /creds/upload` — upload a credentials file (multipart form)

11.21.1.2.4 1.2.4. 4 User Management

Deprecated Command	Replacement Endpoint	Method	Notes
`load_web_users`	`/users/upload`	`POST`	Multipart form with a `file` field (JSON) and an optional `delete_existing_users` field. Synchronous.

11.21.1.2.5 1.2.5. 5 Agent Deployment

The three separate deploy commands (`deploy_all_agents`, `deploy_single_agent`, `deploy_agents`) are unified into a single `POST /agents/deploy` endpoint. Similarly, the three remove commands are unified into `POST /agents/remove`. Both return `202 Accepted` and are polled via `GET /agents/manage/result`.

Deprecated Command	Replacement Endpoint	Method	Request Body
`deploy_all_agents`	`/agents/deploy`	`POST`	`{"all": true}`
`deploy_single_agent`	`/agents/deploy`	`POST`	`{"switches": ["<switch>"]}`
`deploy_agents`	`/agents/deploy`	`POST`	`{"switches": ["<switch1>", "<switch2>", ...]}`
`remove_all_agents`	`/agents/remove`	`POST`	`{"all": true}`
`remove_single_agent`	`/agents/remove`	`POST`	`{"switches": ["<switch>"]}`
`remove_agents`	`/agents/remove`	`POST`	`{"switches": ["<switch1>", "<switch2>", ...]}`

Polling: `GET /agents/manage/result`

11.21.1.2.6 1.2.6. 6 Validation

Deprecated Command	Replacement Endpoint	Method	Request Body
`start_validation`	`/validation`	`POST`	`{"timeout": "..."}` (optional)
`stop_validation`	`/validation`	`DELETE`	—

Polling: `GET /validation/result`

11.21.1.2.7 1.2.7. 7 Switch Status and Connectivity

Deprecated Command	Replacement Endpoint	Method	Notes
`check_switch_status`	`/agents/connectivity/check`	`POST`	Returns `202 Accepted`. Poll `GET /agents/connectivity/check/result`.
`show_switches`	`/resources/nodes`	`GET`	Synchronous. Returns all nodes with full device information.

11.21.1.2.8 1.2.8. 8 Commands Without a Replacement

Deprecated Command	Reason
`load`	Loaded topology from a saved file prefix. Use `POST /topology/load/json` for JSON topologies instead.
`add_certificate`	Certificate management is handled at the Apache/infrastructure layer.
`show_switch_history`	Not currently exposed as a REST endpoint.
`amber_show_latest`	The full amber report is available via `GET /report/amber`, but the filtered "latest" view has no direct equivalent.

11.21.1.2.9 1.2.9. 9 Server Shutdown

Deprecated Command	Replacement Endpoint	Method	Notes
`exit`	`/cv_controller/stop`	`POST`	Synchronous. **Important:** this endpoint is served by the CV Controller service (Apache location `/cv_controller`), not by the main `/cablevalidation` service where `/commands/*` used to live.

11.21.1.3 1.3. Device Exclude/Unexclude Endpoints

The device exclusion mechanism has been replaced by a unified Device Status Management API. Instead of a simple excluded/not-excluded binary, each device now has a status (e.g. `VALIDATION_READY`, `UNMANAGED`, `UNSUPPORTED`) with controlled transitions and audit history.

11.21.1.3.1 1.3.1. 1 Endpoint Migration

Deprecated Endpoint	Method	Replacement Endpoint	Method	Request Body / Notes
`/devices/excluded`	`GET`	`/resources/nodes`	`GET`	Each node object now includes a `status` field. Filter for `UNMANAGED` status to find previously "excluded" devices.
`/devices/exclude`	`POST`	`/devices/status`	`PUT`	`{"devices": [...], "status": "UNMANAGED", "reason": "..."}`
`/devices/unexclude`	`DELETE`	`/devices/status`	`PUT`	`{"devices": [...], "status": "VALIDATION_READY", "reason": "..."}`

11.21.1.3.2 1.3.2. 2 New Device Status Endpoints

The following endpoints provide richer device lifecycle management:

Endpoint	Method	Description

Endpoint	Method	Description
`/devices/status/{device_name}`	`PUT`	Set the status of a single device. Body: `{"status": "...", "reason": "..."}`
`/devices/status`	`PUT`	Set the status of multiple devices in bulk. Body: `{"devices": [...], "status": "...", "reason": "..."}`
`/devices/status/{device_name}/transitions`	`GET`	Get allowed status transitions for a device (returns current status and valid next statuses).
`/devices/status/{device_name}/history`	`GET`	Get the status change history for a device. Supports `limit` and `offset` query parameters.

11.21.1.4 1.4. Summary

Category	Deprecated Endpoints	With Replacement	Without Replacement
CommandProc generic routes	5	2 (execute, output)	3 (list, status, help)
CommandProc commands	24	20	4 (`load`, `add_certificate`, `show_switch_history`, `amber_show_latest`)
Device exclude/unexclude	3	3	0
Total	**32**	**25**	**7**

All replacement endpoints are served under the `/cablevalidation` Apache location unless otherwise noted (the `exit` replacement is under `/cv_controller`).

11.22 Dashboard REST APIs

The Dashboard REST APIs provide the aggregated information displayed by the Operational Dashboard and AI Factory Dashboard. Requests made through the CVT HTTPS proxy use the **`/cablevalidation`** prefix shown below.

11.22.1 Get Dashboard Summary

- Description – Gets the aggregated data used by the Operational Dashboard, including validation status, topology metadata, agent health, device-status KPIs, issue statistics, circuit statistics, the last precheck time.
- Request URL – **`GET /cablevalidation/dashboard/summary`**
- Response Content Type – **`application/json`**
- Status Codes
  - **`200`** – OK
  - **`400`** – BAD REQUEST

For administrators, the response also includes **buildup_start_time**, **buildup_go_live_target**, and **active_operations**. These fields are omitted for non-administrator users.

- Response Example


```
{
  "validation": {
    "status": "started",
    "started_at": 1767225600,
    "started_by": "admin",
    "duration_seconds": 3600,
    "last_cycle_seconds_ago": 12
  },
  "version": "1.4.0",
  "topology": {
    "file_name": "site-topology.xlsx",
    "file_id": "94a8c27f",
    "load_time": 1767225400,
    "switches_count": 32,
    "hosts_count": 96,
    "links_count": 128
  },
  "agents": {
    "total_device_count": 128,
    "correct_agent_version_installed": 120,
    "no_ping": 2,
    "no_agent": 4,
    "late_reports": 1,
    "stale_reports": 1
  },
  "device_status": {
    "period": "today",
    "total_devices": 128,
    "statuses": {
      "production": {
        "count": 80,
        "change": 6,
        "direction": 1
      },
      "validating": {
        "count": 20,
        "change": 2,
        "direction": 1
      }
    }
  },
  "issues_summary": [
    {
      "syndrome": "Link Down, No signal",
      "number_of_occurrences": 3,
      "switches_affected": 2,
      "number_of_occurrences_from_stale": 1,
      "switches_affected_from_stale": 1,
      "change": -1,
      "direction": -1,
      "change_period_minutes": 60
    }
  ],
  "circuits_stats": {
    "total_circuits": 128,
    "healthy_circuits": 110,
    "unhealthy_circuits": 12,
    "not_active_circuits": 6
  },
  "last_precheck_ts": 1767225300,
  "scope": {
    deployment_id: null,
    device_count: null,
    is_entire_file: true,
    workspace_id: null,
  },
  "buildup_start_time": 1767225600,
  "buildup_go_live_target": 1768435200,
  "active_operations": []
}
```


11.22.2 Get Port-Issue History

- Description – Gets the port-issue history used by the Operational Dashboard chart. The response is ordered by time and includes the most recent live point when data is available.
- Request URL – **GET /cablevalidation/dashboard/port-issue-history**
- Request Parameters
  - **range** – Time range to return. Supported values: **1h**, **6h**, **24h**, and **7d**.
    - Type: string
    - Optional: true

        - Default: `7d`
- Response Content Type – `application/json`
- Status Codes
    - `200` – OK
    - `400` – BAD REQUEST
- Response Example

```
[
  {
    "timestamp": 1767222000,
    "total_issues": 15,
    "total_devices": 128,
    "validating_devices": 22
  },
  {
    "timestamp": 1767225600,
    "total_issues": 12,
    "total_devices": 128,
    "validating_devices": 20
  }
]
```

11.22.3 Get Build-Up Progress

- Description – Gets device-status history for the AI Factory Dashboard build-up progress chart. Device statuses are grouped as follows:
    - `production` – Production and Production Ready.
    - `validating` – Validating.
    - `setup` – Not Provisioned, Provisioned, Agent Ready, and Validation Ready.
    - `not_ready` – Not Installed, Installed No Power, Unsupported, Undefined, and Validation Only.
    - `maintenance` – Maintenance and Drain/Troubleshooting.
    - `unmanaged` – Unmanaged.
- Request URL – `GET /cablevalidation/dashboard/buildup-progress`
- Access – Administrator only.
- Request Parameters
    - `range` – Supported values: `6h`, `24h`, `7d`, and `30d`.
        - Type: string
        - Optional: true
        - Default: `7d`
    - `resolution` – Supported values: `5 minutes`, `15 minutes`, `1 hour`, `6 hours`, and `1 day`. If omitted, the service selects a resolution appropriate for the requested range.
        - Type: string
        - Optional: true
- Response Content Type – `application/json`
- Status Codes
    - `200` – OK
    - `400` – BAD REQUEST
    - `401` – UNAUTHORIZED
    - `403` – FORBIDDEN
- Response Example

```
{
  "timestamps": [1767222000, 1767225600],
  "series": {
    "production": [74, 80],
    "validating": [24, 20],
    "setup": [20, 18],
    "not_ready": [6, 4],
    "maintenance": [2, 2],
    "unmanaged": [2, 4]
  },
  "total_nodes": [128, 128],
  "range": "7d",
  "resolution": "1 hour"
}
```

11.22.4 Get Build-Up Velocity

- Description – Gets production-readiness velocity statistics for the AI Factory Dashboard. Production and Production Ready devices are included in the production total. The calculation uses up to 30 days of build-up history.
- Request URL – `GET /cablevalidation/dashboard/buildup-velocity`
- Access – Administrator only.
- Response Content Type – `application/json`
- Status Codes
  - `200` – OK
  - `400` – BAD REQUEST
  - `401` – UNAUTHORIZED
  - `403` – FORBIDDEN
- Response Example

```
{
  "average": 8.4,
  "days": 14,
  "today": 6,
  "yesterday": 9,
  "peak": 14,
  "min": 3,
  "remaining": 48,
  "total_nodes": 128,
  "unit": "Nodes/Day"
}
```

11.22.5 Update Build-Up Timestamps

- Description – Updates the build-up start time, go-live target, or both. Values are Unix timestamps in seconds. An omitted field remains unchanged.
- Request URL – `PUT /cablevalidation/dashboard/buildup-timestamps`
- Access – Administrator only.
- Request Content Type – `application/json`
- Response Content Type – `application/json`
- Validation Rules
  - Each supplied value must be a positive numeric Unix timestamp in seconds.
  - The go-live target cannot be earlier than the build-up start time.
  - The go-live target must be within one year of the build-up start time.
- Status Codes
  - `200` – OK
  - `400` – BAD REQUEST
  - `401` – UNAUTHORIZED

    - `403` – FORBIDDEN
- Request Example

```
{
  "buildup_start_time": 1767225600,
  "buildup_go_live_target": 1768435200
}
```

- Response Example

```
{
  "buildup_start_time": 1767225600,
  "buildup_go_live_target": 1768435200
}
```

11.23 Event REST APIs

The Event REST APIs provide recent activity for dashboard feeds and detailed information about individual events.

11.23.1 Get Recent Events

- Description – Gets recent events for the loaded topology. This endpoint is used for dashboard activity and recent device-transition feeds.
- Request URL – `GET /cablevalidation/events/recent`
- Request Parameters
    - `limit` – Maximum number of events to return. If omitted, all matching events are returned.
        - Type: integer
        - Optional: true
    - `types` – Event type to include. Repeat this parameter to request multiple event types, for example: `?types=device_transition&types=issue_detected`.
        - Type: string
        - Optional: true
    - `truncate` – When `true`, long node and issue lists are shortened according to the server preview limit. Use `false` to request the complete stored event payload.
        - Type: boolean
        - Optional: true
        - Default: `true`
- Supported Event Types
    - `issue_detected`
    - `issue_resolved`
    - `device_transition`
    - `agent_online`
    - `agent_offline`
    - `validation_started`
    - `validation_stopped`
    - `topology_loaded`

- `test_started`
- `test_completed`
- `circuit_refresh_started`
- `agents_deployed`
- `agents_removed`
- `precheck_started`
- `precheck_completed`

Supported severity values are `info`, `warning`, `error`, and `success`.

- Response Content Type – `application/json`
- Status Codes
  - `200` – OK
  - `400` – BAD REQUEST
- Response Example

```
[
  {
    "id": 941,
    "timestamp": 1767225600,
    "event_type": "device_transition",
    "severity": "success",
    "message": "6 devices moved to Production",
    "file_id": "94a8c27f",
    "scope_nodes": ["server-01", "server-02"],
    "scope_nodes_count": 6,
    "details": {
      "from_status": "validating",
      "to_status": "production"
    }
  }
]
```

11.23.2 Get Event Details

- Description – Gets a single event by its identifier.
- Request URL – `GET /cablevalidation/events/{event_id}`
- Path Parameters
  - `event_id` – Numeric identifier of the event.
    - Type: integer
    - Required: true
- Request Parameters
  - `truncate` – When `true`, long node and issue lists are shortened according to the server preview limit. Use `false` to request the complete stored event payload.
    - Type: boolean
    - Optional: true
    - Default: `true`
- Response Content Type – `application/json`
- Status Codes
  - `200` – OK
  - `400` – BAD REQUEST
  - `404` – NOT FOUND
- Response Example

```
{
  "id": 941,
  "timestamp": 1767225600,
  "event_type": "device_transition",
  "severity": "success",
  "message": "6 devices moved to Production",
  "file_id": "94a8c27f",
  "scope_nodes": ["server-01", "server-02", "server-03"],
  "scope_nodes_count": 6,
  "details": {
    "from_status": "validating",
    "to_status": "production"
  }
}
```

11.24 Layouts APIs

The Layouts APIs manage saved table layouts for CVT report tables. Layouts are scoped to the authenticated user and the specified table page. Responses can include both system-defined layouts and the current user's personal layouts.

11.24.1 Get Layouts

- Description – Gets the predefined layouts and the authenticated user's personal layouts for a table.
- Request URL – **GET /cablevalidation/report/layouts/{page}**
- Path Parameters
  - **page** – Table identifier. See Supported Page Values.
    - Type: string
    - Required: true
- Response Content Type – **application/json**
- Status Codes
  - **200** – OK
- Response Example

```
[
  {
    "name": "Summary",
    "is_predefined": true,
    "is_default": false
  },
  {
    "name": "rx_errors_debug",
    "is_predefined": false,
    "is_default": false
  }
]
```

Predefined layouts are returned first. Personal layouts follow in alphabetical order.

11.24.2 Create Layout

- Description – Creates a personal layout for the authenticated user and table.
- Request URL – **POST /cablevalidation/report/layouts/{page}**
- Path Parameters
  - **page** – Table identifier. See Supported Page Values.
    - Type: string
    - Required: true
- Request Content Type – **application/json**

- Request Data
  - **layout_name** – Unique name for the personal layout within the table.
    - Type: string
    - Required: true
  - **columns_state** – Changed column properties, keyed by column identifier. Only properties that differ from the table's initial state need to be supplied.
    - Type: object
    - Required: true
- Request Example

```
{
  "layout_name": "rx_errors_debug",
  "columns_state": {
    "source_port": {
      "colId": "source_port",
      "width": 180,
      "hide": false,
      "pinned": "left",
      "order": 0,
      "sort": "asc",
      "sortIndex": 0
    }
  }
}
```

- Status Codes
  - **201** – CREATED
  - **400** – BAD REQUEST
  - **409** – CONFLICT; a predefined or personal layout with this name already exists
  - **500** – INTERNAL SERVER ERROR; the layout could not be stored
- Response – No response body.

11.24.3 Get Layout

- Description – Gets one predefined or personal layout snapshot.
- Request URL – **GET /cablevalidation/report/layouts/{page}/{layout_name}**
- Path Parameters
  - **page** – Table identifier. See Supported Page Values.
    - Type: string
    - Required: true
  - **layout_name** – Name of the requested layout.
    - Type: string
    - Required: true
- Response Content Type – **application/json**
- Status Codes
  - **200** – OK
  - **400** – BAD REQUEST; the layout name is invalid
  - **404** – NOT FOUND
  - **500** – INTERNAL SERVER ERROR; the stored layout is malformed
- Response Example

```
{
  "layout_name": "rx_errors_debug",
  "columns_state": {
```

```
    "source_port": {
      "colId": "source_port",
      "width": 180,
      "hide": false,
      "pinned": "left",
      "order": 0,
      "sort": "asc",
      "sortIndex": 0
    }
  }
}
```

11.24.4 Update Layout

- Description – Replaces the saved column state of an existing personal layout. Predefined layouts cannot be updated.
- Request URL – **PUT /cablevalidation/report/layouts/{page}/{layout_name}**
- Path Parameters
  - **page** – Table identifier. See Supported Page Values.
    - Type: string
    - Required: true
  - **layout_name** – Name of the personal layout to update.
    - Type: string
    - Required: true
- Request Content Type – **application/json**
- Request Example

```
{
  "columns_state": {
    "source_port": {
      "colId": "source_port",
      "width": 220,
      "hide": false,
      "pinned": "left",
      "order": 0
    }
  }
}
```

- Status Codes
  - **200** – OK
  - **400** – BAD REQUEST
  - **403** – FORBIDDEN; predefined layouts cannot be updated
  - **404** – NOT FOUND
  - **500** – INTERNAL SERVER ERROR; the layout could not be stored
- Response – No response body.

11.24.5 Delete Layout

- Description – Deletes a personal layout. Predefined layouts cannot be deleted.
- Request URL – **DELETE /cablevalidation/report/layouts/{page}/{layout_name}**
- Path Parameters
  - **page** – Table identifier. See Supported Page Values.
    - Type: string
    - Required: true
  - **layout_name** – Name of the personal layout to delete.
    - Type: string

        - Required: true
- Status Codes
    - `204` – NO CONTENT
    - `400` – BAD REQUEST; the layout name is invalid
    - `403` – FORBIDDEN; predefined layouts cannot be deleted
    - `404` – NOT FOUND
- Response – No response body.

11.24.5.1 Layout Name Requirements

A layout name must:

- Start with a letter or number.
- Contain only letters, numbers, hyphens (`-`), underscores (`_`), or periods (`.`).
- Not contain `..` , `/` , or `\` .

Examples of invalid names include `../secret` , `secret/child` , `secret\child` , `layout..name` , names containing spaces, and an empty name.

11.24.5.2 Layout Snapshot Fields

The `columns_state` object is sparse: it stores only changed column settings. When a layout is applied, CVT merges these settings with the table's initial column state.

Common column properties include:

Field	Description
`colId`	Column identifier.
`width`	Column width in pixels.
`hide`	Whether the column is hidden.
`pinned`	Pinned position, such as `left` or `right` .
`order`	Visible display order.
`sort`	Sort direction, such as `asc` or `desc` .
`sortIndex`	Position in multi-column sorting.

12 Document Revision History

Version	Date	Description
2.0.1	Aug 27, 2026	Updated the following sections: • Introduction • Changes and New Features in This Release • Installation Notes • Bug Fixes in This Release • Known Issues in This Release • Deploying the Module • Dashboards • Operational Dashboard • Cluster Sizing Guide • AI Factory Dashboard • P2P File • Bringup CLI • CVT Configuration • Devices View • Rack View • Reports • System Admin • Setup & Validate • Agent Management • Collector Management • Layouts APIs • Dashboard REST APIs • Event REST APIs • Rest APIs • Reports APIs •
1.9.1	Jun 2, 2026	Documentation History in UFM Cable Validation Tool v1.9.1
1.8.1	Feb 20, 2026	Documentation History in UFM Cable Validation Tool v1.8.1
1.7.1	Nov 5, 2025	Documentation History in UFM Cable Validation Tool v1.7.1
1.6.0	Aug 7, 2025	Documentation History in UFM Cable Validation Tool v1.6.0
1.5.0	May 5, 2025	Documentation History in UFM Cable Validation Tool v1.5.0
1.4.0	Feb 10, 2025	Documentation History in UFM Cable Validation Tool v1.4.0
1.3.1	Aug 12, 2024	Documentation History in UFM Cable Validation Tool v1.3.1
1.2.0	Nov 5, 2023	Documentation History in UFM Cable Validation Tool v1.2.0
1.1.0	Aug 10, 2023	Documentation History in UFM Cable Validation Tool v1.1.0
1.0.0	May 8, 2023	First release

Notice

This document is provided for information purposes only and shall not be regarded as a warranty of a certain functionality, condition, or quality of a product. Neither NVIDIA Corporation nor any of its direct or indirect subsidiaries and affiliates (collectively: "NVIDIA") make any representations or warranties, expressed or implied, as to the accuracy or completeness of the information contained in this document and assumes no responsibility for any errors contained herein. NVIDIA shall have no liability for the consequences or use of such information or for any infringement of patents or other rights of third parties that may result from its use. This document is not a commitment to develop, release, or deliver any Material (defined below), code, or functionality.

NVIDIA reserves the right to make corrections, modifications, enhancements, improvements, and any other changes to this document, at any time without notice. Customer should obtain the latest relevant information before placing orders and should verify that such information is current and complete.

NVIDIA products are sold subject to the NVIDIA standard terms and conditions of sale supplied at the time of order acknowledgement, unless otherwise agreed in an individual sales agreement signed by authorized representatives of NVIDIA and customer ("Terms of Sale"). NVIDIA hereby expressly objects to applying any customer general terms and conditions with regards to the purchase of the NVIDIA product referenced in this document. No contractual obligations are formed either directly or indirectly by this document.

NVIDIA products are not designed, authorized, or warranted to be suitable for use in medical, military, aircraft, space, or life support equipment, nor in applications where failure or malfunction of the NVIDIA product can reasonably be expected to result in personal injury, death, or property or environmental damage. NVIDIA accepts no liability for inclusion and/or use of NVIDIA products in such equipment or applications and therefore such inclusion and/or use is at customer's own risk.

NVIDIA makes no representation or warranty that products based on this document will be suitable for any specified use. Testing of all parameters of each product is not necessarily performed by NVIDIA. It is customer's sole responsibility to evaluate and determine the applicability of any information contained in this document, ensure the product is suitable and fit for the application planned by customer, and perform the necessary testing for the application in order to avoid a default of the application or the product. Weaknesses in customer's product designs may affect the quality and reliability of the NVIDIA product and may result in additional or different conditions and/or requirements beyond those contained in this document. NVIDIA accepts no liability related to any default, damage, costs, or problem which may be based on or attributable to: (i) the use of the NVIDIA product in any manner that is contrary to this document or (ii) customer product designs.

No license, either expressed or implied, is granted under any NVIDIA patent right, copyright, or other NVIDIA intellectual property right under this document. Information published by NVIDIA regarding third-party products or services does not constitute a license from NVIDIA to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property rights of the third party, or a license from NVIDIA under the patents or other intellectual property rights of NVIDIA.

Reproduction of information in this document is permissible only if approved in advance by NVIDIA in writing, reproduced without alteration and in full compliance with all applicable export laws and regulations, and accompanied by all associated conditions, limitations, and notices.

THIS DOCUMENT AND ALL NVIDIA DESIGN SPECIFICATIONS, REFERENCE BOARDS, FILES, DRAWINGS, DIAGNOSTICS, LISTS, AND OTHER DOCUMENTS (TOGETHER AND SEPARATELY, "MATERIALS") ARE BEING PROVIDED "AS IS." NVIDIA MAKES NO WARRANTIES, EXPRESSED, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE MATERIALS, AND EXPRESSLY DISCLAIMS ALL IMPLIED WARRANTIES OF NONINFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE. TO THE EXTENT NOT PROHIBITED BY LAW, IN NO EVENT WILL NVIDIA BE LIABLE FOR ANY DAMAGES, INCLUDING WITHOUT LIMITATION ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, PUNITIVE, OR CONSEQUENTIAL DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF ANY USE OF THIS DOCUMENT, EVEN IF NVIDIA HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. Notwithstanding any damages that customer might incur for any reason whatsoever, NVIDIA's aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms of Sale for the product.

Trademarks
NVIDIA, the NVIDIA logo, and Mellanox are trademarks and/or registered trademarks of NVIDIA Corporation and/or its affiliates in the U.S. and in other countries. Other company and product names may be trademarks of the respective companies with which they are associated.

Copyright
© 2026 NVIDIA Corporation & affiliates. All Rights Reserved.