Vader Programming Guide

September 12, 2013 By Jeff Sheffel

[Purpose and Scope 7](#_Toc364348703)

[References 7](#_Toc364348704)

[Definition of Terms 7](#_Toc364348705)

[Vader Overview 8](#_Toc364348706)

[Vader Application Philosophy 8](#_Toc364348707)

[Vader Programming Concepts 8](#_Toc364348708)

[Coding Philosophy 8](#_Toc364348709)

[Programmer Skills 8](#_Toc364348710)

[Vader Use Cases 9](#_Toc364348711)

[Vader Instances 9](#_Toc364348712)

[Vader Test Instance 9](#_Toc364348713)

[Vader Integration Instance 10](#_Toc364348714)

[Vader Production Instance 10](#_Toc364348715)

[Vader Configurations 10](#_Toc364348716)

[Vader Versioning With Git 10](#_Toc364348717)

[Vader Users 10](#_Toc364348718)

[QwestRx 10](#_Toc364348719)

[PollDSLAM 10](#_Toc364348720)

[Network Diagnostic Portal (NDP) 11](#_Toc364348721)

[Loop Provisioning Center (LPC) 14](#_Toc364348722)

[Network Monitoring and Analysis (NMA) Team 14](#_Toc364348723)

[Other CenturyLink Systems 14](#_Toc364348724)

[NCON 14](#_Toc364348725)

[ISYS 15](#_Toc364348726)

[Service Orders 15](#_Toc364348727)

[Service Order Device Tags 16](#_Toc364348728)

[Integrator 17](#_Toc364348729)

[Network Components 17](#_Toc364348730)

[CenturyLink (CTL) Management Systems (EMS and CMS) 17](#_Toc364348731)

[Remote Terminal Devices 17](#_Toc364348732)

[Adtran and Calix Differences 18](#_Toc364348733)

[Adtran 19](#_Toc364348734)

[Adtran OLTs 19](#_Toc364348735)

[Adtran ONTs 19](#_Toc364348736)

[Calix 20](#_Toc364348737)

[Calix OLTs 20](#_Toc364348738)

[Calix E7 OLT 20](#_Toc364348739)

[Calix ONTs 21](#_Toc364348740)

[Single Family Unit and Small Business ONTs 22](#_Toc364348741)

[Multi Dwelling Unit ONTs 22](#_Toc364348742)

[Business Service and Mobile Backhaul ONTs 24](#_Toc364348743)

[Calix 760G Series ONTs 25](#_Toc364348744)

[Client Interfaces 26](#_Toc364348745)

[Vader Requests 26](#_Toc364348746)

[Vader Responses 26](#_Toc364348747)

[XML Responses 26](#_Toc364348748)

[Device Status 27](#_Toc364348749)

[Calix ONT Administrative State 27](#_Toc364348750)

[Calix ONT Operational State 27](#_Toc364348751)

[Adtran ONT Administrative State 27](#_Toc364348752)

[Adtran ONT Operational State 28](#_Toc364348753)

[Device Bounces 28](#_Toc364348754)

[Vader Administrator Website 28](#_Toc364348755)

[Vader Application Administration 28](#_Toc364348756)

[Vader Logfiles 29](#_Toc364348757)

[XML Response Logs 29](#_Toc364348758)

[Error Logs 29](#_Toc364348759)

[Administrator Notifications 29](#_Toc364348760)

[Service Disabling 29](#_Toc364348761)

[Vader User Documentation 30](#_Toc364348762)

[Diagnosing Vader Errors 30](#_Toc364348763)

[Built-in Error Management 30](#_Toc364348764)

[OLT TID Mismatch 30](#_Toc364348765)

[Remedy IT Helpdesk Tickets Against Vader 31](#_Toc364348766)

[Application Framework 31](#_Toc364348767)

[Apache 31](#_Toc364348768)

[Apache Logfile Analysis 31](#_Toc364348769)

[PHP 31](#_Toc364348770)

[PHP Runtime Errors 31](#_Toc364348771)

[PHP Mail Usage 32](#_Toc364348772)

[MySQL Database 32](#_Toc364348773)

[Database Schema 32](#_Toc364348774)

[Database Backups 32](#_Toc364348775)

[Vader Tables 32](#_Toc364348776)

[GPON\_OLT\_CHASSIS 32](#_Toc364348777)

[GPON\_ONT\_EQUIPMENT 33](#_Toc364348778)

[GPON\_CUSTOMER\_NTWK 33](#_Toc364348779)

[NCON\_FTTH\_GPON 34](#_Toc364348780)

[NCON\_FTTH\_BPON 34](#_Toc364348781)

[WEBSITE\_ACCESS 34](#_Toc364348782)

[vader\_message\_log\_test 34](#_Toc364348783)

[vader\_message\_log\_integration 35](#_Toc364348784)

[vader\_message\_log\_production 35](#_Toc364348785)

[Crontabs 35](#_Toc364348786)

[get\_gpon\_ncon\_info.ksh 35](#_Toc364348787)

[ncon\_gpon\_dump.ksh 35](#_Toc364348788)

[get\_gpon\_equip.ksh 35](#_Toc364348789)

[manage\_gpon\_batt\_alarms.ksh (should be *verify\_adtran\_OLT\_connectivity.ksh*) 36](#_Toc364348790)

[rotate\_apache\_logs.sh 36](#_Toc364348791)

[rotate\_vader\_logs.ksh 36](#_Toc364348792)

[apachecheck 37](#_Toc364348793)

[SNMP 37](#_Toc364348794)

[General MIB Definitions 38](#_Toc364348795)

[MIBs of Interest 38](#_Toc364348796)

[ONT Description Table 38](#_Toc364348797)

[SOAP 39](#_Toc364348798)

[Vader SOAP Tester 40](#_Toc364348799)

[PHP NuSoap Example 40](#_Toc364348800)

[PHP NuSoap Example Error Response 42](#_Toc364348801)

[Production Scripts 43](#_Toc364348802)

[get\_gpon\_ont\_alarms.exp Expect Script 43](#_Toc364348803)

[get\_ont\_alarms.ksh 44](#_Toc364348804)

[get\_battery\_alarms.ksh 44](#_Toc364348805)

[TL-1 Commands 44](#_Toc364348806)

[Application Security 44](#_Toc364348807)

[HTTPS Web Interface 44](#_Toc364348808)

[LDAP Authentication 44](#_Toc364348809)

[Vader Administrative User Accounts 45](#_Toc364348810)

[Webservice Interface 45](#_Toc364348811)

[Software Development 45](#_Toc364348812)

[Documentation 45](#_Toc364348813)

[Vader Development 45](#_Toc364348814)

[Release Milestones and Significant Code Enhancements 45](#_Toc364348815)

[Vader Code Structure 46](#_Toc364348816)

[Coding Standards 46](#_Toc364348817)

[Standard Modules 46](#_Toc364348818)

[Configuration Settings 46](#_Toc364348819)

[Database Interface 46](#_Toc364348820)

[Debug Interface 46](#_Toc364348821)

[Custom Utilites 46](#_Toc364348822)

[Device Analysis Utilities 46](#_Toc364348823)

[get-ontids-for-calix-olts.pl 46](#_Toc364348824)

[show-olt-config-adtran.sh 47](#_Toc364348825)

[get-olt-ip-addresses.pl 47](#_Toc364348826)

[get-ontids-for-calix-olts.pl 47](#_Toc364348827)

[show-olt-to-ont-pairs.sh 47](#_Toc364348828)

[run-clei-code-analysis-all-vader-logs.sh 47](#_Toc364348829)

[convert-mib-index-to-ont-id.pl 47](#_Toc364348830)

[SNMP Utilities 47](#_Toc364348831)

[run-walk-mib.sh 47](#_Toc364348832)

[run-snmpwalk-over-vader-olts.sh 47](#_Toc364348833)

[run-snmpwalk-all-vader-devices.sh 47](#_Toc364348834)

[run-snmpget-all-vader-devices.sh 47](#_Toc364348835)

[run-snmpwalk-ifDescr-all-calix.sh 47](#_Toc364348836)

[run-snmpwalk-SNMPv2-MIB-all-calix.sh 47](#_Toc364348837)

[Code Test Utilities 48](#_Toc364348838)

[wget-vader-request.sh 48](#_Toc364348839)

[wget-vader-test-request.sh 48](#_Toc364348840)

[wget-vader-integration-request.sh 48](#_Toc364348841)

[wget-vader-prod-request.sh 48](#_Toc364348842)

[compare-request-test-to-prod.sh 48](#_Toc364348843)

[compare-request-integration-to-prod.sh 48](#_Toc364348844)

[compare-bounce-test-to-prod.sh 48](#_Toc364348845)

[compare-bounce-integration-to-prod.sh 48](#_Toc364348846)

[test-class-LogDatabase.php 48](#_Toc364348847)

[test-getChassisOnts-method.php 48](#_Toc364348848)

[test-getChassisMulticastProfiles-method.php 48](#_Toc364348849)

[test-doSnmpGet.php 48](#_Toc364348850)

[Code Management Utilities 49](#_Toc364348851)

[format-git-log-output.sh 49](#_Toc364348852)

[describe-table-GPON\_OLT\_CHASSIS.sh 49](#_Toc364348853)

[Test Server 49](#_Toc364348854)

[checkout-vader-to-htdocs.sh 49](#_Toc364348855)

[clear-vader-logfiles.sh 49](#_Toc364348856)

[configure-vader-test-website.sh 49](#_Toc364348857)

[create-vader-release.sh 49](#_Toc364348858)

[pull-vader-to-clone.sh 49](#_Toc364348859)

[vader-checkout-configure.sh 49](#_Toc364348860)

[list-vader-logfiles.sh 49](#_Toc364348861)

[save-vader-logfiles.sh 49](#_Toc364348862)

[copy-vader-docs-from-doc-server.sh 49](#_Toc364348863)

[copy-vader-docs-to-integration-server.sh 49](#_Toc364348864)

[Production Server 49](#_Toc364348865)

[analyze-apache-log-vader.pl 49](#_Toc364348866)

[generate-apache-log-stat-files.sh 49](#_Toc364348867)

[create-apache-log-summary-webpage.pl 49](#_Toc364348868)

[run-create-apache-log-summary-webpage.sh 49](#_Toc364348869)

[test-week-nums-generation.sh 49](#_Toc364348870)

[get-vader-gpon-olt-chassis-table.sh 49](#_Toc364348871)

[run-vader-olts-snmp-check.pl 49](#_Toc364348872)

[Production Server Vader User 50](#_Toc364348873)

[configure-vader-integration-website.sh 50](#_Toc364348874)

[backup-vdsltechsupp-home-vader.sh 50](#_Toc364348875)

[show-vader-transaction-errors.sh 50](#_Toc364348876)

[mysqldump-GPON\_OLT\_CHASSIS.sh 50](#_Toc364348877)

[archive-table-GPON\_OLT\_CHASSIS.sh 50](#_Toc364348878)

[create-vader-logfiles-tarfile.sh 50](#_Toc364348879)

[list-vader-logfiles.sh 50](#_Toc364348880)

[run-pkginfo-git-dependencies.sh 50](#_Toc364348881)

[configure-vader-production-website.sh 50](#_Toc364348882)

[Logfile Analyzer Utilites 50](#_Toc364348883)

[summarize-vader-logfile-sample-simple.pl 50](#_Toc364348884)

[summarize-vader-logfile-xml-smart.pl 50](#_Toc364348885)

[extract-vader-response-elements-ont-values.pl 50](#_Toc364348886)

[summarize-vader-logfile-xml-twig.pl 50](#_Toc364348887)

[Unit Testing 50](#_Toc364348888)

[Laboratory Device Testing 50](#_Toc364348889)

[Software Emulation of Hardware Devices 51](#_Toc364348890)

[Benchmarks 51](#_Toc364348891)

[Vader Deployment 51](#_Toc364348892)

[Test Server Testden3 51](#_Toc364348893)

[Integration Server Vdsltechsupp 52](#_Toc364348894)

[Production Server Vdsltechsupp 53](#_Toc364348895)

[Vader Logfile Analysis 54](#_Toc364348896)

[GUI User Interface 54](#_Toc364348897)

[Adding a US State to Vader Support 54](#_Toc364348898)

[US Map HTML Image Map 54](#_Toc364348899)

[Vader OLT Database Edit Webpages 54](#_Toc364348900)

[Releasing Vader Upgrades 54](#_Toc364348901)

[Integration Patch Upgrades 54](#_Toc364348902)

[Production Patch Upgrades 55](#_Toc364348903)

[Appendix 56](#_Toc364348904)

[Resources 56](#_Toc364348905)

[CenturyLink 56](#_Toc364348906)

[PHP 56](#_Toc364348907)

[Git 56](#_Toc364348908)

[Vendors 56](#_Toc364348909)

[Adtran 56](#_Toc364348910)

[Calix 56](#_Toc364348911)

[Technologies 57](#_Toc364348912)

[Unicast and Multicast 57](#_Toc364348913)

[Vader Source Lines of Code (SLOC) Count 57](#_Toc364348914)

[Vader 8.1.0 57](#_Toc364348915)

[Vader 7.0 57](#_Toc364348916)

[Telecommunications Acronyms at CenturyLink 58](#_Toc364348917)

[Personnel Contacts 59](#_Toc364348918)

[Field Engineers 59](#_Toc364348919)

[Equipment Lab 60](#_Toc364348920)

[Network Architects 60](#_Toc364348921)

[Applications 60](#_Toc364348922)

[QwestRx 60](#_Toc364348923)

[Network Diagnostics Portal (NDP) 60](#_Toc364348924)

[PollDSLAM 61](#_Toc364348925)

[Loop Provisioning Center (LPC) 61](#_Toc364348926)

[Network Monitoring and Analysis (NMA) 61](#_Toc364348927)

[NCON 61](#_Toc364348928)

[Project Management 61](#_Toc364348929)

[Systems 61](#_Toc364348930)

[The Good (the Bad and the Ugly) Stories 61](#_Toc364348931)

[Pipe Speed Required Immediately 61](#_Toc364348932)

# Purpose and Scope

This document provides a programming guide for the CenturyLink (CTL) Vader application. The intended audience is for software engineers that need an understanding of how the Vader application is designed and implemented.

The scope of this document, as it relates to Vader, is described in the Vader Overview section below.

The exact scope of the Vader application is somewhat vague, and to some people, may include some peripheral functions (like pdslam and battery management). One reason for this is that historically, there isn’t a clear definition of terms or an organized design of the (VROC) applications. Rather, there is a spagetti relationship between webpages, webservices, cron scripts, user scripts, database tables, and other undocumented agents.

What Vader is, is a CenturyLink (Legacy Qwest) GPON diagnostics application, and nothing more. Yet, it is a complex and distributed application. Refer to the Vader Overview section, immediately below.

Vader was originally coded by Rick Lorenz, circa 2009, as an in-house, slowly evolving device diagnostics tool. Over a few ensuing years, several corporate-level production applications begin to rely on the Vader services. On June 3, 2013, the Vader 8.1.0 version of the application was released into production. The 8.0 branch of Vader began the transition of Vader to an enterprise class application. There is a potential to add many enhancements and features, as CenturyLink requirements continue to influence the Vader scope.

# References

Refer to these additional resources for specific information about Vader:

1. Vader Application Overview
2. Vader Interface Specification
3. Vader Project Plan
4. Vader Debugging Guide
5. Vader Install Guide
6. Vader Git Guide
7. Vader Coding Details
8. Vader SNMP Guide

The above *Vader programmer* resources can be found at the Vader website (and maintained in the Git repository):

* http://<vader-base-url>/doc

Users should be directed to the production Vader user documentation webpage:

* http://<vader-base-url>/doc/list.php

The difference is that users then wouldn’t know that the /doc folder is (Apache) indexable and accessible.

Also refer to the *References* section in the appendix of this document.

# Definition of Terms

The terms *Vader client* and *Vader user* are somewhat synonymous. A Vader user may imply a person, while a Vader client implies another software application, and may also be considered a *consumer* of *Vader services*.

A *Vader Administrator* is a highest level Vader user, who is authorized to suspend Vader services (which has never practically been done for Vader 24x7 production services).

A *Vader Advanced User* is granted authorizations to edit the Vader OLT Database.

Refer to the *constants-security.php* source file for a complete list of authorization classifications.

A *Vader User* is a generic term for any CTL employee that logs in to use the Vader GUI. This term could distiinguish Vader clients, which use the webservices (ie. software client programs).

# Vader Overview

Vader provides a webservice and an administrator’s (GUI) website, for diagnosing and resetting production GPON ONT field devices.

The *Vader Application Overview* document contains a concise description of what Vader is used for and by whom. In summary, Vader provides a webservice and an administration GUI website, for a few CenturyLink customer support applications and personnel teams:

* QwestRx – this client uses the Vader webservices
* PollDSLAM – this client uses the Vader webservices
* Network Diagnostics Portal (NDP) – this client uses the Vader webservices
* Loop Provisioning Center (LPC) – this group uses the Vader GUI for ONT provisioning diagnostics
* Network Monitoring and Analysis (NMA) – this group maintains the *Vader OLT Database*

## Vader Application Philosophy

The Vader system has the following principles and objectives:

1. provide a webservice that serves as a middleware component to hardware devices
2. interpret device states into a logical and understandable meaning (eg. numeric codes into text)

## Vader Programming Concepts

Key Vader programming concepts are here, while the entirety of this document provides further details.

### Coding Philosophy

Ideally, the Vader code should follow programming best practices, with emphasis on:

* enterprise software industry standards
  + use of a concurrent versioning system (Git)
  + unit and integration testable (needs improvement)
  + agile methodology: staged releases
  + project management (integrated with corporate projects)
* modularization or object orientation
* code deduplication
* error free execution (no or minimal PHP warnings)
* reliable and efficient execution (in production environment)
* self documenting code and good code comments
* accurate and deduplicated user and programmer documentation
* well-formed system output (needs XML improvements)
* adequate logging and system diagnostics (needs improvement)
* adherence to multiple client requirements

### Programmer Skills

The following skills are mostly required for successful programming and administration of the Vader application:

* PHP coding knowledge (including OOP)
* Javascript coding (intermediate knowledge)
* Git version control and Eclipse IDE (or equivalent)
* SNMP/SOAP/XML knowledge
* Shell scripting abilities (bash)
* Apache server configuration and management
* MySQL queries
* Communications equipment knowledge, particularly Calix/Adtran GPON technologies

A Vader programmer’s time will be consumed with the following activities:

* PHP coding and debugging (15%)
* Project management, teleconferences, support consultations (15%)
* Vendor device analysis (15%)
* Technical writing (15%)
* Linux/Solaris administration and shell scripting (10%)
* Lab device testing (5%)
* XML programming
* Webservices programming
* Team meetings and corporate politics

The challenge is filling the gap as a (Vader application) middleman, between managers, programmers, and support applications (QwestRx, NDP, PollDSLAM, LPC) that want intelligent actionable information, from provisioned production customer end-equipment in the field (Calix, Adtran) and similar new equipment being planned for upgrades and tested in the labs. The obstacles are:

* those Vader customers that don’t understand how equipment is deployed (network architecture and process) or how devices are designed to be provisioned to the customer and the function of those devices (eg. parameter significance and meaning, device and port reset options); and so, can’t design good support applications.
* system architects, equipment vendors, and test lab technicians that don’t understand the value of creating good support applications, and so, don’t respond to the needs (production-like provisioned lab equipment) and requirements (precise device parameter specifications, SNMP!) of those support applications.
* the rollout of new equipment (to green-field or first office application (FOA) and even production!) before being tested or blessed.

### Vader Use Cases

Network Engineer 1A) Engineer (eg. LPC technician) uses Vader website to diagnose a network device.

Support Technician 2A) Technician uses support tool (QRx, NDP, PollDSLAM) to diagnose a customer problem.

Vader Administrator 3A) Administrator uses Vader website to add or change a Vader device configuration.

### Vader Instances

Test to Integration to Production.

The Vader test instance uses a simplified user authorization function, instead of the LDAP authorization that is used by the production instance. The integration and production instances use LDAP authorization, in the AD domain, so that any employee with a domain account can login to the Vader GUI website.

#### Vader Test Instance

https://10.0.20.3/vdsl/vader/login.php

The test instance runs on a dedicated server that resides in the 3rd floor Mineral Lab #4. The server is labeled and sits on a shelf in rack RR0283.37 (ie. row 3, rack 37) in the back left corner of the lab.

Since the test instance runs on a server that does not have LDAP server authorization, a separate test administrative account is used for login: ***hansolo***. The password is maintained in the vader.ini initialization file. There is also the non-privileged ***droid*** user account, with the password also maintained in the initialization file.

#### Vader Integration Instance

https://vdsltechsupp.uswc.uswest.com/vader-i/login.php

LDAP accounts are used for login to the Vader integration version.

#### Vader Production Instance

https://vdsltechsupp.uswc.uswest.com/vader/login.php

### Vader Configurations

Part of the Vader application configuration is set and stored in the *vader.ini* file.

Programmers can test and verify a Vader instance by accessing the *show-vader-config.php* webpage:

* https://< baseUrlVaderInstance >/admin/show-vader-config.php

Other CenturyLink applications, that are consumers of Vader webservices, can check the Vader support configuration by accessing the *vader-support.php* webservice:

* https://<baseUrlVaderInstance>/vader-support.php

### Vader Versioning With Git

Version control is implemented with the Git application.

Refer to the *Vader Git Guide* document for more details on using Git.

## Vader Users

Any CenturyLink employee that has an AD Domain account (username/password) and access to the corporate intranet, can login to the Vader application and use the GUI interface. And even more open (and concerning) is the Vader webservice, which is provided without any secure authorization protocol.

### QwestRx

QwestRx is a lagacy diagnostic and remedy tool used by customer support technicians. QwestRx calls both PollDSLAM (for requests) and Vader directly (for device bounces).

### PollDSLAM

PollDSLAM is a web application that provides support diagnostics, about many corporate devices and systems. PollDSLAM is maintained by Steven Wonchoba (refer to contact information in the appendix). The term PollDSLAM is somewhat of a misnomer, and a DSLAM is a hub for copper technology, but PollDSLAM (now) provides parameter information for GPON (fiber) technology.

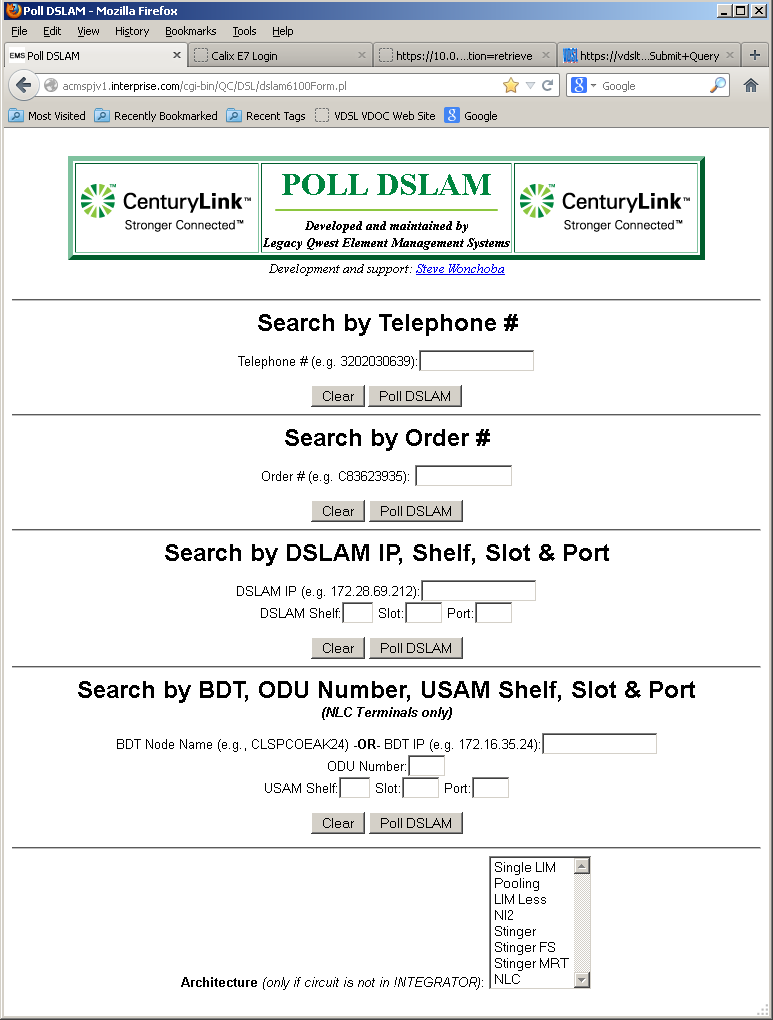
PollDSLAM makes webservice calls to Vader, to obtain diagnostic information for BPON and GPON equipment that is within the realm of Vader. Essentially, PollDSLAM may massage and convert some the XML information, but either displays the information, or passes the data on to its client systems. PollDSLAM only calls the Vader request.php method, and does not call the Vader bounce functions.

PollDSLAM can be accessed via a web browser at:

* http://acmspjv1.interprise.com/cgi-bin/QC/DSL/dslam6100Int.pl
  + (Not sure which server instance this is)
* http://suomp25n.qintra.com/cgi-bin/QC/DSL/dslam6100Int.pl?telephoneNum=4024319748
  + Customer’s telephone number as URL request parameter
* http://suomp25n.qintra.com/cgi-bin/QC/DSL/dslam6100Int.pl?telephoneNum=4024319748V
  + Added “V” character returns a raw XML response

PollDSLAM servers are:

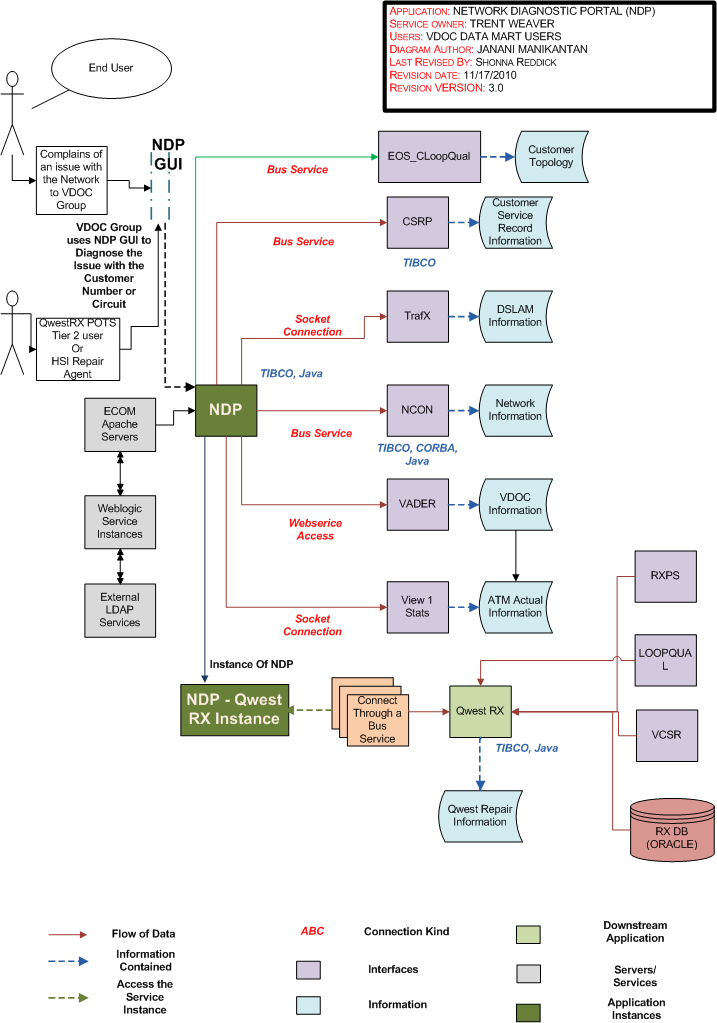
|  |  |  |
| --- | --- | --- |
| Suomd28i.dev.qintra.com (yosemite) | 151.117.41.239 | Development server; which is accessed by QwestRx and NDP developers. |
| suomp25n.qintra.com | 151.117.39.61 | Majority of production requests to Vader. |
| acmspjv1.interprise.com | 172.28.62.236 | Also a majority of production requests to Vader ; Wonchoba is trying to retire this machine. |
| sudnp01h.qintra.com (acmspjv2.interprise.com) | 151.119.110.70 |  |
| suomp63h.qintra.com | 151.117.39.22 | Very little, if any, usage to Vader. |
| wpfltrafx.nnet | 10.52.49.42 | Exclusive to Legacy CTL; most likely no Vader usage. |
| suomt80q.dev.qintra.com | 10.6.49.80 | Another development server; not used much. |



### Network Diagnostic Portal (NDP)

The Network Diagnostics Portal (NDP) is a Tier-N support application that provides capabilities similar to QwestRx. Because of the overlap with QwestRx, it has been stated that while NDP upgrades have been funded, that there are no planned corporate users, and so will eventually be obsoleted. Most of the NDP development is outsourced to a team located in Chennai, India (refer to contact information in the appendix).

The following diagram shows the NDP architecture:

**NDP Architecture Diagram**

### Loop Provisioning Center (LPC)

The Loop Provisioning Center (LPC) is part of the CIA group for service delivery. Orders for (ONT) hardware in the field are managed by the LPC. The LPC uses the Vader GUI for verifying ONT additions in the field. The LPC uses a EMS/CMS to initially discover a new ONT, and populate the correct address information.  After the EMS/CMS is configured by the LPC, then Vader is used to verify the ONT addition. As ONTs are added and serviced, field technicians will also call the LPC to have them run equipment checks using Vader.

The LPC is organized as follows:

* Denver team manages: AZ,CO,WY
* Minneapolis team manages: MN,NE,IA,ND,SD
* Salt Lake team manages: UT,AZ (outstate)
* Seattle team manages: WA,OR,ID

Refer to LPC contact information in the appendix.

### Network Monitoring and Analysis (NMA) Team

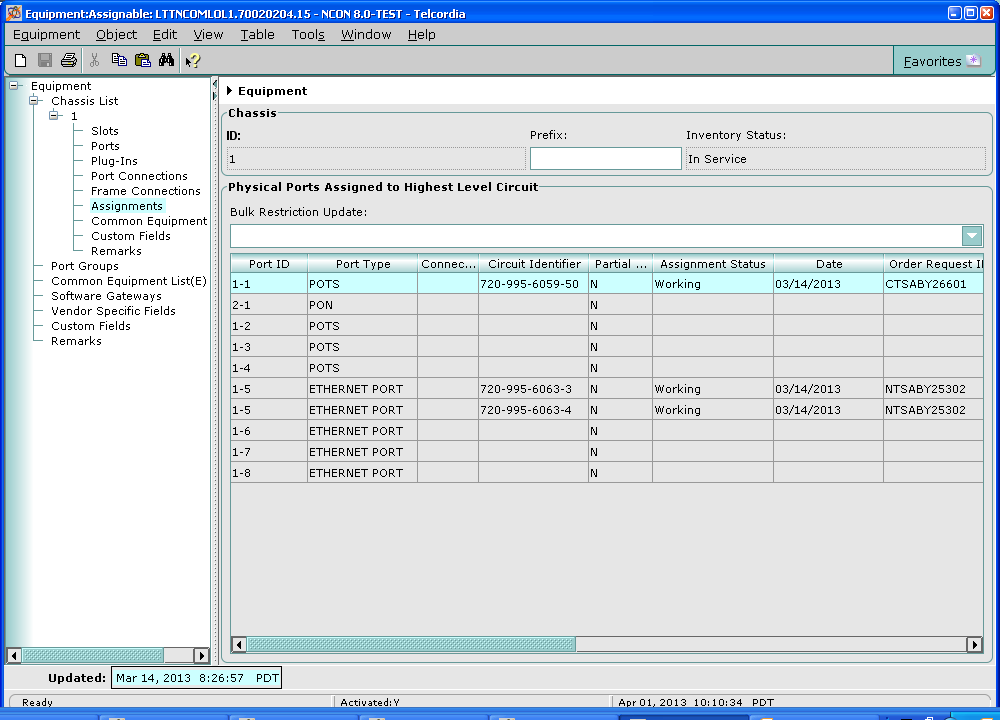
The *NMA Team* manages the Vader OLT database (as of July 2013, prior to that Mike Haakonson mostly managed the database). There are several NMA team members that have *Vader Advanced User* privileges, that grants them access to edit the OLT database (ie. access to the gpon-admin-edit.php and chassis-table-\*.php webpages).

Refer to the *GPON\_OLT\_CHASSIS* section of the *MySQL Database* section, for more information about the Vader OLT database. Refer to NMA Team contact information in the appendix.

## Other CenturyLink Systems

### NCON

The NCON application was originally developed by Telcordia and is now used by CenturyLink to perform the final provisioning of services on a customer’s device. NCON uses the ISYS database to get device and customer configuration information.



### ISYS

The ISYS is a database (or application?) that provides access to an extensive database of corporate information.

### Service Orders

Example service order XML:

    <HSIDetailList>  
        <HSIDetail>  
            <Tn>**4024319748**</Tn>  
            <Service>  
                <ServiceObjectRef>  
                    <ObjectReferenceID>**7705604**</ObjectReferenceID>  
                    <ObjectTypeName>**MegasubscriberService**</ObjectTypeName>  
                    <SystemReferenceID>**ISYS**</SystemReferenceID>  
                </ServiceObjectRef>  
                <ServiceStatus>**ACTIVE**</ServiceStatus>  
                <ServiceId>**SV7705604**</ServiceId>  
                <ServiceCategoryType>**RADSL DMT**</ServiceCategoryType>  
                <PipeUpSpeed>**50144K**</PipeUpSpeed>  
                <PipeDownSpeed>**100128K**</PipeDownSpeed>  
            </Service>  
            <ServiceOrderList>  
                <ServiceOrder>  
                    <ServiceOrderObjectRef>  
                        <ObjectReferenceID>**57634202**</ObjectReferenceID>  
                        <ObjectTypeName>**ServiceOrder**</ObjectTypeName>  
                        <SystemReferenceID>**ISYS**</SystemReferenceID>  
                    </ServiceOrderObjectRef>  
                    <ServiceOrderId>**D81614168**</ServiceOrderId>  
                    <ServiceRequestId>**DNNAFT45101**</ServiceRequestId>  
                    <AdminStatus>**Completed**</AdminStatus>  
                    <ProcessStatus>**Action Not Applicable**</ProcessStatus>  
                    <RequestedAction>**Disconnect**</RequestedAction>  
                    <IssuedDate>**2010-12-15**</IssuedDate>  
                    <DueDate>**2010-12-15**</DueDate>  
                    <SubscriberName>**EYTALIS,KATE**</SubscriberName>  
                    <Address1>**105 MAHER DR BOYS TOWN**</Address1>  
                </ServiceOrder>

                <ServiceOrder>  
                    <ServiceOrderObjectRef>  
                        <ObjectReferenceID>**93456012**</ObjectReferenceID>  
                        <ObjectTypeName>**ServiceOrder**</ObjectTypeName>  
                        <SystemReferenceID>**ISYS**</SystemReferenceID>  
                    </ServiceOrderObjectRef>  
                    <ServiceOrderId>**N45069486**</ServiceOrderId>  
                    <ServiceRequestId>**NNNCJZ41701**</ServiceRequestId>  
                    <Revision>**A**</Revision>  
                    <AdminStatus>**Completed**</AdminStatus>  
                    <ProcessStatus>**Completed**</ProcessStatus>  
                    <RequestedAction>**Installation**</RequestedAction>  
                    <IssuedDate>**2013-05-08**</IssuedDate>  
                    <DueDate>**2013-05-13**</DueDate>  
                    <SubscriberName>**HUTCHINS,MARK**</SubscriberName>  
                    <Address1>**16006 SAHLER CIR OMAHA**</Address1>  
                </ServiceOrder>  
            </ServiceOrderList>  
        </HSIDetail>  
    </HSIDetailList>  
</HsiDetailsResponse>

#### Service Order Device Tags

The following text is paraphrased from *Integrator* documentation (and an email) from Sheryl Wiggins.

Service orders show a *Device Tag* as: OMAHNECSOL1.70050204.14

**SMVLNJMTOL1.10240101.1**

^----------^^^^^^ ^ ^

         |    | | \>- (1-32 **ONT** #)

         |    | |

          |    | \>-- **OLT Port (01,02…), OLT Shelf Number assumed to be 1 only when the OLT Node Number exists**

         |    |>---- OLT Slot

         \>------- **Separator (.) and either of the following:**

* **the OLT Node Number: 4 digits zero filled**
* **the OLT Shelf Number: Device Shelf Indicator (example:‘70’) + (01,02..)**

\>------------- Shelf CLLI (***w/OLT PON Card)***

The 4-digit number in hot pink below could mean one of 2 different things, either an OLT Node Number OR a Device Shelf Indicator + Shelf Number.

 In the case of Adtran we get OLT Node Number and we have to assume the shelf is 1. In the case of Calix, we get a Shelf Device Indicator (70) which tells us to read the next digits as a shelf.

### Integrator

Integrator is a source of record for layer-two inventories which supports DSL services. Upstream systems send requests to add, change and delete DSL services. Integrator provides services to systems requiring information on DSL services.

# Network Components

## CenturyLink (CTL) Management Systems (EMS and CMS)

The remote terminal (RT) devices that are managed by Vader, are also managed by vendor specific management systems. For Adtran devices, an Element? Management Systems (EMS) is used. For Calix devices, a Calix Management System (CMS) is used. The specific CTL management system used by a particular device is specified in the Vader database (table GPON\_ OLT\_CHASSIS).

For Calix devices, Vader makes SOAP calls to a Calix management system to acquire device configuration information. This information is used to partially formulate the Vader client responses. (Refer to adtran/? and calix/gpon-request.php for code details).

Vader requires that Calix CMS management systems to be (initially) configurated with a unique Vader username and password. This allows Vader to make SOAP requests to the CMS. Refer to ? for more details about account information.

For Adtran devices, most of the configuration information obtained by Vader, is obtained by making SNMP calls to each OLT. Vader does obtain ONT alarm information from the EMS. (Refer to adtran/? for code details).

At some point in time (2013/2014), Adtran will be tranisitioning to a more powerful AOE (Adtran Operating Environment) application. The AOE will add a database and encompass service turn-up and device management. Within the Adtran company, the AOE group is merging with EMS group. The AOE will include a browser interface (like Calix). This management application change could affect Vader and enable a better interface (more like the Calix CMS SOAP interface).

## Remote Terminal Devices

… basically, ONTs are terminators that reside at a customer site. Many ONTs aggregate at an OLT. The following simple diagram shows the basic remote terminal configuration:

CTL uses two Vendors for RT equipment: Adtran and Calix.

Network Diagram (by Dennis Cheng)

DSLAM Cross Connect Concept Diagram (by Dennis Cheng)



Refer to the Vader Interface Specification for Vader cross connect details.

### Adtran and Calix Differences

While the GPON technology is somewhat standardized, there are subtle differences between the different vendor devices (OLTs and ONTs).

Vader uses different PHP modules to interface with the devices. For Adtran, Vader uses SNMP to query diagnostic data and perform ONT bounces (?). To obtain ONT alarm conditions, Vader calls an Expect script (adtran/get\_gpon\_ont\_alarms.exp) that runs a shell script on the EMS server.

For Calix, a couple of SNMP calls are made to identify the OLT (TID), but then the diagnostics data queries and ONT bounces are all performed with SOAP calls to the CMS. Refer to the *SOAP* section for more details.

Refer to the *Vader Code Structure* section for how the code separates the different devices.

### Adtran

Interfaces to Adtran devices were part of the original Vader design. The Vader PHP code for Adtran is not as modular (compared to the more recent Calix code), and extensively calls SNMP queries to each ONT device (see ftth/adtran/gpon-request.php).

Refer to the appendix for Adtran vendor information.

#### Adtran OLTs

Only Adtran has the provisioning for RF video, not Calix. Refer to PHP source in the ftth/adtran/web/gpon-prov-rfvideo.php file.

OLTs: TA5000, TA5006

#### Adtran ONTs

ONTs: TA324, TA334, TA362, TA362S, TA374

The following table shows Vader support, lab availability, and device port configuration for Adtran ONTs (as of 130312):

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Adtran ONT  Model | Vader  Support | Lab  Availability | POTS | Data | RF  Video | DS1 | Configuration |
| TA324 | 7.0 | Mineral | 2 | 4 |  |  | Indoor |
| TA324RG |  |  | 2 | 4 |  |  | Residential Gateway |
| TA334 | 7.0 |  | 2 | 4 | 1 |  | Indoor |
| TA351 | 7.0 |  |  |  |  |  |  |
| TA352 |  | Mineral | 2 | 2 |  |  |  |
| TA362 | 7.0 | Mineral | 2 | 2 | 1 |  |  |
| TA362S | 7.0 | Mineral | 2 | 2 | 1 |  |  |
| TA372 |  |  | 8 | 2 |  |  |  |
| TA372R |  |  | 8 | 2 | 1 | 4 |  |
| TA374 | 7.0 |  | 4 | 4 |  |  |  |
| TA385 |  |  | 12 | 12 | 10 + 2 |  | Rack Mounted |
|  |  |  |  |  |  |  |  |

The following table is from an Adtran (PDF) document that shows the 300-series ONT part numbers and descriptions.

Total Access 300 Series -Single Family Unit ONT (Issue Date: July 2012 - Document P/N: 612877GONT-22A)

|  |  |
| --- | --- |
| **Adtran Part Number** | **Description** |
| 1287701G1 | Total Access 351 2nd Generation 2 POTS/1 Ethernet SFU ONT |
| 1287702G1  1287702G1Q | Total Access 352 2nd Generation 2 POTS/2 Ethernet SFU ONT |
| 1287702G3 | Total Access 352H 2nd Generation SFU ONT |
| 12877711G1 | Total Access 361 2nd Generation 2 POTS/1 Ethernet/1RF Overlay SFU ONT |
| 1287712G1 | Total Access 362 2nd Generation 2 POTS/2 Ethernet/1 RF Overlay SFU ONT |
| 1287712G3 | Total Access 362H 2nd Generation SFU ONT |
| 1287715G1 | Total Access 362R 2nd Generation 2 POTS/2 Ethernet/1 RF Return SFU ONT |
| 1287735G1 | Total Access 324 SFU ONT |
| 1287735G2 | Total Access 324 SFU ONT with UPS |
| 1287736G1 | Total Access 334 SFU ONT |
| 1287736G2 | Total Access 334 SFU ONT with UPS |

### Calix

Interfaces to the newer Calix devices were added to Vader version 7.0 (2012Q2). For Calix, Vader makes SOAP calls to a Calix Management System (CMS) to get ONT device information (as opposed to SNMP calls directly to each ONT device as in the case of Adtran devices). SOAP calls are in source files:

* ftth/calix/gpon-request.php
  + Retrieve E7 chassis parameters
  + Retrieve PON card parameters
  + Retrieve PON port parameters
  + Retrieve ONT common parameters
  + Include ONT model-dependent routines (via include of get-ont-state-700-series.php)
  + Retrieve alarms (via include)
* ftth/calix/get-ont-state-700-series.php
* ftth/calix/gpon-bounce.php

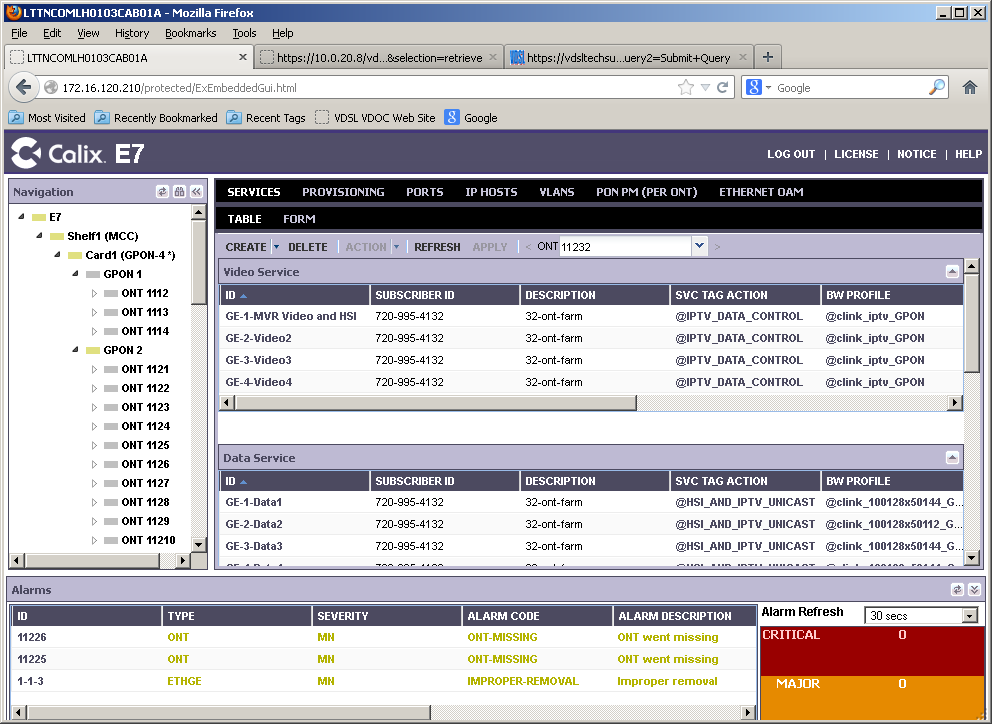
Refer to the appendix for Calix vendor information.

#### Calix OLTs

OLTs: E7-2, C7 (not used)

##### Calix E7 OLT

A Calix E7 OLT can be accessed from a web browser, but requires a login name and password.



#### Calix ONTs

Refer to the Vader Interface Specification for a similar Calix ONT support table.

The following table shows Vader support, lab availability, and device port configuration for Calix ONTs (as of 130311):

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Calix ONT  Model | Vader  Support | Lab  Availability | POTS | GE  Ethernet | 1GHz RF  Video | DS1 | RF  Avo | RF Hot Video | FE Fast  Ethernet |
| 711GE | 7.0 | Mineral | 2 | 2 |  |  |  |  |  |
| 716GE | TBD | On Order | 2 | 4 |  |  |  |  |  |
| 716GE-I | 7.0 | Mineral | 2 | 4 |  |  |  |  |  |
| 717GE | 7.0 | Mineral | 4 | 4 |  |  |  |  |  |
| 721GE | TBD |  | 2 | 2 | 1 |  |  |  |  |
| 726GE | TBD |  | 2 | 4 | 1 |  |  |  |  |
| 726GE-I | TBD |  | 2 | 4 | 1 |  |  |  |  |
| 727GE | 7.0 | None | 4 | 4 | 1 |  | 1 |  |  |
| 762GX | 7.0 | Mineral | 8 | 8 | 1 |  | 1 | 1 |  |
| 763GX | TBD |  | 8 | 8 | 8 |  |  |  |  |
| 763GX-R | TBD |  | 8 | 8 | 8 |  |  |  |  |
| 766GX-R | 7.0 | Mineral | 8 | 4 | 1 | 8 | 1 | 1 |  |

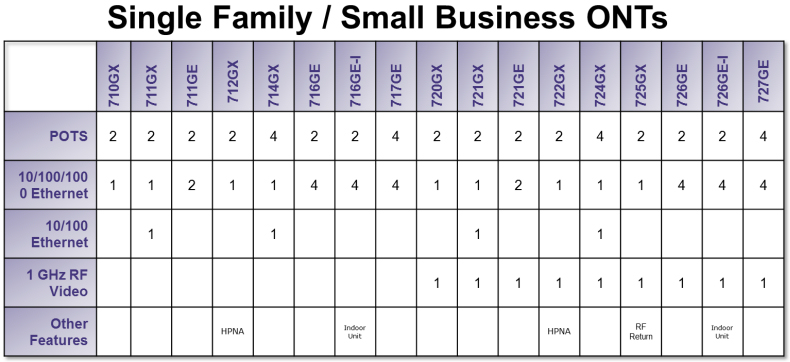
<http://portal.calix.com/systems/p-series/calix_700GE_ONTs.html>

**Calix 716GE ONT**



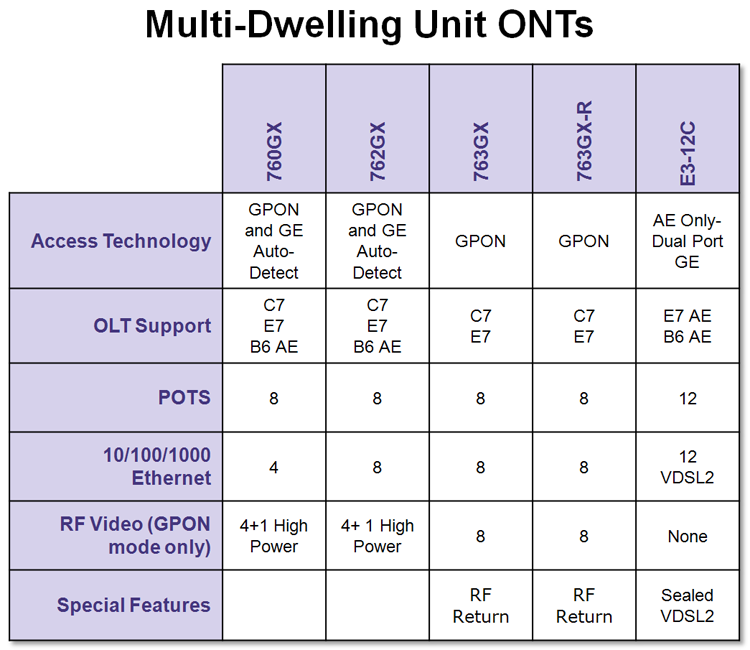
##### Single Family Unit and Small Business ONTs

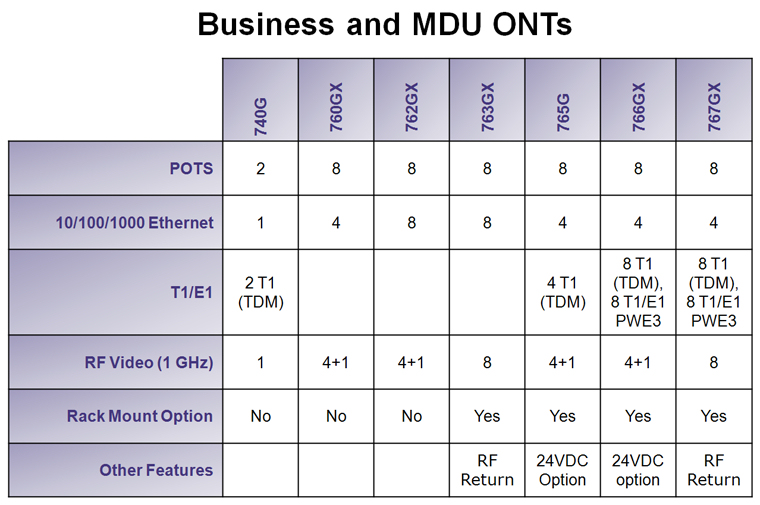
<http://portal.calix.com/systems/p-series/calix_sfu_ONTs.html>



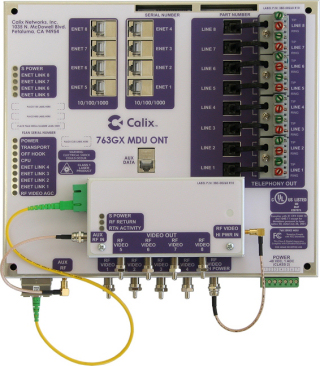
##### Multi Dwelling Unit ONTs

<http://portal.calix.com/systems/p-series/calix_mdu_ONTs.html>



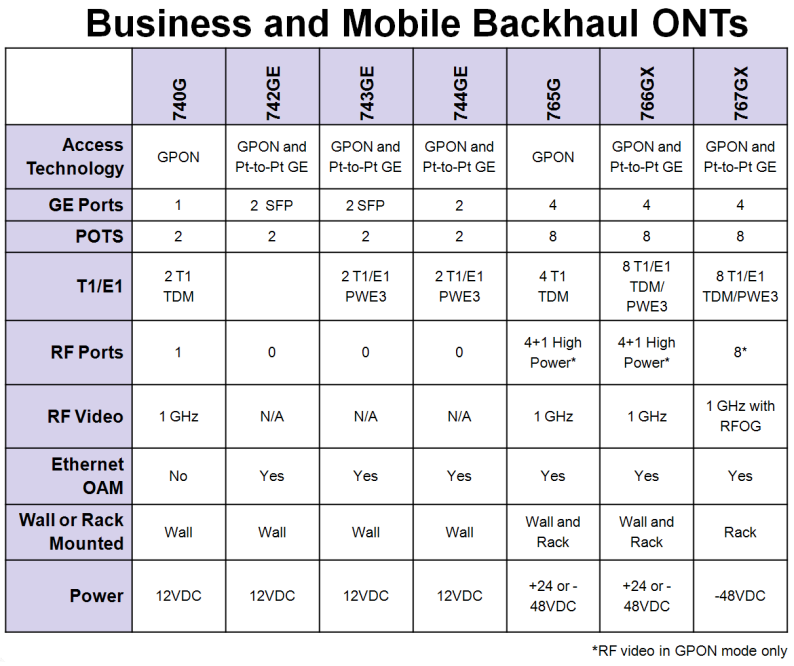


**Calix 763GX ONT**

****

##### Business Service and Mobile Backhaul ONTs

<http://portal.calix.com/systems/p-series/calix_business_and_mobile_backhaul_ONTs.html>



##### Calix 760G Series ONTs

|  |  |
| --- | --- |
| **Calix 765G ONT – Wall Mounted** | **Calix 766GX ONT** |
| Calix 760G ONT | Calix 766GX ONT |

**Calix 766GX-R ONT Rack Mounted**



# Client Interfaces

Refer to the Vader Interface Specification for more details on Vader requests and responses.

## Vader Requests

Request variables:

service = (str bpon|gpon)

oltip = (str xxx.xxx.xxx.xxx) - IP Address of TA5K or TA5006 Chassis

aid = (str ONT-AID)

aid must have the following format:

Alcatel ONT AID structure

ONT - <RACK> - <SHELF> - <PONCARD> - <PORT> - <ONT #>

1 1 1-22 1-2 1–32

AdTRan ONT AID structure

ONT - <RACK> - <SHELF> - <PONCARD> - <PORT> - <ONT #>

1 1 1-22 1-2 1–32

Calix ONT AID structure

ONT - <SHELF> - <SLOT> - <PORT> - <ONT #>

1-10 1-2 1-4 1–64

debug = (str yes|no)

use debug=no for straight xml responses to PollDSLAM and NDP

Debug mode will return html and xml, and will be used for troubleshooting purposes via a web browser...

selection = (str retrieve)

bounce-coax is being left in the interface document and design, in case it is decided to do RF-Video

Example:

http://vdsltechsupp.uswc.uswest.com/vader/ftth/request.php?service=gpon&oltip=172.16.120.242&aid=ONT-1-1-16-1-2&debug=no&selection=retrieve

## Vader Responses

### XML Responses

Refer to the Vader Interface Specification document for definitive XML response definitions. This section adds specific response information that is more useful to a programmer.

For Calix devices, Vader will respond with a <RESULT> root element name (for all request-type responses), and for Adtran devices, Vader will respond with a <NDP-RESULT> as the root element name (for all request-type responses).

### Device Status

Refer to the fttp/calix/gpon-request.php file for the following code:

$ont\_admin\_state = $rtrv\_E7ONT\_response['data']['top']['object']['admin'];

$ont\_oper\_state = $rtrv\_E7ONT\_response3['data']['top']['object']['children']['child']['op-stat'];

#### Calix ONT Administrative State

The administrative state is a value that can be set; ie. it is a writeable value, typically by a technician, as the device is undergoing reconfigurations. The following table shows the range of values for the Calix ONT administrative state:

<RESULT><ONT-STAT><ONT-ADMIN-STATE> - for each single ONT

<RESULT><ONTPOTS-PORT-STAT><POTS-PORT-*n*><PROVISIONED-STATE> - for each port *n* on an ONT

<RESULT><T1E1-PORT-STAT><T1E1-PORT-*n*><PROVISIONED-STATE> - for each port *n* on an ONT

<RESULT><ETHERNET-PORT-STAT><GE-PORT-*n*><PROVISIONED-STATE> - for each port *n* on an ONT

<RESULT><RFVIDEO-PORT-STAT><RFVIDEO-PORT-*n*><PROVISIONED-STATE> - for each port *n* on an ONT

<RESULT><AVO-PORT-STAT><AVO-PORT-*n*><PROVISIONED-STATE> - for each port *n* on an ONT

<RESULT><HOTRF-PORT-STAT><HOTRF-PORT-*n*><PROVISIONED-STATE> - for each port *n* on an ONT

|  |  |
| --- | --- |
| **Value** | **Condition** |
| enabled | ONT or port is enabled with alarm generation. |
| enabled-no-alarms | ONT or port is enabled without alarm generation. |
| disabled | ONT or port is disabled, either by a user or the system. |
| “” | State was not successfully returned from a CMS query. |

#### Calix ONT Operational State

The operational state is a value that is determined by the system, and is not set by a user. Basically, it tells when there is a *disabled* administrative state (as above), whether it was due to a user or the system. The following table shows the range of values for the Calix ONT operational state:

<RESULT><ONT-STAT><ONT-OPER**ATIONAL**-STATE> - for each single ONT

<RESULT><ONTPOTS-PORT-STAT><POTS-PORT-*n*><OPERATIONAL-STATE> - for each port *n* on an ONT

<RESULT><T1E1-PORT-STAT><T1E1-PORT-*n*><OPERATIONAL-STATE> - for each port *n* on an ONT

<RESULT><ETHERNET-PORT-STAT><GE-PORT-*n*><OPERATIONAL-STATE> - for each port *n* on an ONT

<RESULT><RFVIDEO-PORT-STAT><RFVIDEO-PORT-*n*><OPERATIONAL-STATE> - for each port *n* on an ONT

<RESULT><AVO-PORT-STAT><AVO-PORT-*n*><OPERATIONAL-STATE> - for each port *n* on an ONT

<RESULT><HOTRF-PORT-STAT><HOTRF-PORT-*n*><OPERATIONAL-STATE> - for each port *n* on an ONT

|  |  |
| --- | --- |
| **Value** | **Condition** |
| enable | ONT or port is determined to be enabled and operational. |
| user-disable | ONT or port has been disabled by a user. |
| sys-disable | ONT or port has been disabled by the system. |
| “” | State was not successfully returned from a CMS query. |

#### Adtran ONT Administrative State

<NDP-RESULT><ONT-STAT><ONT-ADMIN-STATE>

1. In Service
2. Out Of Service UAS
3. Out Of Service MA
4. Missing (unable to retrieve the ponport service state, value missing)

#### Adtran ONT Operational State

<NDP-RESULT><ONT-STAT><ONT-OPER-STATE>

1. Initializing
2. Discovering
3. Discovered
4. Rejected
5. Up
6. Down
7. Unknown (unable to determine the ont operational state, value missing)

### Device Bounces

Refer to the Vader Interface Specification for device behavior and response details.

Delete the following text, as it’s now included in the 8.0.1 interface specification.

(The following was an email response on February 5th, 2013)

I spent much of my day today, trying to reproduce the *bounce-to-alarm* scenario, on the Vader test system; but I haven’t yet reproduced it. So, I will answer questions, based on my research and findings, so far.

It should be noted that, all device bounce actions to Vader, are sent to a CMS (Calix Management System) server, which then forwards the bounce commands to the Vader-requested Calix OLT device. After the ONT has been bounced (either a reset or reboot), the ONT takes some period of time to come back online; Vader tells us it is 2 minutes, but it depends on the device. During that time, the CMS considers the ONT to be in an alarmed state, until the CMS can further determine that the ONT is back online.

Yes, based on the above explanation, the ONT being in an alarmed state, is valid.

After the wait period, the device should come back online (after the reset or reboot). At that time, a Vader device status request (ie. request.php) will no longer return an alarm condition, but will respond with the current device online status. Until such time, the CMS will continue to report the alarm condition. (In a rare condition, maybe the ONT never comes back online.)

Note: there can also be specific port bounces (eg. bounce-pots-port-4); we haven’t discussed those, and I’m not sure you are yet testing those Vader capabilities.

# Vader Administrator Website

Refer to the *Vader Application Overview* document to get a general idea of the website capabilities and functions, from a Vader client perspective. Yet, the website

Insert website layout diagram here.

# Vader Application Administration

View the *Vader configuration* webpage (admin/show-vader-config.php) to review Vader instance configuration parameters, that pertain to this section. Refer to the *Vader Configurations* section of this document for details about the webpage.

## Vader Logfiles

Aside from the Apache logfiles, Vader writes to dedicated logfiles.

### XML Response Logs

$fileAdslChangeSpeedLog = "$directoryAdsl/change-speed.log"; // **TODO**: This may be obsolete? (See view1/adsl for working code.)

$fileFtthBponBounceLog = "$directoryFtth/bpon\_bounce.log";

$fileFtthBponLog = "$directoryFtth/bpon\_ftth.log";

$fileFtthGponBounceLog = "$directoryFtth/gpon\_bounce.log"; // Only logs (infrequent) bounces by adtran/gpon-bounce.php

$fileFtthGponLog = "$directoryFtth/gpon\_ftth.log"; // Logs Adtran retrieves, and all Calix transactions

$fileFtthGponWebBounceLog = "$directoryFtth/gpon\_web\_bounce.log"; // Only logs (frequent!) bounces by adtran/web/gpon-web-bounce.php

$fileFtthManageBattAlarmLog = "$directoryFtth/manage\_batt\_alarm.log"; // **TODO**: This may be obsolete?

$fileFtthManageBattAlarmOldLog = "$directoryFtth/manage\_batt\_alarm.log.old"; // **TODO**: This may be obsolete?

Refer to the rotate\_vader\_logs.ksh script, that is executed by root’s crontab on Saturday/Sunday midnight.

Listing of Vader 7.0 logfiles; note that some logfiles are obsolete with ancient timestamps, and that the *.old* files are the result of a one-week log rotation scheme:

aa53378@vdsltechsupp: find vader-p-7.0/ -name "\*log" -ls -o -name "\*log.old" -ls | sort -k11

122637 232 -rw-rw-rw- 1 webuser staff 226843 Nov 20 2009 ../vader-p-7.0/adsl/change-speed.log

122545 40 -rw-rw-rw- 1 webuser staff 40257 Nov 21 2009 ../vader-p-7.0/adsl/margin-adjust.log

110904 5 -rw-rw-rw- 1 webuser staff 4311 Jun 15 2012 ../vader-p-7.0/ftth/adtran/web/gpon\_ftth.log

104930 1 -rw-rw-rw- 1 webuser staff 902 Apr 15 12:10 ../vader-p-7.0/ftth/bpon\_bounce.log.old

104939 12 -rw-rw-rw- 1 webuser staff 11796 May 30 14:55 ../vader-p-7.0/ftth/bpon\_ftth.log.old

104931 2 -rw-rw-rw- 1 webuser staff 1522 Jun 3 09:10 ../vader-p-7.0/ftth/gpon\_bounce.log

104923 26 -rw-rw-rw- 1 webuser staff 26494 Jun 1 12:14 ../vader-p-7.0/ftth/gpon\_bounce.log.old

104908 160 -rw-rw-rw- 1 webuser staff 153190 Jun 3 11:58 ../vader-p-7.0/ftth/gpon\_ftth.log

104926 4744 -rw-rw-rw- 1 webuser staff 4843479 Jun 1 19:13 ../vader-p-7.0/ftth/gpon\_ftth.log.old

104937 1240 -rw-rw-rw- 1 webuser staff 1257222 May 31 10:18 ../vader-p-7.0/ftth/gpon\_web\_bounce.log

105094 1 -rw-rw-rw- 1 webuser staff 423 Jun 3 07:40 ../vader-p-7.0/ftth/manage\_batt\_alarm.log

105948 880 -rw-rw-rw- 1 webuser staff 888609 Jun 3 07:40 ../vader-p-7.0/ftth/manage\_batt\_alarm.log.old

105940 1432 -rw-rw-rw- 1 webuser staff 1455541 Feb 25 2010 ../vader-p-7.0/ftth/mass\_batt\_disable.log

105950 46 -rw-rw-rw- 1 webuser staff 46151 Feb 25 2010 ../vader-p-7.0/ftth/mass\_batt\_enable.log

104921 83 -rw-rw-rw- 1 webuser staff 84800 Mar 13 2009 ../vader-p-7.0/ftth/mass\_bounce.log

128356 39 -rw-r--r-- 1 webuser staff 39193 Apr 12 2012 ../vader-p-7.0/soap/lib/changelog

104928 3 -rw-rw-rw- 1 webuser staff 2182 Jun 3 10:00 ../vader-p-7.0/vdsl/xconnlu.log

104910 16 -rw-rw-rw- 1 webuser staff 15688 Jun 1 01:00 ../vader-p-7.0/vdsl/xconnlu.log.old

### Error Logs

* <instanceBaseDir>/log/error.log

## Administrator Notifications

Emails are sent for error conditions:

* OLT TID Mismatch (triggered by a GUI or webservice request)

Refer to the *PHP Mail Usage* section for coding details.

## Service Disabling

Part of the Vader website GUI has a capability to disable certain Vader services. Only Vader administrators are shown the Vader service controls. These service controls are rarely (if ever) used, as Vader should always be available to support production services. The services that can be controlled are:

* BPON service
* Calix GPON service
* Adtran GPON service

The services are turned off, by using the Vader administrator’s interface, which creates a service *lockfile*. If the lockfile exists, then the services are not available to users.

## Vader User Documentation

The Vader documentation is composed of Word and text documents, and is (ideally) stored in the Git repository, or alternatively maintained on the Vader administrator’s workstation. All of the documentation is thereby released in the *vader/doc* directory.

Scripts reside on the test server to manage the documentation:

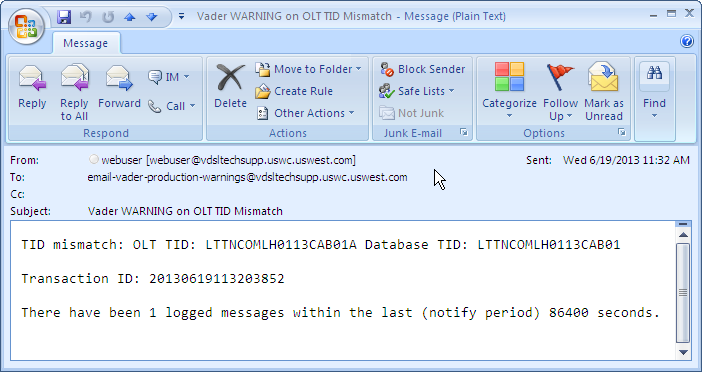
* <testSever>:~vader/bin/copy-vader-docs-from-doc-server.sh
* <testSever>:~vader/bin /copy-vader-docs-to-integration-server.sh

## Diagnosing Vader Errors

### Built-in Error Management

#### OLT TID Mismatch

A TID mismatch is (only) detected when an ONT diagnostics request is made (ie. gpon-web-provision.php or request.php). (Also refer to the run-vader-olts-snmp-check.pl which also verifies TID matches.)



Use the transaction ID in the warning email, to optionally grep for additional error information:

aa53378@vdsltechsupp: grep 20130619113203852 /opt/apache/htdocs/vdsl/vader/log/\*

log/error.log:20130619\_113203 20130619113203852 10.1.204.74 gpon-web-provision TID mismatch: OLT TID: LTTNCOMLH0113CAB01A Database TID: LTTNCOMLH0113CAB01

log/transaction.log:20130619\_113203 20130619113203852 10.1.204.74 gpon-web-provision Calix 172.16.120.210 ONT-1-1-2-34

log/web-access.log:20130619\_113206 20130619113203852 10.1.204.74 gpon-web-provision Page access by aa53378 as Jeff Sheffel

## Remedy IT Helpdesk Tickets Against Vader

Any *NTM Remedy* tickets should be place in the *VDOC SERVER TEAM (77008)* group. This group has the Vader team manager (Rick Lorenz) as a member. Note that *NTM Remedy* is different and mutually exclusive from the ITHD Remedy, to which there can be no Vader tickets assigned(?).

# Application Framework

## Apache

Redirect to SSL, only one entry in configuration files:

httpd-vhosts.conf: Redirect Permanent /vader/ftth/web-request-query.php https://vdsltechsupp.uswc.uswest.com/vader/ftth/web-request-query.php

### Apache Logfile Analysis

Beta code has been developed to analyze Apache logfiles. The code is written in Perl and resides in the Git repository.

A webpage tracks Vader usage, by batch analysis of the weekly Apache logfiles. See…

## PHP

Versions:

* Production - PHP 5.2.9
* Test - PHP 5.3.9 (OpenCSW for Solaris)

PHP configuration details for each server can be obtained at:

* http://10.0.20.8/phpinfo.php
* http://vdsltechsupp.uswc.uswest.com/phpinfo.php

If the PHP version becomes 5.5 or greater, then the mysql\_query() calls will need to be upgraded to mysqli or PDO. Refer to the php.net mysql\_query reference webpage for details. Refer to planned upgrades?

### PHP Runtime Errors

Coding efforts are made to eliminate PHP runtime errors, such that most PHP runtime errors should be considered as real PHP or Vader application problems. Yet, there may be some set of PHP runtime errors that are known issues or can be considered as negligible.

PHP runtime error output is determined by: the PHP application code, the PHP configuration (php.ini), and the Apache (mod\_php) configuration. Each Vader server (test versus production servers) will have different configurations, which is preferred. The test instance should show most (or all) runtime errors.

For the test instance server, PHP errors are written to:

* /opt/csw/apache/var/log/error\_log

For the integration and production instances server, PHP errors are written to:

* ~rlorenz/logs/php\_errors

Refer to the Vader Debugging Guide for more information about debugging PHP runtime errors.

### PHP Mail Usage

Some Vader warnings and errors are communicated to administrators, through the operating system’s sendmail utility. Sendmail aliases are used by Vader (see config.php) to control group distribution. Refer to the aliases manpage, and the *Vader Install Guide* for more details.

## MySQL Database

vdsltechsupp: /usr/local/mysql

|  |  |
| --- | --- |
| **MySQL Data Directory (in /usr/local/mysql/)** | **Size in KB (on 2013-04-25)** |
| data/PRMDATA | 3549844 |
| data/VNOC | 22219 |
| data/VNOC\_DEV | 12654 |
| data/mysql | 864 |
| data/MIRANDA | 482 |
| data/CHANGE | 273 |
| data/CSG | 79 |
| data/test | 1 |
| data | Total Size: 4061811 |

### Database Schema

### Database Backups

### Vader Tables

Refer to the *Crontabs* section of this document to see details on how these tables are populated.

#### GPON\_OLT\_CHASSIS

This table is known as the *Vader OLT database*. The table is maintained by human interaction, using the Vader website. Other DBA tools such as mysql, Oracle’s SQL Developer, and SQLyog, can be used to edit the table. Privileged administrators (*Vader Administrators* and *Advanced Users*) can create, update, and delete table entries.

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Sample Value** | **Description** |
| ID | 8 |  |
| STATE | Washington |  |
| TECHNOLOGY | GPON |  |
| VENDOR | AdTran |  |
| CENTRAL\_OFFICE | BLLVWAGL |  |
| OLT\_CLLI | BLLVWAGLOL2 |  |
| OLT\_TID | BLLVWAGLOL202020025C |  |
| OLT\_IP\_ADDRESS | 172.21.7.97 |  |
| COT\_TID | BLLVWAGLG0202020025A |  |
| COT\_IP\_ADDRESS | 172.21.7.94 |  |
| RF\_VIDEO\_VLAN | 4050 |  |
| EMS\_IP\_ADDRESS | 10.6.8.54 |  |
| NCON\_SOURCE\_DB | NE |  |
| STATUS | Production |  |
| CUID | rlorenz |  |
| TIMESTAMP | 012-07-02 215:14:58.0 |  |

#### GPON\_ONT\_EQUIPMENT

This table tracks known customer (Adtran) ONT devices; **created hourly**, during extended working hours (05-21), by the *get\_gpon\_equipment.ksh* script.

|  |  |
| --- | --- |
| **Date** | **Number of Records** |
| 2013-04-23 | 4510 |
| 2013-01-xx | 4163 |

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Sample Value** | **Description** |
| PULL\_DATE | 2013-01-21 13:57:56.0 |  |
| OLT\_CLLI | HRHLCOABOL2 |  |
| ONT\_AID | ONT-1-1-1-1-32 | Parsed from ONT\_BIN\_IFINDEX in SQL. “ONT-1-1-“ is hardcoded. |
| ONT\_BIN\_IFINDEX | 1000000000010011000000000000000 | Converted from ONT\_IFINDEX in SQL. |
| ONT\_IFINDEX | 1074364416 |  |
| SERIAL | ADTN104000be | **Obtained with snmpwalk on**: 1.3.6.1.4.1.664.6.10000.76.1.1.2.3.1.1 |
| ADDRESS | 10313 BELVEDERE LN ONT 32 | 1.3.6.1.4.1.664.6.10000.76.1.1.2.3.1.2 |
| CLEI | BVM8K00ERA | 1.3.6.1.4.1.664.6.10000.76.1.1.3.1.1.5 |
| HW\_VER | A | 1.3.6.1.4.1.664.6.10000.76.1.1.3.1.1.13 |
| ONT\_PARTNO | 1187712G2 | 1.3.6.1.4.1.664.6.10000.76.1.1.3.1.1.12 |
| ONT\_SWVER | A3.11.A | 1.3.6.1.4.1.664.6.10000.76.1.1.2.3.1.7 |
| ONT\_ACTIVE\_SWVER | A3.11.A | 1.3.6.1.4.1.664.6.10000.76.1.1.3.1.1.10 |
| BATTERY\_BACKUP | 1 | 1.3.6.1.4.1.664.6.10000.76.1.1.2.3.1.3 |
| ONT\_ADMIN\_STATE | 1 | 1.3.6.1.4.1.664.6.10000.76.1.1.2.3.1.4 |
| ONT\_OPER\_STATE | 4 | 1.3.6.1.4.1.664.6.10000.76.1.1.3.1.1.7 |
| ONT\_UPTIME | 69 days,08:21:53 | 1.3.6.1.4.1.664.6.10000.76.1.1.3.1.1.6 |

#### GPON\_CUSTOMER\_NTWK

Created once a day, every morning, by the *get\_gpon\_ncon\_info.ksh* script. The table is created by joining tables **GPON\_ONT\_EQUIPMENT** and **NCON\_FTTH\_GPON.** At the beginning of 2013 were 4167 records.

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Sample Value** | **Description** |
| PULL\_DATE | 2013-01-21 07:21:04.0 |  |
| OLT\_CLLI | RVTNUTCSOL2 |  |
| ONT\_AID | ONT-1-1-1-1-21 |  |
| TN1 | 801-253-8625-50 |  |
| TN2 | <Null> |  |
| DATA\_TN | 801-254-1078-4 |  |
| CUS\_NAME | SIMMONS, GENE |  |
| ONT\_ADDRESS | 11066 S KESTREL RISE RD ONT 21 |  |
| NCON\_ADDRESS | 11066 S KESTREL RISE RD SOUTH JORDAN |  |
| CITY | SOUTH JORDAN |  |
| STATE | UT |  |

#### NCON\_FTTH\_GPON

Created once a day, every morning, by the *get\_gpon\_ncon\_info.ksh* script. At the beginning of 2013 were 4292 records.

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Sample Value** | **Description** |
| PULL\_DATE | 2013-01-21 07:15:06.0 |  |
| CKT\_NMBR | 5837879 |  |
| VERSION | 999 |  |
| PORT\_TYPE | ETHERNET PORT |  |
| LOCATION | RVTNUTCSOL2 |  |
| EQUIP\_ID | RVTNUTCSOL2.00030101.15 |  |
| NCON\_AID | 1-1-1-1-15-1 |  |
| VADER\_AID | ONT-1-1-1-1-15 |  |
| PORT\_ID | 1-3 |  |
| PACKET\_ID | 201 |  |
| PANS\_EFF\_END\_DATE | 2037-12-31 |  |
| CKT\_ID | 801-446-3957-4 |  |
| FORMAT | S |  |
| STATUS | W |  |
| LOC\_A | RVTNUTCSOL2 |  |
| EQUIP\_ID\_A | <Null> |  |
| EQUIP\_ID\_Z | <Null> |  |
| ACTIVATION\_STATUS | U |  |
| CUS\_NAME | KINISON, SAM |  |
| CUS\_ADDR\_A | 10867 S NAVARRO WAY SOUTH JORDAN |  |

#### NCON\_FTTH\_BPON

Details about this table need to be determined. At the beginning of 2013 there were only 132 records.

#### WEBSITE\_ACCESS

This table stores authentication and authorization records for Vader. The table is also shared with other applications(?). Refer to *Application Security* section below for more details.

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Sample Value** | **Description** |
| USERNAME | bgoober | Should match LDAP account name (which is used for Vader login). |
| PASSWORD | hello | Plain-text password |
| SECURITY\_LEVEL | “FIELD TECH” | TOC|VDOC|TIER III|TIER I|FIELD TECH|GUEST|ADMIN |
| authPermBits | 45296 | = 1011000011110000, see constants-security.php |
| FIRST\_NAME | Bob | First name |
| LAST\_NAME | Goober | Last name |
| TN | “303 123 4567” | Phone number |
| USER\_GROUP | “FIELD TECH” | Mostly matches SECURITY\_LEVEL (some exceptions) |
| E-MAIL | Bob.Goober@CenturyLink.com | Company email address |

The webpage interface for adding *Vdsltechsupp accounts* is: http://vdsltechsupp.uswc.uswest.com/admin.php

There are no webpage authorization restrictions for adding accounts in this table!

#### vader\_message\_log\_test

This table stores messages that are logged from the test instance. This implementation is new as of Vader 8.0.

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Sample Value** | **Description** |
| timestamp |  |  |
| domain |  |  |
| state |  |  |
| level |  |  |
| message |  |  |
| clientip |  |  |
| script |  |  |
| transid |  |  |

#### vader\_message\_log\_integration

This table stores messages that are logged from the integration instance. This implementation is new as of Vader 8.0.

#### vader\_message\_log\_production

This table stores messages that are logged from the production instance. This implementation is new as of Vader 8.0.

## Crontabs

The following Vader-related crontabs are run by the *scripts* user. Refer to the *Vader Tables* section of this document for details on table schemas.

### get\_gpon\_ncon\_info.ksh

*15 07 \* \* \* /opt/scripts/scripts/get\_gpon\_ncon\_info.ksh > /dev/null 2>&1*

ncon\_dumpfile=/export/home/prm/ncon.ftth.gpon.dat

Foreach OLT

Run ncon\_gpon\_dump.ksh >> $ncon\_dumpfile

Truncate and load to **NCON\_FTTH\_GPON**

Truncate and load to **GPON\_CUSTOMER\_NTWK** by joining **GPON\_ONT\_EQUIPMENT** and **NCON\_FTTH\_GPON**

Set city names in **GPON\_CUSTOMER\_NTWK** for:

South Jordan, St. George, Saratoga Springs, Lone Tree, Bellevue, Scottsdale, Littleton

NOTE: Additional city names are not implemented.

### ncon\_gpon\_dump.ksh

This script is called by the get\_gpon\_ncon\_info.ksh script.

Pull NCON ADSL Data & POTS Info (via java ThinClient pansco-ddp.qintra.com)

### get\_gpon\_equip.ksh

*10 5-21 \* \* \* /opt/scripts/scripts/get\_gpon\_equip.ksh > /dev/null 2>&1*

output\_dir=/export/home/prm/gpon/equip

Foreach OLT\_IP\_ADDRESS from **GPON\_OLT\_CHASSIS**

Get 11 ONT parameters using snmpwalk

Drop and recreate 11 temporary tables

Load the SNMP queried data to the tables

Join the temporary tables into one **GPON\_ONT\_EQUIPMENT**

Set table fields: ONT\_BIN\_IFINDEX and ONT\_AID

NOTE: ONT\_BIN\_IFINDEX and ONT\_AID are simply set by running SQL commands to parse other fields in the same row, and that “ONT-1-1-“ is hardcoded.

NOTE: The SNMP queries used by this script are only compatible with Adtran OLTs, yet this script does not filter out the Calix OLTs.

NOTE: This script takes more than an hour to complete, and therefore overlaps itself; which is probably ok since there is a sequential file usage order so there aren’t file overwrites.

### manage\_gpon\_batt\_alarms.ksh (should be *verify\_adtran\_OLT\_connectivity.ksh*)

*30 23 \* \* \* /opt/scripts/scripts/manage\_gpon\_batt\_alarms.ksh > /dev/null 2>&1*

This script was originally written (back in 2011) to monitor battery alarms for the Adtran GPON ONTs. It calls an expect script to get alarm information from every Adtran OLT, then loads database tables (which are no longer used). This GPON methodology is separate and distinct from the BPON battery alarm methodology (refer to the *FTTH BPON Battery Tracker Application* document, by Jeff Sheffel, June 5, 2013).

**NOTE**: This script doesn’t do what it used to do; see 130326 change note below.

This script sends notifications to customers that have an OLT with a battery alarm (eg. their battery is low) (which is currently disabled with script code commented out).

logfile=/export/home/vader/logs/battery/manage\_gpon\_battery.log

result\_file=/export/home/prm/gpon\_batt\_alarms.dat

vcsr\_msg\_dir=/export/home/vader/BusConnector/src/examples/bus

Foreach Production Adtran OLT\_IP\_ADDRESS from **GPON\_OLT\_CHASSIS**

Ping check OLT connectivity, send email on error

Run get\_gpon\_olt\_alarms.exp (telnet to send command RTRV-ALM-EQPT::ALL)

Verify OLT login was successful, send email on error

Load alarm data to **GPON\_ALARMS** table

Create alarm records in **GPON\_BATT\_TRACK** by joining **GPON\_ALARMS** and **GPON\_CUSTOMER\_NTWK**

Create contact records in **GPON\_BATT\_CONTACT**

Create the vcsrRequest text file to publish to Tibco Bus

Update alarm records in **GPON\_BATT\_TRACK** by adding customer contact information obtained from VCSR

Currently Disabled: Complicated logic to send customer notifications (via NFS to Tibco?)

On 130326, Lorenz changed to: only check Adtran OLTs, add OLT ping check, add OLT login check, remove email alert for zero alarm count. These modifications essentially changed the meaning of the script (to *verifying Vader OLT connectivity*); although the battery alarm code still runs without any resulting action taken. But, the script doesn’t check Calix OLTs!

### rotate\_apache\_logs.sh

*00 00 \* \* 0 /opt/scripts/scripts/rotate\_apache\_logs.sh*

This script rotates the apache logfiles on a weekly basis, by copying the existing logfiles to an archive directory, then truncating them.

### rotate\_vader\_logs.ksh

*00 00 \* \* 0 /opt/scripts/scripts/rotate\_vader\_logs.ksh > /dev/null 2>&1*

This script has been refactored to archive the Vader logfiles, in addition to rotating the logfile version for the week to a *.old* copy, which provides an interactive webpage capability to search the previous week’s (and the current week’s) logs by transaction ID. Refer to the *vdsl/xconn-lookup.php* webpage script for details.

### apachecheck

*00,05,10,15,20,25,30,35,40,45,50,55 \* \* \* \* /opt/scripts/scripts/apachecheck > /dev/null 2>&1*

## SNMP

Also refer to the *Vader SNMP Guide* for more details about the Vader code’s use of SNMP. The use of SNMP by the Vader code mostly applies to querying the Adtran devices.

Vader uses native PHP SNMP version 1 function calls.

Refer to the online php.net manual for details about SNMP functions.

An example snmpget function call is:

$oidSystemName = "1.3.6.1.2.1.1.5.0";

$snmpReply = @snmpget($oltip, $community\_string, $oidSystemName);

If there is no such OID on the device, or the community string is invalid, then a *PHP Notice* is generated. Note that the at-sign (@) suppresses a *PHP Notice* that will be generated if there is an error or timeout. The Vader lib-snmp.php library also has a *doSnmpget*() function to simplify SNMP get calls.

In general, Vader devices can use a community string of “public” for PHP calls to snmpget. This can easily be tested with the (Linux) command line snmpget command. If an unauthorized community string is specified in an snmpget call, the device will not respond, and the call will timeout.

A recent PHP function, that is defined in the *lib-snmp.php* library, should be used to get an SNMP community string:

**function** getSnmpCommunityString($deviceType = *NULL*)

Vader uses specific SNMP community strings, that must be configured into all OLTs. Refer to the *Vader OLT Database Management Procedure* document for community string requirements.

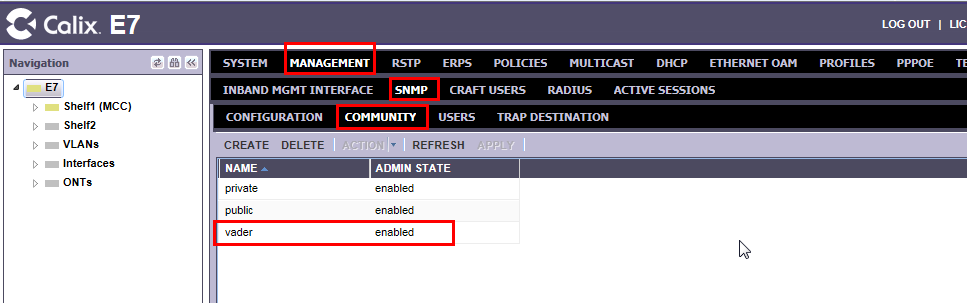
Adsl/margin-adjust.php (obsolete?):

* Community Read = “public”
* Community Write = “test2”

The corporate standard for SNMP community strings (for Calix E7s) (and E3s which are not GPON) are:

* Read-only - bln10ro
* Read-write - bln10rw

Here is a screenshot showing the SNMP community string settings on a Calix E7:



Refer to the Adtran PHP modules *gpon-request.php* and *gpon-web-provision.php* to see coded use of MIB variables.

### General MIB Definitions

MIB definitions, in general, can be cryptic and difficult to understand. MIB definition files are difficult to understand. An example of an easier format is as follows:

Name:  adGenEthernetInterfaceType

Type:  OBJECT-TYPE

OID:   1.3.6.1.4.1.664.5.70.2.1.1.5

Full path:

iso(1).org(3).dod(6).internet(1).private(4).enterprises(1).adtran(664).

adShared(5).adGenCndSystem(70).adGenEthernetDslamFlow(2).

adGenEthernetInterfaceTable(1).adGenEthernetInterfaceEntry(1).

adGenEthernetInterfaceType(5)

Module:  ADTRAN-ETHERNET-DSLAM-FLOW-MIB

Parent:        adGenEthernetInterfaceEntry

Prev sibling:  adGenEthernetInterfaceSourceAuthentication

Next sibling:  adGenEthernetInterfaceTypeSpecific

Numerical syntax:  Integer (32 bit)

Base syntax:       INTEGER

Composed syntax:   GenSystemInterfaceType

Status:            current

Max access:        read-only

Description:       Describes the interface type of ethernet interface

### MIBs of Interest

#### ONT Description Table

Some (Adtran) MIBs are in a table form, and possibly multi-dimensional.

The MIB variable that is the base of the table is:

SNMPv2-SMI::enterprises.664.6.10000.76.1.1.3.1.2 (= 1.3.6.1.4.1.664.6.10000.76.1.1.2.3.1.2)

The MIB variable of an example table entry is:

SNMPv2-SMI::enterprises.664.6.10000.76.1.1.3.1.2.1074353152

The bit structure of a table index can be seen, with an 11-bit slot number (Slot), the 3-bit port number (P), and the 6-bit ONT number (ONT#). Note that the following (multi-shelf) encoding scheme is one of two possible (Adtran) encodings; refer the the *Vader SNMP Guide* and the *convert-mib-index-to-ont-id.pl* script for more details.

**| Slot ||P||ONT#|**

**1000000000010010101010000000000 = 1074353152 Popular ONT - 14 of 15 devices have this ONT (ONT 21)**

**1000000000010010101100000000000 = 1074354176 Next ONT - 12 of 15 devices have this ONT (ONT 22)**

**1000000000010010110100000000000 = 1074358272 Popular ONT - 14 of 15 devices have this ONT (ONT 26)**

The following shows the actual string values returned by an SNMP command:

**| Slot ||P||ONT#|**

**1000000001000010000010000000000 = 1075905536 (ONT 01) STRING: "11066 S KESTREL RISE RD ONT 21"**

**1000000100000011000000000000000 = 1082228736 (ONT 32) STRING: "15648 Colorado Central Way Ont 2"**

## SOAP

Vader uses a SOAP interface to get network device information from the CTL management systems (for Calix CMS only, refer to the *Network Components* section for more details). SOAP is only used as a Calix device interface, and not Adtran devices (which is forthcoming by the Adtran AOE interface).

The SOAP calls provide:

* Device (OLT and ONT) configurations
* Alarm status

The SOAP interface used by Vader to make calls to a Calix CMS, was determined by (Rick Lorenz) using the CMS GUI (or an E7 GUI will work) in a debug mode, to view the equivalent SOAP calls. (Find prior document of following procedure or move to better document.)

1. In the E7 Navigation Tab, click the tiny binoculars button, which brings up a window to view debug output
   1. Optionally resize and move the debug output window for easier main menu navigation
   2. Set PROTOCOL to Finest, then click SET LEVEL
   3. Optionally click the CLEAR button to remove any debug output
2. Use the main menu to navigate to the E7 page of interest, taking note of the SOAP output in the debug window
   1. Optionally click the FREEZE button to freeze the debug window
3. Copy-n-paste the SOAP requests and responses
4. Upon completion, be sure to set the PROTOCOL and DEFAULT levels back to Info, and click SET LEVEL for both

Refer to the *Vader Debugging Guide* for SOAP debugging information.

Refer to the *Calix Management System (CMS) R11.3 XML Northbound Interface API Guide* (September 2012, or more recent version). See the chapter *XML Read (get-config and show-ont) Requests* (~ page 195) for details about making a SOAP request for an ONT configuration. But, most of the Vader SOAP calls were designed using the debug mode (just described).

For Adtran AOE SOAP specifications, refer to document *Automated Interfaces Command Reference Guide System Release: AOE6.2* (Document Number: 64150ANG62-35A March 2013).

The PHP SOAP library used by Vader is NuSoap.

### Vader SOAP Tester

A custom-built SOAP tester is locate at: http://<vader-root-url>/admin/test/test-soap-call-cms.php

Refer to the simple code header comments (in test-soap-call-cms.php) for usage details.

### PHP NuSoap Example

Example PHP NuSoap function call, to send() function:

$rtrv\_E7ONT\_OntPots\_port\_response =

$client->send(rtrv\_E7ONT\_OntPOTS($nodename, $ontid, $POTSPort, $user, $calix\_sessid), '', $connection\_timeout, $response\_timeout);

In the above NuSoap send() call, Vader assembles the SOAP request using utility functions, in this case the rtrv\_E7ONT\_OntPOTS() function. The send() call blocks and waits for the SOAP server to reply with a response. The following example was obtained by simply setting the URL debug request variable to “yes”.

Example SOAP Request (rtrv\_E7ONT\_OntPOTS VOICE-1) which is accessible in $client->request:

POST //cmsexc/ex/netconf HTTP/1.1

Host: 10.0.21.140:18080

User-Agent: NuSOAP/0.9.5 (1.123)

Connection: Keep-Alive

Content-Type: text/xml; charset=utf-8

SOAPAction: ""

Content-Length: 537

Cookie: JSESSIONID=2F74E941B446408F11DE25406A88A55F;

<soapenv:Envelope xmlns:soapenv="http://www.w3.org/2003/05/soap-envelope">

<soapenv:Body>

<rpc message-id="9581" nodename="NTWK-LTTNCOMLH0103CAB01A" timeout="35000" username="vader" sessionid="128">

<get-config>

<source>

<running/>

</source>

<filter type="subtree">

<top>

<object>

<type>OntPots</type>

<id>

<ont>22432</ont>

<ontslot>6</ontslot>

<ontpots>1</ontpots>

</id>

</object>

</top>

</filter>

</get-config>

</rpc>

</soapenv:Body>

</soapenv:Envelope>

Example SOAP Response (rtrv\_E7ONT\_OntPOTS VOICE-1) which is accessible in $client->response:

HTTP/1.1 200 OK

X-Powered-By: Servlet 2.4; Tomcat-5.0.28/JBoss-4.0.1sp1 (build: CVSTag=JBoss\_4\_0\_1\_SP1 date=200502160314)

Content-Type: text/xml;charset=ISO-8859-1

Content-Length: 512

Date: Tue, 29 Jan 2013 16:41:33 GMT

Server: Apache-Coyote/1.1

<soapenv:Envelope xmlns:soapenv="http://www.w3.org/2003/05/soap-envelope">

<soapenv:Body>

<rpc-reply message-id="9581" nodename="NTWK-LTTNCOMLH0103CAB01A">

<data><top><object>

<type>OntPots</type>

<id>

<ont>22432</ont>

<ontslot>6</ontslot>

<ontpots>1</ontpots>

</id>

<admin>enabled</admin>

<subscr-id/>

<descr/>

<impedance>600-ohm</impedance>

<signal-type>loop-start</signal-type>

<system-tx-loss>gr909</system-tx-loss>

<system-rx-loss>gr909</system-rx-loss>

</object></top></data>

</rpc-reply>

</soapenv:Body>

</soapenv:Envelope>

The NuSoap function returns a PHP associative array, which mirrors the SOAP response, and in the above example is contained the the variable $rtrv\_E7ONT\_OntPots\_port\_response:

Array

(

[!message-id] => 9581

[!nodename] => NTWK-LTTNCOMLH0103CAB01A

[data] => Array

(

[top] => Array

(

[object] => Array

(

[type] => OntPots

[id] => Array

(

[ont] => 22432

[ontslot] => 6

[ontpots] => 1

)

[admin] => enabled

[subscr-id] =>

[descr] =>

[impedance] => 600-ohm

[signal-type] => loop-start

[system-tx-loss] => gr909

[system-rx-loss] => gr909

)

)

)

)

Notice how the XML attributes and values are encoded in the PHP associative array; by using the exclamation marks. Also note that empty elements have an associative array entry that is an empty string (“”).

Another example using a SOAP response subset (ie. only showing a child element):

<children>

<child>

<type>McastProf</type>

<id>

<mcastprof attrname="blue">1</mcastprof>

</id>

<name>@DEFAULT&amp;3901</name>

</child>

<child>

<…>

where the equivalent PHP associative array is:

[children] => Array (

[child] => Array (

[0] => Array (

[type] => McastProf

[id] => Array (

[mcastprof] => Array (

[!attrname] => blue

[!] => 1

)

)

[name] => @DEFAULT&amp;3901

)

[1] => Array (

Next child continues here

)

)

The XML element values are accessed by the PHP “!” array key, and the XML attributes are accessed by the attribute name prefixed with a “!”.

The <child> elements are contained in an array element named *child*, and numerically indexed. But, there is an exception, if there is a single child element, then the array level is omitted. The following excerpt from the getChassisMulticastProfiles() method codes for each case.

**if**( **isset**($response['data']['top']['object']['children']['child']['type']) ) {

// Case when there is a single child; nusoap omits an array level, ie. not $response->[0]->['data'], but $response->['data']

$childrenArray[0] = $response['data']['top']['object']['children']['child'];

} **else** {

// Case when there is are multiple children; nusoap loads an array, ie. $response->[n]->['data']

$childrenArray = **isset**($response['data']['top']['object']['children']['child']) ? $response['data']['top']['object']['children']['child'] : **array**();

}

### PHP NuSoap Example Error Response

An example SOAP error response is as follows:

Example SOAP Reponse (rtrv\_E7ONT\_OntPOTSA VOICE-1):

HTTP/1.1 200 OK

X-Powered-By: Servlet 2.4; Tomcat-5.0.28/JBoss-4.0.1sp1 (build: CVSTag=JBoss\_4\_0\_1\_SP1 date=200502160314)

Content-Type: text/xml;charset=ISO-8859-1

Content-Length: 418

Date: Tue, 29 Jan 2013 16:41:33 GMT

Server: Apache-Coyote/1.1

<soapenv:Envelope xmlns:soapenv="http://www.w3.org/2003/05/soap-envelope">

<soapenv:Body>

<rpc-reply message-id="5004" nodename="NTWK-LTTNCOMLH0103CAB01A">

<rpc-error>

<error-type>application</error-type>

<error-tag>invalid-value</error-tag>

<error-severity>error</error-severity>

<error-message xml:lang="en">unknown attribute pkt-rate in attr-list</error-message>

</rpc-error></rpc-reply></soapenv:Body></soapenv:Envelope>

where the equivalent PHP associative array is:

Array

(

[rpc-error] => Array

(

[error-type] => application

[error-tag] => invalid-value

[error-severity] => error

[error-message] => Array

(

[!xml:lang] => en

[!] => unknown attribute pkt-rate in attr-list

)

)

[!message-id] => 5004

[!nodename] => NTWK-LTTNCOMLH0103CAB01A

)

Notice how the XML attributes and values are encoded in the PHP associative array; by using the exclamation marks.

## Production Scripts

### get\_gpon\_ont\_alarms.exp Expect Script

To obtain Adtran ONT alarm conditions, Vader (PHP) calls an Expect script (*adtran/get\_gpon\_ont\_alarms.exp*) that runs a (ksh) shell script on an EMS server. The expect script uses *ssh* to run the remote ~/get\_ont\_alarms.ksh script. The shell script simply greps for lines matching ONT search criteria, in the recent alarm logfiles. A sample Adtran alarm logfile entry is:

"04/03/2013 19:13:34","SCDOAZAVOL201CAB01B\_172.23.10.109, Shelf: 1, Slot: 1, Pon: 2, ONT: 7, …

ONT Serial No: ADTN1105009d, ONT Reg ID: ","172.23.10.109","SNMP Notification",3,"Minor","", …

"1.3.6.1.4.1.664.6.10000.76.1.1.5.1.0.12","adGenGponOntSetBatteryLowAlarm", …

"ADTRAN.SYSTEM.TA5000.TA5000\_23Inch","adGenGponOntSetBatteryLowAlarm", …

"1.3.6.1.4.1.664.6.10000.76.1.1.5.1.0.12","",474472406,1,"Open"

Refer to the get\_ont\_alarms.ksh script (immediately below) for more details. Refer to the *Vader Debugging Guide* (*Debugging Adtran ONT Alarms* section).

### get\_ont\_alarms.ksh

This script is called by the get\_gpon\_ont\_alarms.exp script and resides on each Adtran EMS. The script source is stored in the *vader/etc/script/adtran* directory, for reference and new EMS configurations.

The script greps by the OLT IP address and ONT ID of interest, in the six most recent alarm logs (.0 through .5), which can go back six months in time:

ssh rlorenz@151.119.110.67 # Adtran EMS server

rlorenz@sudnp91g.LAB$ ls –lt /opt/dorado/owareapps/redcell/Log

total 209552

-rw-r--r-- 1 ems staff 2260968 May 28 13:15 alarm.log

-rw-r--r-- 1 ems staff 10485834 Apr 9 22:11 alarm.log.1

-rw-r--r-- 1 ems staff 10485942 Feb 25 02:02 alarm.log.2

-rw-r--r-- 1 ems staff 10485991 Jan 3 14:36 alarm.log.3

-rw-r--r-- 1 ems staff 10485799 Nov 28 2012 alarm.log.4

-rw-r--r-- 1 ems staff 10485826 Oct 31 2012 alarm.log.5

-rw-r--r-- 1 ems staff 10485959 Oct 13 2012 alarm.log.6

Alarms get grep filtered out, returning only the last alarm in each alarm category (see *ftth/adtran/gpon-request.php*), and only if it is not a *Clear* or *Delete* entry.

### get\_battery\_alarms.ksh

Refer to the *FTTH-BPON-battery-tracker-application-details.docx* document for more battery tracker details.

## TL-1 Commands

The retrieval of Alcatel BPON OLT alarms uses a socket connection to the Amsden server, and sends TL-1 commands. The Amsden server is a Alcatel Alarm Management System (AMS?), that acts as a consolidator (that can communicate with each Alcatel device). Note that BPON is a legacy technology that is in end-of-life maintenance.

The NCON platform (by Telcordia) sends TL-1 commands to a (Calix) CMS for provisioning. The specifications for the Calix TL-1 commands can be found in the *Automated Interfaces Command Reference Guide System Release: AOE6.2* (Document Number: 64150ANG62-35A March 2013). The same document also contains AOE SOAP command specifications.

# Application Security

## HTTPS Web Interface

The Vader website is designed to be used by Vader administrators and the LPC team. Administrators can …

Since the Apache conf files may not secure the Vader directories against auto indexing and random file access, then .htaccess files are used.

* <vader-baseurl>/.htaccess – turns off auto indexing
* <vader-baseurl>/doc/.htaccess – turns on auto indexing
* <vader-baseurl>/include/config/.htaccess – turns off access to secure configuration files

## LDAP Authentication

Any CTL employee can login and use Vader GUI interface, by using their LDAP (AD domain) account and password.

## Vader Administrative User Accounts

Vader does have a user authorization layer, that is separate from the LDAP authentication. The administrative accounts are stored in the Vader (PRMDATA) database. The WEBSITE\_ACCESS table is used, which is shared among other VROC applications.

Only the following webpages enforce a legacy form of authorization, in which the logged in user’s SECURITY\_LEVEL (from the WEBSITE\_ACCESS table, see *Vader Tables* section above) must be one of "ADMIN"|"TIER III"|"VDOC":

* BPON GUI interface (ftth/web-request-query.php)
* BPON Request (ftth/web-request.php)
* BPON Bounce (ftth/web-bounce.php)

Because the Vader BPON component will eventually be EOL’d, this older form of authentication is simply left as is.

As of Vader 8.1, use of the *Authorization Permission Bits* (AuthPermBits) are used, that limits access to the *Vader OLT Database (Chassis Table) Edit* webpages. A *Vader Advanced User* permission (or higher) is required to access the webpages.

## Webservice Interface

The Vader webservice interface does not implement security!

# Software Development

## Documentation

Most of the Vader software development documentation is stored within the Git repository, under the *vader/doc* directory. As such, the documentation can be accessed via the website:

* http://<vader-top-url>/doc

The user documentation is at:

* http://<vader-top-url>/doc/list.php

This user documentation URL should be the URL that is advertised to general Vader users, as they will then not know about the ability to navigate to the programmer’s software development documentation. Furthermore, if the user is not logged into the Vader website, then accessing the list.php webpage will not even display the Vader website navigation links, so that Vader non-user can still browse the documenation (and not be aware of the Vader website).

## Vader Development

Refer to the *Vader Git Guide* document for more details on using Git.

Use Git …

A master repo resides on Testden3. Code is pushed to the master repo. Clones are pulled for test and integration instances.

### Release Milestones and Significant Code Enhancements

Refer to the Vader-change-log.txt for a definitive list of changes to the system. Some significant release milestones and enhancements include:

* v8.1 on June 3rd, 2013 – a wholesale refactoring of the Vader code

## Vader Code Structure

Adtran was the original vendor that Vader supported. The Vader code for Adtran support is not as modularized as Calix.

Calix support was added to the Vader application in the year 2012.

The basic Vader website code directory structure is as follows (incomplete):

htdocs/vdsl/vader/

admin/

doc/

etc/

script/

adtran/

ftth/

adtran/

calix/

## Coding Standards

The following basic coding conventions are used for the Vader code:

* Most newer (version 8.0+) variable names are in camelCase; while older variable names are not.
* The PHP contains comment strings with the “// TODO:” text pattern convention, to denote blocks of code that will need some form of an update. The Eclipse IDE recognizes the *TODO* pattern, and grep could also be utilized.

### Standard Modules

#### Configuration Settings

**require\_once** dirname(*\_\_FILE\_\_*).'/../../include/config.php';

This require is typically the first line in a Vader PHP module. After requiring config.php, then further module requiring is simplified, as the PHP include path is extended in config.php.

#### Database Interface

**require\_once** 'database.php';

Alternatively, there is an object-oriented database interface.

#### Debug Interface

**require\_once** 'lib-debug.php';

## Custom Utilites

Several utility programs exist to assist with programming, analyzing, diagnosing, and debugging. All of the utilities should reside in the Vader (Git) repository. Some of the utilities may be designed to run on a particular server; either the test or production server.

There are also utility scripts used to manipulate, build, configure, and install the Vader application.

### Device Analysis Utilities

##### get-ontids-for-calix-olts.pl

get-ontids-for-calix-olts.pl [--debug] [<ont-ip-address>]

Without the <ont-ip-address>, all Calix OLTs in the Vader OLT database will be scanned, which could take many minutes.

##### show-olt-config-adtran.sh

This script runs several SNMP queries to show interesting configuration parameters.

show-olt-config-adtran.sh <ont-ip-address>

##### get-olt-ip-addresses.pl

This script queries the Vader OLT database to obtain most of the fields for each OLT, including IP address.

get-olt-ip-address.pl [--addresses] [--calix] [--adtran]

--addresses – only display the OLT IP addresses; useful for (pipe) input to other scripts

--calix – only return Calix OLTs

--adtran – only return Atran OLTs

##### get-ontids-for-calix-olts.pl

##### show-olt-to-ont-pairs.sh

##### run-clei-code-analysis-all-vader-logs.sh

##### convert-mib-index-to-ont-id.pl

This script (mostly) applies to Adtran MIBs. This script has been enhanced to handle both the single-shelf and the multi-shelf indexing scheme; refer to the script for details.

When querying certain SNMP MIB variables, or reading Vader debug output, the MIB index is encoded; for example:

* Here is ifIndex of ONT: 1074931712
* Here is oid\_ont\_serial: 1.3.6.1.4.1.664.6.10000.76.1.1.2.3.1.1.1074931712

This script can be called with a single command line parameter, or read a list from STDIN.

convert-mib-index-to-ont-id.pl [<MIB-index>]

### SNMP Utilities

##### run-walk-mib.sh

##### run-snmpwalk-over-vader-olts.sh

##### run-snmpwalk-all-vader-devices.sh

This script is useful for running an snmpwalk on a particular MIB, against all of the Vader OLTs, or against the Vader OLTs of one of the vendors (ie. Adtran or Calix). The output is written to a single file that is created. Progress is reported to STDOUT.

USAGE: ./run-snmpwalk-all-vader-devices.sh <vendorOption> <mibSpec> ["<snmpOpts>"]

<vendorOption> = -adtran | -calix | -both

<mibSpec> = MIB specification

<snmpOpts> = eg. "-On"

An example run is:

testden3:sheffel: ./run-snmpwalk-all-vader-devices.sh -adtran 1.3.6.1.4.1.664.5.70.2.6.1.3

Running snmpwalk on 74 agents

Writing results to: snmpwalk-adtran-130826\_1139-1.3.6.1.4.1.664.5.70.2.6.1.3.out

10.231.148.62

10.231.148.106

…

##### run-snmpget-all-vader-devices.sh

##### run-snmpwalk-ifDescr-all-calix.sh

##### run-snmpwalk-SNMPv2-MIB-all-calix.sh

### Code Test Utilities

##### wget-vader-request.sh

##### wget-vader-test-request.sh

##### wget-vader-integration-request.sh

##### wget-vader-prod-request.sh

##### compare-request-test-to-prod.sh

##### compare-request-integration-to-prod.sh

##### compare-bounce-test-to-prod.sh

##### compare-bounce-integration-to-prod.sh

##### test-class-LogDatabase.php

##### test-getChassisOnts-method.php

Example usage:

php test-getChassisOnts-method.php 10.228.39.166 # An OLT with a dozen ONTs

##### test-getChassisMulticastProfiles-method.php

Example usage:

php test-getChassisMulticastProfiles-method.php 172.16.120.210 # Lab OLT with 6 or so profiles

php test-getChassisMulticastProfiles-method.php 10.228.39.166 # An OLT with one profile

##### test-doSnmpGet.php

### Code Management Utilities

##### format-git-log-output.sh

##### describe-table-GPON\_OLT\_CHASSIS.sh

#### Test Server

##### checkout-vader-to-htdocs.sh

##### clear-vader-logfiles.sh

##### configure-vader-test-website.sh

##### create-vader-release.sh

##### pull-vader-to-clone.sh

##### vader-checkout-configure.sh

##### list-vader-logfiles.sh

##### save-vader-logfiles.sh

##### copy-vader-docs-from-doc-server.sh

##### copy-vader-docs-to-integration-server.sh

#### Production Server

##### analyze-apache-log-vader.pl

##### generate-apache-log-stat-files.sh

##### create-apache-log-summary-webpage.pl

##### run-create-apache-log-summary-webpage.sh

##### test-week-nums-generation.sh

##### get-vader-gpon-olt-chassis-table.sh

##### run-vader-olts-snmp-check.pl

#### Production Server Vader User

##### configure-vader-integration-website.sh

##### backup-vdsltechsupp-home-vader.sh

##### show-vader-transaction-errors.sh

##### mysqldump-GPON\_OLT\_CHASSIS.sh

##### archive-table-GPON\_OLT\_CHASSIS.sh

##### create-vader-logfiles-tarfile.sh

##### list-vader-logfiles.sh

##### run-pkginfo-git-dependencies.sh

##### configure-vader-production-website.sh

### Logfile Analyzer Utilites

##### summarize-vader-logfile-sample-simple.pl

##### summarize-vader-logfile-xml-smart.pl

##### extract-vader-response-elements-ont-values.pl

##### summarize-vader-logfile-xml-twig.pl

## Unit Testing

There are a few unit tests that are coded. The vision was to use PHPunit. The unit tests reside in the *admin/test/unit* directory.

## Laboratory Device Testing

NDP or QwestRx testing … for proper testing, NCON provisioning of a test ONT …

Refer to the Appendix - Personnel Contacts – Applications for an NCON engineer contact.

|  |  |
| --- | --- |
| **NCON Build Request Parameter** | **Lab Device Specification** |
| OLT#: | LTTNCOMLH0 |
| TID: | LTTNCOMLH0113CAB01A |
| Node ID: | 1-1-2-34 |
| ONT type (product code): | 717GE |
| Slot ID: | (Not required for NCON build request?) |
| Port types: | 1xPOTS, 1xHSI, 1xVideo |

## Software Emulation of Hardware Devices

Software emulation of certain hardware devices, that are typically managed by Vader, was added in Vader version 8.0. The emulation implementation uses a simple approach by returning mostly hardcoded parameter responses to specific ONT AIDs (eg. ONT-1-1-1-1). The emulated devices are accessed in the same way that production devices are accessed, ie. by making webrequests to the request and bounce methods. But, the OLT IP address to which the requests are made, is the IP address of the Vader test server.

The emulation code is:

* ftth/test/gpon-request-test.php called from ftth/request.php
* ftth/test/gpon-bounce-test.php called from ftth/bounce.php

A better method for emulating devices would have been to have the software emulate the returned SOAP and SNMP calls (from an emulated EMS, CMS, and OLT). While such a method would be much more involved to implement, it would then actually test the Vader code modules (instead of simply returning hardcoded parameter responses).

The emulator is mostly unused.

## Benchmarks

There are no good programmatic benchmarks of the Vader code. Ideally, there would be some timing benchmarks of the webservice response times.

An estimate of webservice response times can be obtained by comparing timestamps in the Vader reponse logfiles, and probably use of the Apache logfiles.

gpon\_ftth.log

<TRANSNUM>TRANSNUMSTART=20130728001237188</TRANSNUM>

<DATETIME>07/28/2013 00:12:40</DATETIME>

Here, the transaction ID is created near the beginning of the Vader request processing, while the XML response timestamp (<DATETIME>) is captured near the end of the Vader request processing. The difference between the two timestamps measures all of the device (SNMP and SOAP) interfacing. Most of the Apache overhead would be excluded in the time difference.

Using the above timestamp differencing, typical webservice response times are:

* Adtran (SNMP): 4 seconds
* Calix (SOAP): 15 seconds

## Vader Deployment

Key concepts are in the Vader Application Framework section of this document. Each following subsection describes the deployment to the various Vader instances.

### Test Server Testden3

The Vader test instance resides on a separate development server (once known as Testden3). The server resides in Littleton’s (Washington) Mineral Laboratory 4. The IP address of the Vader test server is:

* 10.0.20.8

Refer to the *testden3:/export/home/admin/system-change-log.txt* file for log of changes to the (Solaris) operating system, that were done to support the Vader operating environment.

The ~vader user home directory contains …

The ~vader/bin directory …

The ~git/repo directory …

The Vader website is contained in the Apache /opt/csw/apache2/share/htdocs/vdsl/vader directory.

1. <testSever>:~vader/bin/ pull-vader-to-clone.sh (as vader user)
2. <testSever>:~vader/bin/vader-checkout-configure.sh (as root user)

A set of additional support files are required for the test instance website. The file set mostly consists of graphic files that reside above the <vaderInstance> directory, and are used in a shared context in the production instance. The files are copied to the required directories once, and won’t need reconfiguring (until a reinstall or graphic reference changes to the webpages).

* htdocs/vdsl/Connections/
* htdocs/vdsl/Graphics/
* htdocs/vdsl/SpryAssets/
* htdocs/vdsl/secure\_trailer.php
* htdocs/vdsl/tl1\_error.php
* htdocs/vdsl/trailer.php
* htdocs/vdsl/vader/ - this is the (test instance) Vader website

The supplemental file copies are stored at:

* <testServer>:~vader/src/supplemental

### Integration Server Vdsltechsupp

This section describes deployment of the Vader integration instance. The ~vader user home directory on the integration server …

The following script will only create a release tarfile from the (vader user’s) cloned copy of the Vader Git repository:

* <testServer>:~vader/bin/create-vader-release.sh (should be done as Vader user)

A release tarfile will be created in the ~vader/src/release directory.

Copy the new release to the integration server:

* <testServer>% scp vader-8.0.6.tar aa53378@vdsltechsupp:~vader/src/release/

Install the source to the integration webserver htdocs directory:

* <integrationServer>% sudo bash ~vader/bin/configure-vader-integration-website.sh 8.0.6

Review the script output for errors and information. Manually follow the steps that are displayed at the end of the script, ie.:

* <integrationServer>% sudo vi include/config.php
  + Set the $serverType variable, by commenting out TEST and uncommenting INTEGRATION
* <integrationServer>% sudo vi include/config/vader.ini
  + Again, set the proper vaderRelease and vaderInstance initialization variables

Copy the latest user (PDF) documentation …

* <testServer>:~vader/bin/copy-vader-docs-from-doc-server.sh
* <testServer>:~vader/bin/copy-vader-docs-to-integration-server.sh

Validate the new configuration:

* https://<integrationInstance>/vader-i/admin/show-vader-config.php

### Production Server Vdsltechsupp

When the latest Vader integration version has been tested (which may take many iterations on the days leading to release), prepare the new production htdocs directory (ahead of release time):

* sudo bash ~vader/bin/configure-vader-production-website.sh

Note that if a new temporary production directory is created for preparation, then the new prepared Vader instance can be (partially?) tested by editing the following entries in the config.php file:

* $urlVader = "https://vdsltechsupp.uswc.uswest.com/vader-p-8.1.0";
* $directoryAppHome = "$directoryAppClass/vader-p-8.1.0";

Obviously ensure the variables are reinstated before production cutover.

At the time of production cutover, rename the current Vader production instance directory, to archive it in case it needs to be reinstantiated

* mv vader/ vader-p-7.0/

Move the new version into production:

* mv vader-p-8.1.0/ vader/

Validate the new production version:

* Run tests
* Check logfiles

Copy the latest user documentation (PDF files) to the new production doc directory. This should really be done above, when preparing the new production directory.

Tag the Git repository with integration version tag (eg. v8.0.16) and new production tag (eg. v8.1.0).

## Vader Logfile Analysis

Beta code has been developed to analyze Vader logfiles. The code is written in Perl and resides in the Git repository. Refer to the *Vader Logfiles* section for more information.

## GUI User Interface

### Adding a US State to Vader Support

#### US Map HTML Image Map

The Vader administrator’s GUI website uses an HTML image map of the United States, that allows users to navigate to particular states that have OLT devices that are supported by Vader. When a new state needs to be added to Vader, the US map image and HTML webpage needs to be updated. An image of the individual US state also needs to be created for display of the state’s image when the user selects the state. Refer to the *gimp-vader-us-map-editing.txt* document for details on updating the image and image map.

The US map GUI page is: *vader/ftth/web/US-Map.php*

The XCF (Gimp) layered US map image is kept in the Git repository: *vader/etc/graphics/vader-us-map-layered-01.xcf*

The individual state images are stored in the *vader/ftth/web/graphics* directory.

#### Vader OLT Database Edit Webpages

The following database table edit webpages must also be updated, when adding a US state to Vader support:

1. ftth/web/chassis-table-add.php
2. ftth/web/chassis-table-edit.php

Note that space characters are removed from 2-word state names; which averts any problems with spaces in database queries or PHP processing. By adding the new state to the webpages, the database support should be automatic.

## Releasing Vader Upgrades

A distinction is made, between a major (8.1.0) or minor (8.3.0) Vader **release**, versus a patch **upgrade** (8.3.1). (This Vader terminology is debatable and subject to improvement.)

For production releases…

|  |  |  |
| --- | --- | --- |
| **Vader Client Application** | **Primary Sponsor** | **Other Sponsors** |
| QwestRx | Sue Leuschen | Soumyadeep Choudhuri, Arindam Dasgupta |
| PollDslam | Steve Wonchoba |  |
| NDP | Sue Leuschen | Arunakiran Nulu, Grant Olson, Kyla Frith, Shonna Reddick, Clark Fishback |
| LPC | Katie Murray | Stephanie Fischer, Lia Jeski |

### Integration Patch Upgrades

For integration releases … follow a process that is similar to the production process below.

cd <htdocs-integration-instance>

tar tf /export/home/vader/src/release/vader-8.1.3-patch.tar

for File in `tar tf /export/home/vader/src/release/vader-8.1.3-patch.tar`; do ls -l $File; done

# Backup files that will be updated

mkdir -p /export/home/vader/src/backup/release/8.1.6

for File in `tar tf /export/home/vader/src/release/vader-8.1.6-patch.tar`; do cp -p $File /export/home/vader/src/backup/release/8.1.6; done

sudo tar xvf /export/home/vader/src/release/vader-8.1.3-patch.tar

for File in `tar tf /export/home/vader/src/release/vader-8.1.3-patch.tar`; do ls -l $File; done

sudo vi include/config/vader.ini # Optionally configure and verify settings

sudo vi include/config.php # Optionally configure and verify settings

### Production Patch Upgrades

When a minor subversion (8.1.x), or a bug patch is required for the production instance, there is more of a manual process to perform.

Run the following commands on the development (and repository) server (Testden3):

cd /export/home/vader/src/git/clone/vader/

git checkout v8.1

Determine the list of files to include in the patch tarfile and create a text tar include file:

git log # Determine Git <commit> of prior release and most recent commit; verify prior release Git ID with Git tag

git diff --name-only <commit-prior-release> <commit-most-recent>

vi ~vader/src/release/etc/vader-tar-include-8.1.2.txt

Create the patch tarfile and upload to the production server:

tar cvf ~vader/src/release/vader-8.1.2-patch.tar -I ~vader/src/release/etc/vader-tar-include-8.1.2.txt

scp vader-8.1.2-patch.tar aa53378@vdsltechsupp:/export/home/vader/src/release

Run the following commands on the production server:

cd /opt/apache/htdocs/vdsl/vader # To the production htdocs directory

tar tf ~vader/src/release/vader-8.1.2-patch.tar # Verify the tarfile contents

Review the Unix file permissions of the production files that will be updated. Then, extract the files:

sudo tar xvf ~vader/src/release/vader-8.1.2-patch.tar # Extract the patch files over production files!

Set any required configurations that were altered:

sudo chgrp staff <file-list>

Update the new Vader version number:

sudo vi include/config/vader.ini # Set the new Vader production version number (eg. 8.1.2)

Optionally, the config.php file may need to be updated:

sudo vi include/config.php # Set $serverType = PRODUCTION

Verify the new version number on the Vader website, and run appropriate production tests.

# Appendix

## Resources

### CenturyLink

Planning and Engineering Guidelines

* http://peg

All Things GPON

* http://gpon

### PHP

Error Handling – A simple summary of best practices for handling errors in PHP.

* http://phpmaster.com/error-handling-in-php/

### Git

Pro Git Book

* http://git-scm.com/book

Git Reference Manual

* http://git-scm.com/doc

### Vendors

#### Adtran

An Adtran account is required for some website access permission.

Adtran Support Forum

* https://supportforums.adtran.com/
* [Total Access 5000 Bridging System Controller Module User Interface Guide - System Release: 7.1 (1187011G1Q)](https://supportforums.adtran.com/docs/DOC-5947)

**3 months ago** in [CenturyLink Documents (TA5K)](https://supportforums.adtran.com/community/total-access/total-access-5000-series/centurylink-documents)

**This reference is relevant because the part number (1187011G1Q) matched a SNMP MIB returned from the Adtran OLT in the Mineral Lab:**

snmpwalk -c public 172.16.120.243 1.3.6.1.4.1.664.5

SNMPv2-SMI::enterprises.664.5.13.2.4.1.2.254 = STRING: "1187011G1Q"

#### Calix

Summary of Calix GPON Technology

* http://www.calix.com/solutions/fiber\_access.html

Calix SOAP APIs are documented and accessible on the Calix website; an account is required for access, which can be obtained by ?.

* https://portal.calix.com/portal/site/resourcecenter/docsCMSManagement/

User Interface Reference Documentation

OSS Integration - Northbound Interfaces

CMS R11.3 XML Northbound Interface API Guide (09/2012)

### Technologies

#### Unicast and Multicast

Basic unicast versus multicast description

* http://www.broadcastbuyersguide.com/ipinbroadcast/2011/05/unicast-vs-multicast/

Cisco video series on the multicast technology

* Part 1 - http://www.youtube.com/watch?v=AvbSMBKBZgY

Slide presentations of multicast routing details

* http://www.slideshare.net/Metaswitch\_NTD/ip-multicast-explained
* http://www.slideshare.net/RockyS11/multicastsummaryppt

## Vader Source Lines of Code (SLOC) Count

### Vader 8.1.0

Of significance, this major release of Vader had 22 more files (which includes many file deletions), 49 more lines of code (!!, which includes many duplicate-removed/refactored lines), and 892 more comment lines.

testden3:sheffel: cat cloc-vader-8.1.0.out

/export/home/vader/src/git/clone

testden3:sheffel: cloc --exclude-dir=soap vader

141 text files.

141 unique files.

2502 files ignored. <--- NOTE: .git directory

http://cloc.sourceforge.net v 1.56 T=12.0 s (10.5 files/s, 2983.9 lines/s)

-------------------------------------------------------------------------------

Language files blank comment code

-------------------------------------------------------------------------------

PHP **104** 2713 **4072** **23919**

XSD 13 0 5 3294

Javascript 1 19 26 597

XML 2 0 0 332

Expect 2 79 129 267

CSS 2 23 23 138

Korn Shell 2 26 83 62

-------------------------------------------------------------------------------

SUM: 126 2860 4338 28609

-------------------------------------------------------------------------------

### Vader 7.0

$ cloc --exclude-dir=soap vader

Digest::MD5 not installed; will skip file uniqueness checks.

118 text files.

118 unique files.

59 files ignored.

http://cloc.sourceforge.net v 1.56 T=8.0 s (12.4 files/s, 4330.1 lines/s)

-------------------------------------------------------------------------------

Language files blank comment code

-------------------------------------------------------------------------------

PHP **82** 3366 3180 **23870**

XSD 13 0 5 3300

Javascript 2 21 26 601

Expect 1 44 59 134

CSS 1 3 0 32

-------------------------------------------------------------------------------

SUM: 99 3434 3270 27937

-------------------------------------------------------------------------------

## Telecommunications Acronyms at CenturyLink

|  |  |  |
| --- | --- | --- |
| AE | Active Ethernet | The term Active Ethernet refers to a point-to-point Ethernet topology between an ONT and an aggregation switch. The Active term is used to differentiate the technology from the FSAN Passive optical network technology in which a number of ONTs share bandwidth over a fiber network. The Active implies active components in the field, a point-to-point network topology, and an intelligent termination point. |
| AID | Address Identifier | Used to address specific network devices, like ONTs. (Telabs) |
| BPON | Broadband Passive Optical Network |  |
| BRAS | Broadband Remote Access Server | An IP network element that aggregates user sessions from the access network, provides policy management, and provides QoS. |
| CPE |  |  |
| CLLI | Common Language Location Identifier | A unique ID for an OLT (eg. TNLKCOADOL1). |
| CSG |  | 3rd-party customer billing standard application, ask James Thomas or Lisa Caton or Lela Howard. |
| C-RAS | CenturyLink Remote Access Solutions |  |
| CRV | Call Reference Value | Parameter to a (Calix C7) voice gateway. |
| DSL | Digital Subscriber Line |  |
| DSLAM | Digital Subscriber Line Access Multiplexer |  |
| FOA | First Office Application | The initial installation of new capability equipment, typically as a semi-production customer status, to prove out the technology. Also known as green-field (very new) or brown-field (not as new?). |
| FTTH | Fiber to the Home |  |
| FTTP | Fiber to the Premise | A fiber transmission scheme to deliver broadband access between the central office and customer premise. |
| GE | Gigabit Ethernet |  |
| GPON | Gigabit Passive Optical Network |  |
| HIS | High Speed Internet |  |
| ISYS | Interprise Systems | A database for everything (tickets, alarms, inventory, provisioning), from 1992 (when USWest). |
| LARG | Line Access Reliability Group |  |
| LEEP | Loop Electronics Engineering Package |  |
| LPC | Loop Provisioning Center | CTL service delivery group that processes orders for customer ONTs. |
| NBA | Network Backbone Activity |  |
| NBI | North Bound Interface | A SOAP (et. al.?) interface to a Calix CMS. |
| NCON | Network Configuration Manager | Remote terminal and customer device provisioning system (from Ericsson). |
| NLC POTS/ADSL |  |  |
| NMA | Network Monitoring and Analysis |  |
| OLT | Optical Line Terminal | Service provider endpoint of the passive optical network (PON). |
| ONT | Optical Network Terminator | Device at the customer’s premise that will convert the optical signal to electrical signal. |
| OSMINE | Operations Systems Modifications for the Integration of Network Elements |  |
| QC | Qwest Classic |  |
| QCC | Qwest CenturyLink Classic |  |
| RONTA | Remote ONT Administration | A (newer) method of managing ONTs. |
| SFP | Small Form-factor Pluggable | An optical transceiver module used in (Calix) OLT cards. |
| Telcordia |  | NCON original developer/vendor (was Bell Labs, now Ericsson). |
| TID | Terminal Identifier | A unique ID of an OLT (eg. TNLKCOADOL101CAB01A). |
| TIRKS | Trunk Inventory Record Keeping System |  |
| VDSL | Video Digital Subscriber Line |  |
| VDOC | Video and Data Operations Center |  |
| VLAN | Virtual Local Access Network | A logical grouping of LAN devices. |
| VoIP | Voice over Internet Protocol | A real-time protocol that allows voice delivery over IP networks (e.g. SIP). |
| VROC | Video Reliability Operations Center |  |
| xDSL |  | Any number of DSL flavors. In this case, xDSL primarily implies ADSL, ADSL2, ADSL2+, VDSL2, which are different physical layer standards for delivering broadband service across "last mile" networks. |

## Personnel Contacts

### Field Engineers

Mike.Haakonson@CenturyLink.com Sr Eng, Core Support 5 - St. Cloud, MN

Exchange Group Membership: Broadband Access Summit, Omaha Prism, Tech Supp Distr

w 320.255.8167, m 320.291.4064

Levi.Haro@CenturyLink.com Sr Eng, Core Support 5 - Denver, CO RETIRED

Exchange Group Membership: Broadband Access Summit, Omaha Prism, Tech Supp Distr

w 303.707.8053, m 720.581.0840

David.Gilmore@CenturyLink.com Facilities Specialist, Serv Delivery - Denver, CO

Exchange Group Membership: Denver LPC Team

w 303.778.4177

Eric.Nelson@CenturyLink.com Eng II, Core Support - Anoka, MN

Exchange Group Membership: Ants Users, CATT Users, Tech Supp Distr

w 763.506.9880, m 763.567.3499

### Equipment Lab

Craig.Sheffield@CenturyLink.com Sr Architect, Test and Integration - Littleton, CO

Manages the **Calix** test equipment in the Mineral Lab.

John.Jordan@CenturyLink.com Lead Architect, Test and Integration - Littleton, CO

Manages the **Adtran** test equipment in the Mineral Lab.

John.Ulanskas@CenturyLink.com Retired but on contract, Test and Integration - Littleton, CO

Calix, Adtran, and *Everything else expert*

Tony.Bieniek@CenturyLink.com

Administers (GPON) field equipment certifications and (Mineral) lab manager.

Joe.Bowers@calix.com 303.544.1992, M 303.886.1367 - Denver, CO

* **Calix** support representative. Joe is frequently on site in the Mineral Lab.

Dave.Pratt@calix.com 214.288.1577 - Texas

* **Calix** support representative. Dave also visits the Mineral Lab.

Richard.Fink@adtran.com 720.810.4096 - Denver, CO

* **Adtran** support representative (he has never responded to inquiries!)

Brent.Stembridge@adtran.com 256.963.6918 - Alabama

* **Adtran** support, very helpful in getting technical answers.

Rick.Mabry@adtran.com w 303.471.9181 m 303.587.8010 - Denver, CO

* Adtran support manager, very helpful in following through.

### Network Architects

Tom.Odle@CenturyLink.com Lead Architect, Access Strategy and Architecture - Littleton, CO

Warren.Novak@CenturyLink.com Principle Architect, Access Strategy and Architecture - Littleton, CO

### Applications

Grant.Olson@CenturyLink.com Sr. Lead Software Developer, OSS Network Assurance - Minneapolis, MN

#### QwestRx

Sue.Leuschen@CenturyLink.com Sr. Process Analyst, Operations Management - Omaha, NE

Arindam.Dasgupta@CenturyLink.com IT Systems Analyst (Lead?/Designer?), OSS Network Assurance - Dublin, OH

Choudhuri, Soumyadeep – IT Project Coordinator II - Minneapolis, MN

Iytha VijayKumar, Naresh – IT Project Coordinator I - Minneapolis, MN

Changalasetty, ANIL – Programmer (Module Lead) - Bangalore, India

Santhosh.V@CenturyLink.com (Krivvidi Sai Mehar, Santhosh) – Software Tester - Bangalore, India

#### Network Diagnostics Portal (NDP)

Arunakiran.Nulu@CenturyLink.com SW Dev Eng, BSS Wholesale Billing - Littleton, CO

* Aruna (pronounced Arune) leads a team of NDP software developers.

Suryanarayana. RajuPvk@CenturyLink.com SW Dev Eng, BSS Wholesale Billing - Chennai, India

Aarthi.Muthuraj@CenturyLink.com SW Dev Eng, BSS Wholesale Billing - Chennai, India

NDP – Outlook email alias to Chennai, India team

#### PollDSLAM

Wonchoba, Steven – Sole application engineer, Minneapolis?, MN

#### Loop Provisioning Center (LPC)

Katie.Murray@CenturyLink.com Loop Provisioning Center Manager, Service Delivery - St. Paul, MN

Lia.Jeske@CenturyLink.com Loop Provisioning Center Technician, Service Delivery - Denver, CO

 Stephanie.Fischer@CenturyLink.com Loop Provisioning Center Technician, Service Delivery - Denver, CO

#### Network Monitoring and Analysis (NMA)

Michael.Cavalli@CenturyLink.com NROC Supervisor, National Network Operations (NSM) - Minneapolis, MN

Lynda.Petersen@CenturyLink.com Sr. Engineer, National Network Operations (NSM) - Denver, CO

#### NCON

Timothy.Bartholomew@CenturyLink.com Lead IT Systems Analyst, OSS Network - Seattle, WA

* Performs NCON provisioning

### Project Management

Beth.Spellicy@CenturyLink.com Service Delivery (Project) Manager , SVS Delivery - WorkFromHome, CO

Kimberly.Roberts@CenturyLink.com IT Proj Mgr, OSS Dev PM Network Assurance - Littleton, CO

### Systems

Sue.Leuschen@CenturyLink.com Sr. Process Analyst, Operations Mgmt - Omaha, NE

Crystal.D.Weigel@CenturyLink.com Sr. Lead Process Analyst, Region Ops - Minneapolis, MN

Kyla.Frith@CenturyLink.com Lead IT Systems Engineer, Systems Engineering - Dublin, OH

Aldo.Sandoval@CenturyLink.com Lead Engineer, Network Lifecycle - Littleton, CO

* helped decipher GR-303/TR08 service/signaling mode requirements

## The Good (the Bad and the Ugly) Stories

### Pipe Speed Required Immediately

a lack of requirements and no knowledge of the (QwestRx) code implementation

halted the product release to the Mesa Ridge (Colorado) market

rumored to be escalated up the ranks to the corporate CEO (Glen Post)

occurred at a time when the sole Vader programmer (Sheffel) was going on a week-long vacation.

at the final Go-NoGo meeting, the repair team was not ready, stating that training and documentation was still to be performed.

Convert a standard video profile name to a profile name that has a standard GPON video "pipe rate" embedded in the name:

"video\_traffic\_GPON" becomes "video\_traffic\_2500000x2500000\_GPON"

This allows QwestRx (health check) to parse a pipe rate so a zero-value is not displayed to the user.