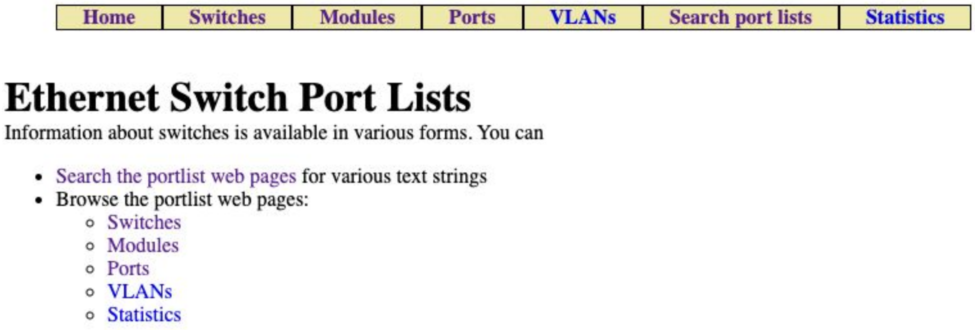
Muntaha Pasha
12/18/2019
Rotation 1 & 2 - NETS

NETS Rotation (NCAR SARP)

I want to start off this report by explaining the purpose of all this documentation. I'm writing this report to recount my experience here in the NETS team of CISL and what I learned, as well as what I am taking away from this rotation!

1. Introduction to NETS

The first thing I did in this rotation was to get up to speed with what NETS is, what it does, and how it works. Matt showed me the inner-workings of the UCAR network, giving me a tour of portlists, and a tour of my first ever data center! (Very exciting).

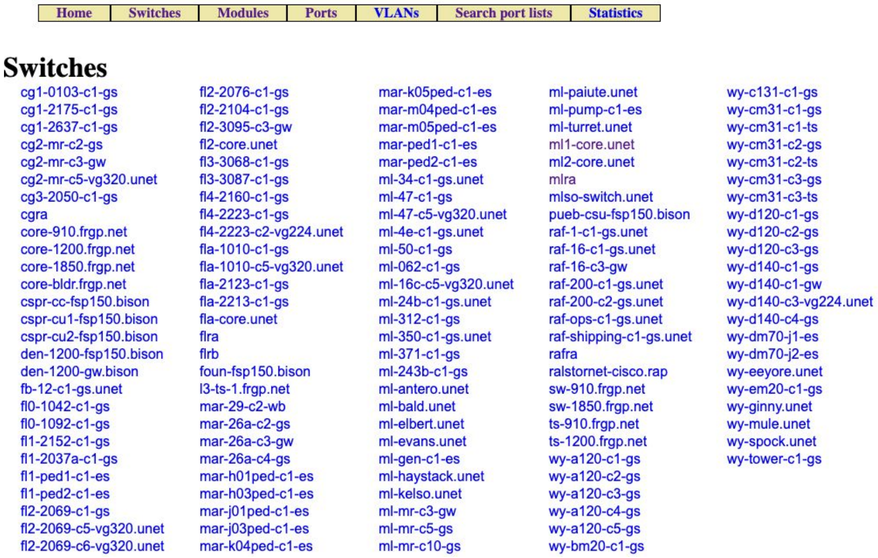


Ethernet Switch Port Lists

Information about switches is available in various forms. You can

- Search the portlist web pages for various text strings
- Browse the portlist web pages:
  - Switches
  - Modules
  - Ports
  - VLANs
  - Statistics

I learned about each of the different subsections to Port Lists, what it encompasses, and what the different tables display. For instance, I worked most closely with the switches information which is shown below.

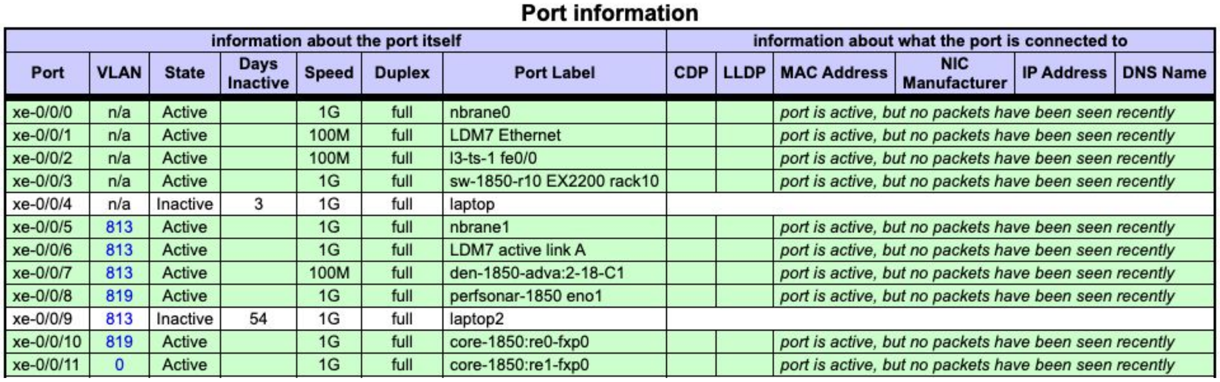


Switches

cg1-0103-c1-gs
cg1-2175-c1-gs
cg1-2637-c1-gs
cg2-mr-c2-gs
cg2-mr-c3-gw
cg2-mr-c5-vg320.unet
cg3-2050-c1-gs
cgra
core-910.frgp.net
core-1200.frgp.net
core-1850.frgp.net
core-bldr.frgp.net
cspr-cc-fsp150.bison
cspr-cu1-fsp150.bison
cspr-cu2-fsp150.bison
den-1200-fsp150.bison
den-1200-gw.bison
fb-12-c1-gs.unet
fl0-1042-c1-gs
fl0-1092-c1-gs
fl1-2152-c1-gs
fl1-2037a-c1-gs
fl1-ped1-c1-es
fl1-ped2-c1-es
fl2-2069-c1-gs
fl2-2069-c5-vg320.unet
fl2-2069-c6-vg320.unet
fl2-2076-c1-gs
fl2-2104-c1-gs
fl2-3095-c3-gw
fl2-core.unet
fl3-3068-c1-gs
fl3-3087-c1-gs
fl4-2160-c1-gs
fl4-2223-c1-gs
fl4-2223-c2-vg224.unet
fla-1010-c1-gs
fla-1010-c5-vg320.unet
fla-2123-c1-gs
fla-2213-c1-gs
fla-core.unet
flra
flrb
foun-fsp150.bison
l3-ts-1.frgp.net
mar-29-c2-wb
mar-26a-c2-gs
mar-26a-c3-gw
mar-26a-c4-gs
mar-h01ped-c1-es
mar-h03ped-c1-es
mar-j01ped-c1-es
mar-j03ped-c1-es
mar-k04ped-c1-es
mar-k05ped-c1-es
mar-m04ped-c1-es
mar-m05ped-c1-es
mar-ped1-c1-es
mar-ped2-c1-es
ml-34-c1-gs.unet
ml-47-c1-gs
ml-47-c5-vg320.unet
ml-4e-c1-gs.unet
ml-50-c1-gs
ml-062-c1-gs
ml-16c-c5-vg320.unet
ml-24b-c1-gs.unet
ml-312-c1-gs
ml-350-c1-gs.unet
ml-371-c1-gs
ml-243b-c1-gs
ml-antero.unet
ml-bald.unet
ml-elbert.unet
ml-evans.unet
ml-gen-c1-es
ml-haystack.unet
ml-kelso.unet
ml-mr-c3-gw
ml-mr-c5-gs
ml-mr-c10-gs
ml-paiute.unet
ml-pump-c1-es
ml-turret.unet
ml1-core.unet
ml2-core.unet
mlra
mlso-switch.unet
pueb-csu-fsp150.bison
raf-1-c1-gs.unet
raf-16-c1-gs.unet
raf-16-c3-gw
raf-200-c1-gs.unet
raf-200-c2-gs.unet
raf-ops-c1-gs.unet
raf-shipping-c1-gs.unet
rafra
ralstornet-cisco.rap
sw-910.frgp.net
sw-1850.frgp.net
ts-910.frgp.net
ts-1200.frgp.net
wy-a120-c1-gs
wy-a120-c2-gs
wy-a120-c3-gs
wy-a120-c4-gs
wy-a120-c5-gs
wy-bm20-c1-gs
wy-c131-c1-gs
wy-cm31-c1-gs
wy-cm31-c1-ts
wy-cm31-c2-gs
wy-cm31-c2-ts
wy-cm31-c3-gs
wy-cm31-c3-ts
wy-d120-c1-gs
wy-d120-c2-gs
wy-d120-c3-gs
wy-d140-c1-gs
wy-d140-c1-gw
wy-d140-c3-vg224.unet
wy-d140-c4-gs
wy-dm70-j1-es
wy-dm70-j2-es
wy-eeyore.unet
wy-em20-c1-gs
wy-ginny.unet
wy-mule.unet
wy-spock.unet
wy-tower-c1-gs

NETS runs a program called "SwitchMap", which runs twice a day and gets all kinds of information about many different switches. Each switch appears on this list, and when you click on the name of the switch, it gives you information that SwitchMap pulls out of that switch. Matt went over with me all the different variables we can pull out of the switch, including but not limited to… LLDP data, Port Names, MAC Addresses, and IP information.

Ports



Port information

information about the port itself							information about what the port is connected to					
Port	VLAN	State	Days Inactive	Speed	Duplex	Port Label	CDP	LLDP	MAC Address	NIC Manufacturer	IP Address	DNS Name
xe-0/0/0	n/a	Active		1G	full	nbrane0			*port is active, but no packets have been seen recently*			
xe-0/0/1	n/a	Active		100M	full	LDM7 Ethernet			*port is active, but no packets have been seen recently*			
xe-0/0/2	n/a	Active		100M	full	l3-ts-1 fe0/0			*port is active, but no packets have been seen recently*			
xe-0/0/3	n/a	Active		1G	full	sw-1850-r10 EX2200 rack10			*port is active, but no packets have been seen recently*			
xe-0/0/4	n/a	Inactive	3	1G	full	laptop						
xe-0/0/5	813	Active		1G	full	nbrane1			*port is active, but no packets have been seen recently*			
xe-0/0/6	813	Active		1G	full	LDM7 active link A			*port is active, but no packets have been seen recently*			
xe-0/0/7	813	Active		100M	full	den-1850-adva:2-18-C1			*port is active, but no packets have been seen recently*			
xe-0/0/8	819	Active		1G	full	perfsonar-1850 eno1			*port is active, but no packets have been seen recently*			
xe-0/0/9	813	Inactive	54	1G	full	laptop2						
xe-0/0/10	819	Active		1G	full	core-1850:re0-fxp0			*port is active, but no packets have been seen recently*			
xe-0/0/11	0	Active		1G	full	core-1850:re1-fxp0			*port is active, but no packets have been seen recently*			

It was a lot for me to understand, I only came in with a very brief knowledge about how networks worked and operated, and was not sure what kinds of variables were essential in understanding this. Some of this information began making a little more sense when I was actually shown some of the switches and ports in real time (in the data center). The gears in my mind started turning… slowly but surely!

2. Welcome to PortLists

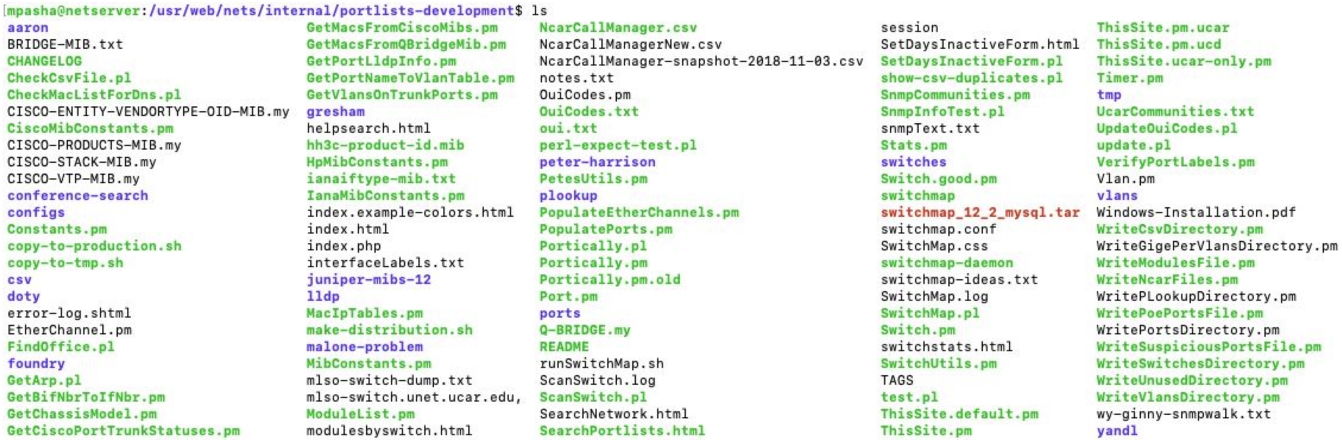
After about 2 days of introduction and tours, I finally began work on some actual tasks. Because there was a little uncertainty about where I'd fit into the team, it was a safe bet to start me off with the basics- pulling cables in the data center. It was definitely an experience, but I knew after about 10 minutes… this was not what I wanted to do! Although it solidified more of my knowledge on how the switches were connected to one another, it wasn't something I saw myself doing for much longer. Luckily, that was when I started working Pete, who introduced me to something I felt much more comfortable in…

CODE!

3. Phase 1 - PortLists (Fixing LLDP Information)

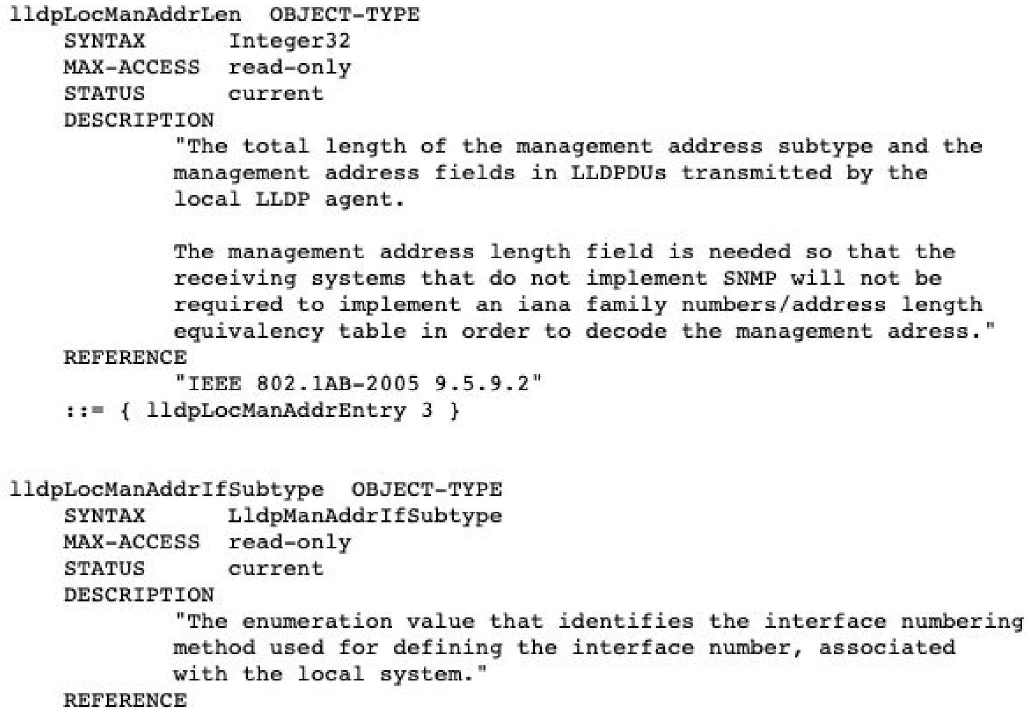
Portlists is written entirely in Perl, which isn't a language that college students learn often, but because of my experience in C++, Python, C, Scala, and various other programming languages, it was easy to catch on and start to piece together what was happening. He introduced me to the miraculous code that produces all of the hundreds of switch data tables that you can see and view on portlists. However, with this large amount of code, there's bound to be small bugs lurking about… and that's where my job really started. I was assigned to fix some missing LLDP information on some brand new Juniper Switches. LLDP was seemingly working fine on cisco's, but there was a disconnect between the two types of switches and how they relayed their data.

Muntaha Pasha
12/18/2019
Rotation 1 & 2 - NETS



```
mpasha@netserver:/usr/web/nets/internal/portlists-development$ ls
aaron                              GetMacsFromCiscoMibs.pm     NcarCallManager.csv                     session                  ThisSite.pm.ucar
BRIDGE-MIB.txt                     GetMacsFromQBridgeMib.pm    NcarCallManagerNew.csv                  SetDaysInactiveForm.html ThisSite.pm.ucd
CHANGELOG                          GetPortLldpInfo.pm          NcarCallManager-snapshot-2018-11-03.csv SetDaysInactiveForm.pl   ThisSite.ucar-only.pm
CheckCsvFile.pl                    GetPortNameToVlanTable.pm   notes.txt                               show-csv-duplicates.pl   Timer.pm
CheckMacListForDns.pl              GetVlansOnTrunkPorts.pm     OuiCodes.pm                             SnmpCommunities.pm       tmp
CISCO-ENTITY-VENDORTYPE-OID-MIB.my gresham                     OuiCodes.txt                            SnmpInfoTest.pl          UcarCommunities.txt
CiscoMibConstants.pm               helpsearch.html             oui.txt                                 snmpText.txt             UpdateOuiCodes.pl
CISCO-PRODUCTS-MIB.my              hh3c-product-id.mib         perl-expect-test.pl                     Stats.pm                 update.pl
CISCO-STACK-MIB.my                 HpMibConstants.pm           peter-harrison                          switches                 VerifyPortLabels.pm
CISCO-VTP-MIB.my                   ianaiftype-mib.txt          PetesUtils.pm                           Switch.good.pm           Vlan.pm
conference-search                  IanaMibConstants.pm         plookup                                 switchmap                vlans
configs                            index.example-colors.html   PopulateEtherChannels.pm                switchmap_12_2_mysql.tar Windows-Installation.pdf
Constants.pm                       index.html                  PopulatePorts.pm                        switchmap.conf           WriteCsvDirectory.pm
copy-to-production.sh              index.php                   Portically.pl                           SwitchMap.css            WriteGigePerVlansDirectory.pm
copy-to-tmp.sh                     interfaceLabels.txt         Portically.pm                           switchmap-daemon         WriteModulesFile.pm
csv                                juniper-mibs-12             Portically.pm.old                       switchmap-ideas.txt      WriteNcarFiles.pm
doty                               lldp                        Port.pm                                 SwitchMap.log            WritePLookupDirectory.pm
error-log.shtml                    MacIpTables.pm              ports                                   SwitchMap.pl             WritePoePortsFile.pm
EtherChannel.pm                    make-distribution.sh        Q-BRIDGE.my                             Switch.pm                WritePortsDirectory.pm
FindOffice.pl                      malone-problem              README                                  switchstats.html         WriteSuspiciousPortsFile.pm
foundry                            MibConstants.pm             runSwitchMap.sh                         SwitchUtils.pm           WriteSwitchesDirectory.pm
GetArp.pl                          mlso-switch-dump.txt        ScanSwitch.log                          TAGS                     WriteUnusedDirectory.pm
GetBifNbrToIfNbr.pm                mlso-switch.unet.ucar.edu,  ScanSwitch.pl                           test.pl                  WriteVlansDirectory.pm
GetChassisModel.pm                 ModuleList.pm               SearchNetwork.html                      ThisSite.default.pm      wy-ginny-snmpwalk.txt
GetCiscoPortTrunkStatuses.pm       modulesbyswitch.html        SearchPortlists.html                    ThisSite.pm              yandl
```

I was going to begin altering some of Pete's code to try and fix the issues going on with LLDP. However, before I could do these things, I obviously needed to be taught more about how Portlists works (we went over various code files), I ran switchmap myself on a few switches just to see how it operated, and I was truly immersed into the depths of networking and all that it encompassed from studying the code. I saw how to write the actual code that pulls variables from the switches, I was given tools like Pete's Perl Handbook (although I honestly googled most of my questions because it's 2019 and that's just what we do now), as well as the MIB files which actually contain data about different variables that you can extract and pull from switches!

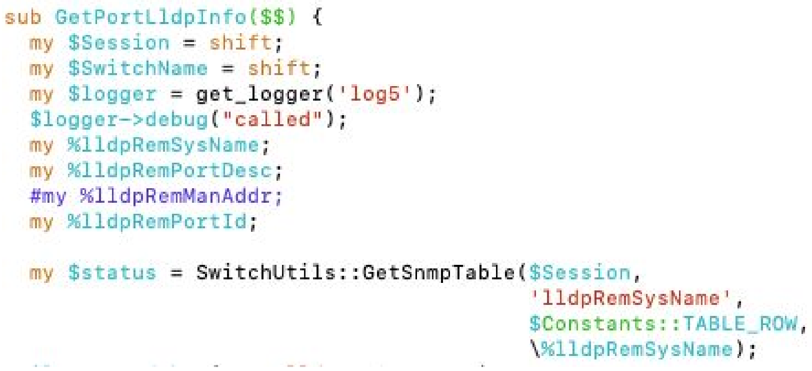


```
lldpLocManAddrLen  OBJECT-TYPE
    SYNTAX      Integer32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The total length of the management address subtype and the
            management address fields in LLDPDUs transmitted by the
            local LLDP agent.

            The management address length field is needed so that the
            receiving systems that do not implement SNMP will not be
            required to implement an iana family numbers/address length
            equivalency table in order to decode the management adress."
    REFERENCE
            "IEEE 802.1AB-2005 9.5.9.2"
    ::= { lldpLocManAddrEntry 3 }


lldpLocManAddrIfSubtype  OBJECT-TYPE
    SYNTAX      LldpManAddrIfSubtype
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The enumeration value that identifies the interface numbering
            method used for defining the interface number, associated
            with the local system."
    REFERENCE
```

Informing me about PortLists took about 2 weeks, and it was just enough time to actually get me on my feet and give me enough information about the system to actually start coding and fixing this

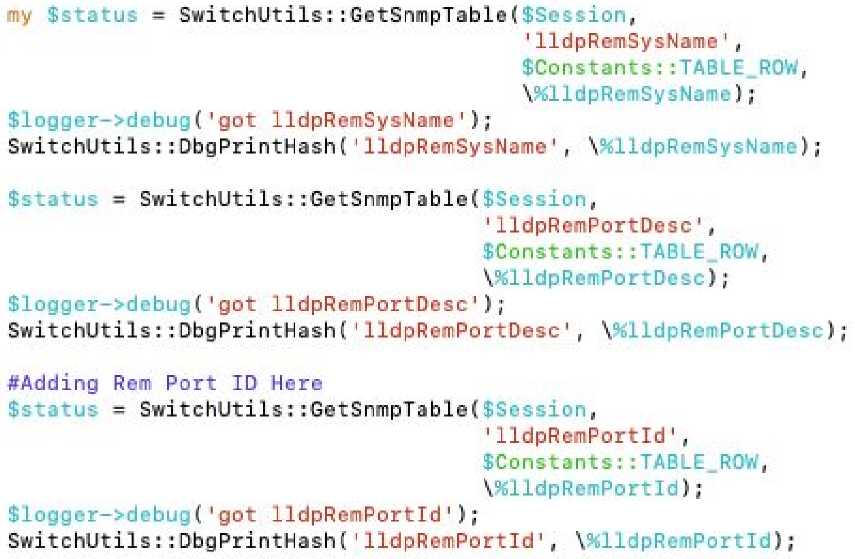
Muntaha Pasha
12/18/2019
Rotation 1 & 2 - NETS

LLDP bug! I was given access to the development files (because altering production code is a REALLY bad idea) and was told to go and solve this issue. Pete and I had looked over the main file which handles LLDP information quite a few times, so I understood what was going on, and what I needed to change.



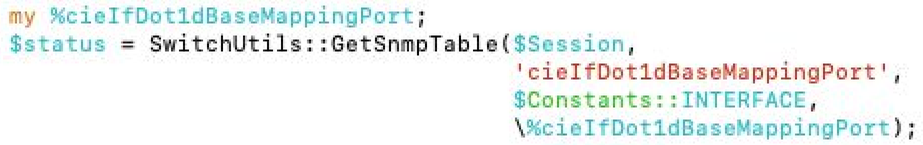
```perl
sub GetPortLldpInfo($$) {
  my $Session = shift;
  my $SwitchName = shift;
  my $logger = get_logger('log5');
  $logger->debug("called");
  my %lldpRemSysName;
  my %lldpRemPortDesc;
  #my %lldpRemManAddr;
  my %lldpRemPortId;

  my $status = SwitchUtils::GetSnmpTable($Session,
                                          'lldpRemSysName',
                                          $Constants::TABLE_ROW,
                                          \%lldpRemSysName);
```

The code above is a snapshot from the main code file that handles getting LLDP information. You give it a list of variables that you want to pull from the LLDP (like Remote System Name, Port Description, etc), and then you actually write the code that goes into each port of each switch, and gives you back all the variable information you want. At first, the code was configured to run on Cisco's alone, and so LLDP information was not showing up on Junipers. We discovered that what we needed to do was alter this code so that LLDP information can be pulled from Junipers AND Ciscos. So what I did was modify the code to pull information from Junipers and Cisco's separately.



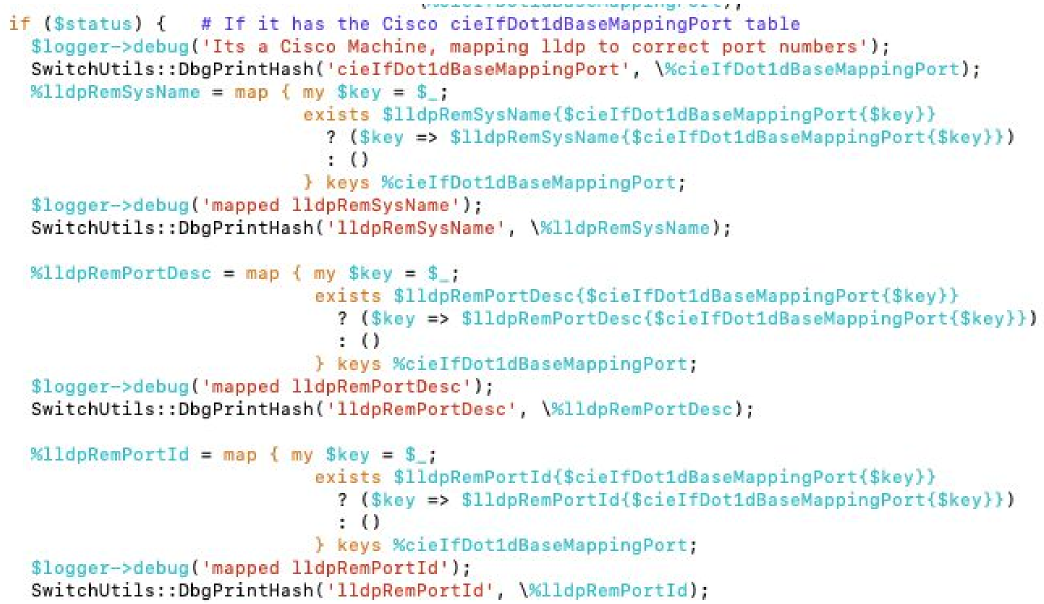
```perl
my $status = SwitchUtils::GetSnmpTable($Session,
                                        'lldpRemSysName',
                                        $Constants::TABLE_ROW,
                                        \%lldpRemSysName);
$logger->debug('got lldpRemSysName');
SwitchUtils::DbgPrintHash('lldpRemSysName', \%lldpRemSysName);

$status = SwitchUtils::GetSnmpTable($Session,
                                     'lldpRemPortDesc',
                                     $Constants::TABLE_ROW,
                                     \%lldpRemPortDesc);
$logger->debug('got lldpRemPortDesc');
SwitchUtils::DbgPrintHash('lldpRemPortDesc', \%lldpRemPortDesc);

#Adding Rem Port ID Here
$status = SwitchUtils::GetSnmpTable($Session,
                                     'lldpRemPortId',
                                     $Constants::TABLE_ROW,
                                     \%lldpRemPortId);
$logger->debug('got lldpRemPortId');
SwitchUtils::DbgPrintHash('lldpRemPortId', \%lldpRemPortId);
```

The code above looks complicated, but in essence it just pulls different LLDP variables from the ports. So in our case, it's pulling Remote System Name, Port Description, and Remote Port ID. This code is generalized, and works for Junipers. However, it does not operate on Cisco's, for which I needed to add a completely separate part. One thing that makes Cisco's stand apart from other devices is that it comes with something called the "cieIfDot1dBaseMappingPort". To me, this made the perfect variable for which to detect if a device was a cisco or not.



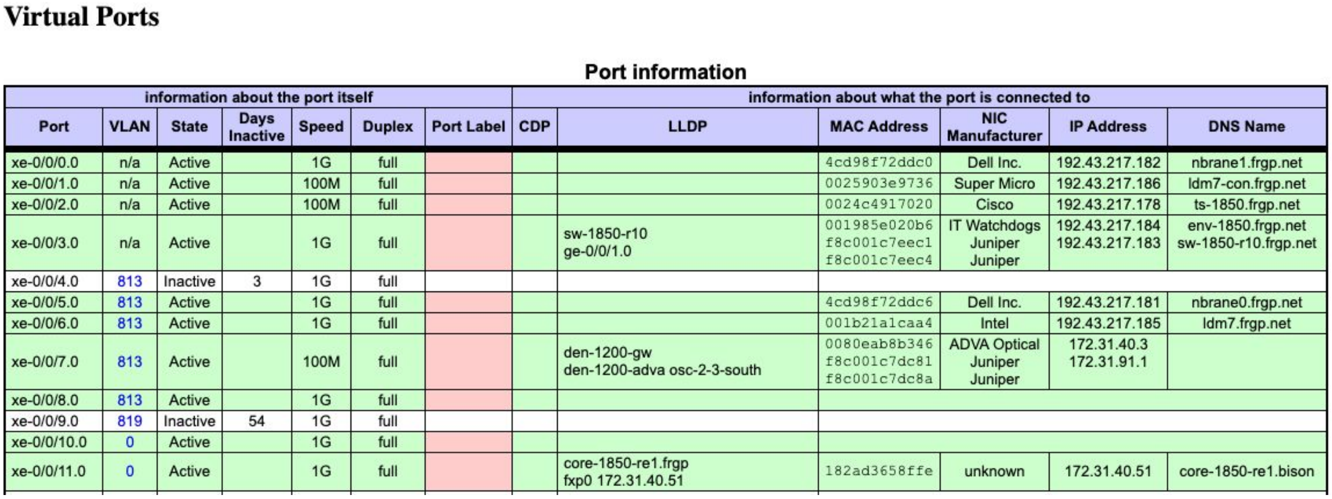
```
my %cieIfDot1dBaseMappingPort;
$status = SwitchUtils::GetSnmpTable($Session,
                                    'cieIfDot1dBaseMappingPort',
                                    $Constants::INTERFACE,
                                    \%cieIfDot1dBaseMappingPort);
```

The status variable that you see above held the cisco base mapping port, which I then used in a simple "if" statement to get LLDP information from Cisco's.



```
if ($status) {   # If it has the Cisco cieIfDot1dBaseMappingPort table
  $logger->debug('Its a Cisco Machine, mapping lldp to correct port numbers');
  SwitchUtils::DbgPrintHash('cieIfDot1dBaseMappingPort', \%cieIfDot1dBaseMappingPort);
  %lldpRemSysName = map { my $key = $_;
                          exists $lldpRemSysName{$cieIfDot1dBaseMappingPort{$key}}
                            ? ($key => $lldpRemSysName{$cieIfDot1dBaseMappingPort{$key}})
                            : ()
                          } keys %cieIfDot1dBaseMappingPort;
  $logger->debug('mapped lldpRemSysName');
  SwitchUtils::DbgPrintHash('lldpRemSysName', \%lldpRemSysName);

  %lldpRemPortDesc = map { my $key = $_;
                           exists $lldpRemPortDesc{$cieIfDot1dBaseMappingPort{$key}}
                             ? ($key => $lldpRemPortDesc{$cieIfDot1dBaseMappingPort{$key}})
                             : ()
                           } keys %cieIfDot1dBaseMappingPort;
  $logger->debug('mapped lldpRemPortDesc');
  SwitchUtils::DbgPrintHash('lldpRemPortDesc', \%lldpRemPortDesc);

  %lldpRemPortId = map { my $key = $_;
                           exists $lldpRemPortId{$cieIfDot1dBaseMappingPort{$key}}
                             ? ($key => $lldpRemPortId{$cieIfDot1dBaseMappingPort{$key}})
                             : ()
                           } keys %cieIfDot1dBaseMappingPort;
  $logger->debug('mapped lldpRemPortId');
  SwitchUtils::DbgPrintHash('lldpRemPortId', \%lldpRemPortId);
```

I checked to see if it was a Cisco, then inside of that statement, I pulled LLDP information from Cisco Machines. As you can see, there is an extra step here to pull LLDP information that was not present in the code for the Junipers. Here, you need to map every key to the cisco base mapping port in order to retrieve those LLDP variables. I did this for each variable, and then ran SwitchMap…



Virtual Ports

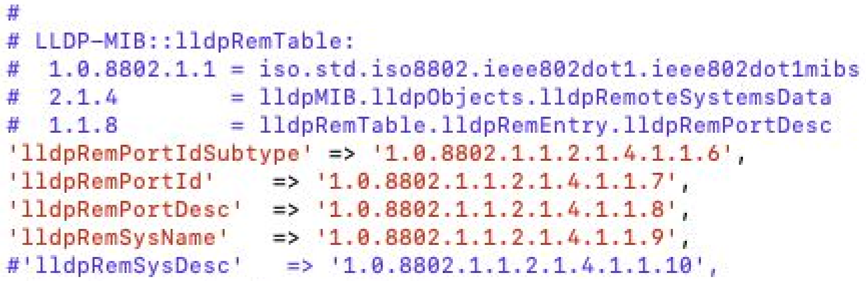
Port information

information about the port itself							information about what the port is connected to					
Port	VLAN	State	Days Inactive	Speed	Duplex	Port Label	CDP	LLDP	MAC Address	NIC Manufacturer	IP Address	DNS Name
xe-0/0/0.0	n/a	Active		1G	full				4cd98f72ddc0	Dell Inc.	192.43.217.182	nbrane1.frgp.net
xe-0/0/1.0	n/a	Active		100M	full				0025903e9736	Super Micro	192.43.217.186	ldm7-con.frgp.net
xe-0/0/2.0	n/a	Active		100M	full				0024c4917020	Cisco	192.43.217.178	ts-1850.frgp.net
xe-0/0/3.0	n/a	Active		1G	full			sw-1850-r10 ge-0/0/1.0	001985e020b6 f8c001c7eec1 f8c001c7eec4	IT Watchdogs Juniper Juniper	192.43.217.184 192.43.217.183	env-1850.frgp.net sw-1850-r10.frgp.net
xe-0/0/4.0	813	Inactive	3	1G	full							
xe-0/0/5.0	813	Active		1G	full				4cd98f72ddc6	Dell Inc.	192.43.217.181	nbrane0.frgp.net
xe-0/0/6.0	813	Active		1G	full				001b21a1caa4	Intel	192.43.217.185	ldm7.frgp.net
xe-0/0/7.0	813	Active		100M	full			den-1200-gw den-1200-adva osc-2-3-south	0080eab8b346 f8c001c7dc81 f8c001c7dc8a	ADVA Optical Juniper Juniper	172.31.40.3 172.31.91.1	
xe-0/0/8.0	813	Active		1G	full							
xe-0/0/9.0	819	Inactive	54	1G	full							
xe-0/0/10.0	0	Active		1G	full							
xe-0/0/11.0	0	Active		1G	full			core-1850-re1.frgp fxp0 172.31.40.51	182ad3658ffe	unknown	172.31.40.51	core-1850-re1.bison

Voila! LLDP information is present now!

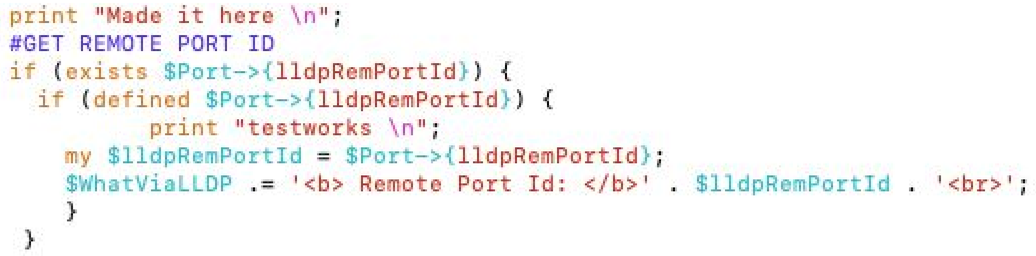
4. Phase 2 - PortLists (RemPortID)

Now that progress was being made on PortLists, I was tasked with getting another variable from the LLDP tables. Pete had discussed the importance of seeing the Remote Port ID (which was initially not in the LLDP code), so my next project was to get that information to appear in the LLDP column. The first thing I had to do was modify the constants file, and actually add the Remote Port ID to that file.



```
#
# LLDP-MIB::lldpRemTable:
#  1.0.8802.1.1 = iso.std.iso8802.ieee802dot1.ieee802dot1mibs
#  2.1.4        = lldpMIB.lldpObjects.lldpRemoteSystemsData
#  1.1.8        = lldpRemTable.lldpRemEntry.lldpRemPortDesc
'lldpRemPortIdSubtype' => '1.0.8802.1.1.2.1.4.1.1.6',
'lldpRemPortId'     => '1.0.8802.1.1.2.1.4.1.1.7',
'lldpRemPortDesc'   => '1.0.8802.1.1.2.1.4.1.1.8',
'lldpRemSysName'    => '1.0.8802.1.1.2.1.4.1.1.9',
#'lldpRemSysDesc'    => '1.0.8802.1.1.2.1.4.1.1.10',
```

Once I added the variable and its mapping to the Constants File, it was time to modify the other files like GetPortLLDPInfo, and SwitchUtils (Which puts the actual variables into the LLDP column). I modified each of those files in order to get RemotePortID's, (which you already saw above in the GetPortLLDPInfo code). Here is a modification of it in the SwitchUtils file.



```
print "Made it here \n";
#GET REMOTE PORT ID
if (exists $Port->{lldpRemPortId}) {
  if (defined $Port->{lldpRemPortId}) {
          print "testworks \n";
    my $lldpRemPortId = $Port->{lldpRemPortId};
    $WhatViaLLDP .= '<b> Remote Port Id: </b>' . $lldpRemPortId . '<br>';
    }
 }
```

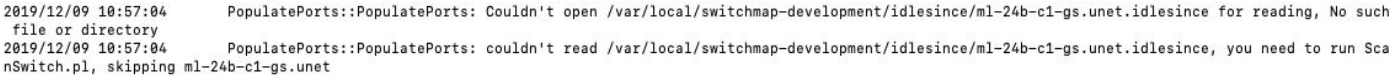
After I added adjustments to both of these files, I expected smooth sailing when I ran SwitchMap.pl on a switch to test and see if the remote port ID appeared in the LLDP column… but it didn't. So I went back, fixed a few bugs, and then ran it again.

Nothing.

I tried some google searches on LLDP variables, whiteboarded my code to no end, stepped through each line and argued the very logic of each statement with myself, and then ran it again.

Nothing.

This process went on for a week, as I scrambled to try and understand why it wasn't working. I rewrote the code so many times that eventually I ended up making it way worse than when I started. I encountered a real issue. Troubleshooting wasn't working, Debugging was taking forever, I even tried a SNMP walk on my code to see where it was failing, but still couldn't get too many clues about the exact location my code was failing. To resolve this, I decided to delete all the complicated things I had done, and re-approach the program with a clear mind. I reverted the files back to their original states, made enough changes to where I could confirm my logic was corrected, and then ran SwitchMap.pl again, this time keeping my eyes out for any errors. Surely, I got some.



```
2019/12/09 10:57:04       PopulatePorts::PopulatePorts: Couldn't open /var/local/switchmap-development/idlesince/ml-24b-c1-gs.unet.idlesince for reading, No such
 file or directory
2019/12/09 10:57:04       PopulatePorts::PopulatePorts: couldn't read /var/local/switchmap-development/idlesince/ml-24b-c1-gs.unet.idlesince, you need to run Sca
nSwitch.pl, skipping ml-24b-c1-gs.unet
```

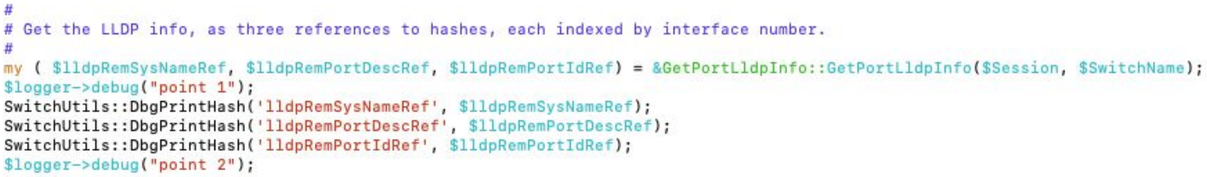
The error involved something in regards to “idle since”, which I was unfamiliar with. That is when I consulted Pete about my work. I walked him through Constants, GetPortLLDPInfo, and SwitchUtils. I explained every line of my code, and then… with one question… he solved 2 weeks worth of issues.

“Did you look at Populate Ports?”

….

Ofcourse.

2 weeks spent in a frenzy maniac of editing code, typing print statements everywhere, and whiteboarding IF-ELSE statements only to find out I had forgotten to modify another file that dealt with LLDP variables.



```
#
# Get the LLDP info, as three references to hashes, each indexed by interface number.
#
my ( $lldpRemSysNameRef, $lldpRemPortDescRef, $lldpRemPortIdRef) = &GetPortLldpInfo::GetPortLldpInfo($Session, $SwitchName);
$logger->debug("point 1");
SwitchUtils::DbgPrintHash('lldpRemSysNameRef', $lldpRemSysNameRef);
SwitchUtils::DbgPrintHash('lldpRemPortDescRef', $lldpRemPortDescRef);
SwitchUtils::DbgPrintHash('lldpRemPortIdRef', $lldpRemPortIdRef);
$logger->debug("point 2");
```

Well, that solved it.

From this, I learned the important lesson of “Check every code file where LLDP is used”.

The remote Port ID finally appeared in the LLDP Column, which was the goal.

5. Conclusion/Final Thoughts

This rotation was an amazing experience, and I am so thankful to all the people who made it as fun and educational as it was. Even though it seemed like a daunting task to take on at the beginning, putting my code to use in a real life setting for the first time was rewarding. It’s a great feeling knowing that all the programming I’ve done with PortLists on this rotation will be put into the production files so it can help the NETS team in the future. From coming into this rotation knowing so little about networking and all that it encompasses to be able to modify real-time code and fix bugs with such a big program was just mind blowing.

Thank you to the NETS team for making this possible!