TL1 Toolkit

Ronald van der Pol

rvdp@sara.nl

RoN spring meeting, May 22nd 2006, Utrecht

overview

- what is TL1?
- what is the TL1 toolkit?
- some implementation details
- example scripts

What is TL1?

- Transaction Language 1 developed by Bellcore (Telcordia)
- management language for telecom equipment
- industry standard sold by Telcordia Technologies
- messages are plain ASCII text
- machine-man and machine-machine interface
- widely used (in telecom world)

TL1 disadvantages

- horrible syntax
- TL1 is typically hidden by Network Management Systems
- NMS usually have GUI
- Works well for incident management
- Not enough for daily management tasks
  - scripting needed

TL1 session example

```
< ACT-USER:"Asd001A_OME1T":NAAM:42::;IP 42
<
  "Asd001A_OME1T" 06-05-21 09:17:11
M  42 COMPLD
;
< RTRV-OM-ETH:"Asd001A_OME1T":ETH-1-1-3:42::;IP 42
<
  "Asd001A_OME1T" 06-05-21 09:17:47
M  42 COMPLD
  "ETH-1-1-
       3::INFRAMES=0,INFRAMESERR=0,INOCTETS=0,INDFR=0,INFRAMESDISCDS=0,INPAUSEFR=0,INCFR=0,F
       RTOOSHORTS=0,FCSERR=0,FRTOOLONGS=0,FRAG=0,JAB=0,SYMBOLERR=0,OUTFRAMES=0,OUTFRA
       MESERR=0,OUTOCTETS=0,OUTFRAMESDISCDS=0,OUTPAUSEFR=0,OUTDFR=0,AUTONEGCYCLES=0,IN
       TERNALMACRXERR=0,INTERNALMACTXERR=0"
;
< CANC-USER:"Asd001A_OME1T":NAAM:42;IP 42
<
  "Asd001A_OME1T" 06-05-21 09:18:09
M  42 COMPLD
;
```

TL1 Toolkit

- Perl module to communicate with TL1 capable devices
- Hides TL1 syntax from the user
- Parses output of the device and returns Perl data structures to the user
- Perl because many ISPs use Perl for scripting
- Not an alternative for NMS, but an addition
- Authors: Andree Toonk and Ronald van der Pol
- Based on Perl module by Arien Vijn

  needed to know exact TL1 command

  no output parsing

Perl script example code

```
#! /usr/bin/perl -w
use tl1;
use strict;

my $ne = tl1->new(hostname => “Wg001A_OME01”,
        username => “NAAM”, password => “xxxxx”);
$ne->open();
@circuits = $ne->retrieve_circuits();
$ne->close();
```

Perl script example code

- Sends the following TL1 commands:

  ACT-USER::NAAM:42::;

  RTRV-EQPT::ALL:42;

  RTRV-CRS-COUNT:::42::;

  RTRV-CRS-STS3C::ALL:42:::DISPLAY=PROV,CKTID=ALL;

  CANC-USER:NAAM:42:;

Perl script example code

- Ouput to parse:

```
< RTRV-CRS-STS3C:"Wg001A_OME01":ALL:42:::DISPLAY=PROV,CKTID=ALL;IP 42
<

   "Wg001A_OME01" 06-04-28 04:48:07
M  42 COMPLD
 "STS3C-1-6-1-22,STS3C-1-1-4-1:2WAY:CKTID=\"Lls001a_Wg001a_GE1(DLO-WUR\":"
 "STS3C-1-6-1-25,STS3C-1-1-4-4:2WAY:CKTID=\"Lls001a_Wg001a_GE1(DLO-WUR\":"
 "STS3C-1-6-1-28,STS3C-1-1-4-7:2WAY:CKTID=\"Lls001a_Wg001a_GE1(DLO-WUR\":"
 "STS3C-1-9-1-190,STS3C-1-1-2-1:2WAYPR:SWMATE=STS3C-1-6-1-190,CKTID=\"Lw001A_Wg001A_GE1(Wur-Hal) \":"
 "STS3C-1-9-1-1,STS3C-1-1-1-1:2WAYPR:SWMATE=STS3C-1-6-1-1,CKTID=\"Gv001A-Wg001A_GE1(Lei-WUR) \":"
"STS3C-1-9-1-31,STS3C-1-2-4-10:2WAY:CKTID=\"Lls001a_Wg001a_GE2(DLO-WUR)\":"
>

   "Wg001A_OME01" 06-04-28 04:48:07
M  42 COMPLD
   "STS3C-1-9-1-34,STS3C-1-2-4-13:2WAY:CKTID=\"Lls001a_Wg001a_GE2(DLO-WUR)\":"
```

Inside TL1 Toolkit

- Implemented as Perl module

```
# Constructor method
sub new {
        my $invocant = shift;
        my $class    = ref($invocant) || $invocant;
        my $self     = {
                username => 'NAAM',  password => 'xxxx',  type => 'ome6500',
        };
        bless( $self, $class );
        $self->{'resp_queue'} = Thread::Queue->new();
        $self->{'auto_queue'} = Thread::Queue->new();
        $self->{'acmd_queue'} = Thread::Queue->new();
        return $self;
}
```

open()

```
sub open {
        my ($self) = @_;
        $self->{'socket'} = new IO::Socket::INET->new(
                Proto    => 'tcp',  PeerAddr => $self->{'hostname'}, PeerPort => $self->{'peerport'}
         ) or $con = 0;
        # Handle alarms
        if ( defined( $self->{'auto_code'} ) ) {
                $self->{'auto_thread'} = async { $self->auto_thread(); }
        }
        # Start receiver
        $self->{'receiver_thread'} = async { $self->receiver_thread(); };
        # Async commands
        $self->{'async_cmd_thread'} = async {  $self->async_cmd_thread();  };
        # login
        my $result = $self->login();
}
```

receiver_thread()

```
sub receiver_thread {
        while (<$socket>) {
                # a line with a ; only means the end of a response
                /^;/ && ( $state =~ /auto|response/ ) && do {
                        my @resp : shared = @response;
                        $self->{'resp_queue'}->enqueue( \@resp )
                }
                # start of an autonomonous message (alarm)
                /^(\*C|\*\*|\* |A ) (\d{1,10}) (.*)/ && do {
                        $state = 'auto';
                }
                # start of a response
                /^M  (.*) (COMPLD|PRTL|DENY)/ && do {
                        $state = 'response';
                }
        }
}
```

part crossconnect script

```
my $ne = tl1->new();
if ($ne->open() == 0) { next;}
@out =  $ne->retrieve_circuits();
$ne->close();           # Close Connection
my $row;
for $row (@out) {
        # $row is an array with all circuit properties
        if (check_circuit($host, $row)) {
           print "updating circuit $host\n";
           update_circuit($host, $row);
        } else {
                print "inserting circuit $host\n";
                insert_circuit($host, $row);
        }
}
```

retrieve_circuits()

- returns an array of arrays, each row has:

id

bandwidth

beginslot

beginsubslot

beginport

beginfromsts

begintosts

endslot

endsubslot

endport

endfromsts

endtosts

swmateslot

swmatesubslot

swmateport

swmatefromsts

swmatetosts

retr_swversion()

```
sub retr_swversion {
        my ( $self  ) = @_;
        my $swversion;
        my $socket = $self->{'socket'};
        print $socket "RTRV-SW-VER:::3;\n";
        my $resp = $self->{'resp_queue'}->dequeue();
        my $n = 0;
        while (defined($resp->[$n])) {
                # SHELF-1:REL0200Z.HQ
                if ($resp->[$n] =~ /"SHELF-1:(\w+).*"/) {
                        $swversion = $1;
                }
                $n++;
        }
        return $swversion;
}
```

Examples

All local demo data, not real network info

- SURFnet6 crossconnect info
- Netherlight alarm info
- OME6500 configuration info
- syslog
- MRTG

Thank you

(rvdp@sara.nl)