TULIP Central Reflector

Purpose

The main purpose of TULIP Central Reflector is to proxy the TULIP queries to landmarks in PlanetLab's Scriptroute service, and the perfSONAR and PingER reverse ping servers. The PlanetLab Scriptroute service provides a cookie which works for a single IP address only. So in this way all the requests will be issued from the Central reflector and the responses will be sent back to the TULIP JNLP Client.

Implementation

The TULIP Central reflector will be a CGI Script (reflector.cgi) deployed at SLAC. The TULIP client will issue a single request and the Reflector will go ahead and probe all the landmarks in that region*[1] and return the results to the TULIP Client. Probing the target site from more vantage points may give us a better estimate of its location.

Requirements

- Should fetch sites.txt or have a local copy of sites.txt, what changes should be made to sites.txt ?
    - A new parameter should be added to sites.txt to include teir0 or tier1. Also the region of tier1 sites needs to be specified in sites.txt
- A separate thread should be used for each landmark and Semaphores should be used for locking, so that data from different threads should not inter-mix.
- There should be a limit on the number of threads that can be launched at a time (say 10).
- Should there be extra logging on the reflector or can we rely on the standard web logs which will log each query including time stamp, the client name. What else it logs depends on whether the request is Get or a Post.
- Where are the results parsed, could be in the reflector or in the Java client. In the client distributes the parsing load, reduces the load on the reflector, simplifies the CGI script.
- What should happen if a landmark responds with a bad data. (Should it process the error or send the raw data back?). Since there will be some anomalies I suspect the reflector will need to return the full response and anyhow needs to inform the user, so I suspect initially the client will process the response and spot errors etc.) Also if the client parses the result it will probably be easily able to spot problems.
- Special consideration for security as the script ultimately has to be deployed at SLAC (Perl taint option, warning option, special open method etc)
- Need to agree on a common format for the exchange of data.
- Needs a blacklisting mechanism for malicious hosts.

After discussing with Yee and Booker it was seen that forks may be too complicated. The version of Perl at SLAC did not support threading. Also the security people will not allow forks running inside a CGI-script. So I had to come up with an alternative. The solution to this problem was to use Asynchronous IO. A bunch of requests could be send to the landmarks without waiting for the response. The LWP::Parallel library provides all this functionality. It supports asynchronous IO. Currently it is not installed so I am using a local version in my home directory. Ultimately this module has to be installed on the production server.

I have implemented most of the functionality. The script is running fine. I will have to taken measures to make the script more secure so that it could not be used as a platform to launch DDOS attacks, by limiting the number of concurrent process of reflector.cgi to 10. Also the script produces customized messages such as (request time out or Connection failed so that the TULIP client can differentiate between the various kind of error conditions). Also there is a crude **blacklisting** mechanism so that particular IP addresses can be blocked.

Implementation

There are two scripts: reflector.cgi and EventHandler.pm. Both have -T (Tainting), warning (-w), use strict, use the 3 parameter version of open, all opens and closes have a die or its equivalent. EventHandler.pm is called by reflector.cgi. The CGI scripts are deployed in the path /afs/slac.stanford.edu/g/www/cgi-wrap-bin/net/shahryar/smokeping/.

Invocation

The reflector script is called by a URL of the form:

```
http://www-wanmon.slac.stanford.edu/cgi-wrap/reflector.cgi?region=North%20America&target=134.79.104.80
&tier=all&type=PlanetLab&ability=1
```

Leaving out the region will assume all regions, leaving out the tier will assume all tiers, leaving out the type will assume both PingER and perfSONAR type landmarks. If the region is included then only landmarks in that region will be used. If the tier is specified then only that tier's landmarks will be used. If the type is specified then only that type landmark(s) (e.g. type=PingER,perfSONAR will use only landmark types PingER and perfSONAR) will be used. Any or all of the tier, region and type region may be specified as "all".

The ability parameter enables the user to specify whether to use enabled landmarks (ability=1, default) or disabled landmarks (by specifying ability=0). The enabled landmarks are found in;

```
http://www.slac.stanford.edu/comp/net/wan-mon/tulip/sites.xml
```

The disabled landmarks are found in:

```
http://www.slac.stanford.edu/comp/net/wan-mon/tulip/sites-disabled.xml
```

The sites xml files are created by a trscrontanb running in pinger@pinger.slac.stanford.edu. The script that writes sites.xml is: create_sites-xml.pl. There is also a debug parameter. Use debug=1 or greater if you want debugging output.

The function parameter, enables one to specify to ping (function=ping, default), to print out usage information (function=help), to print out the log file(function=log), to print out an HTML table of the landmarks (function=landmarks) from the TULIP database, and to analyze the log file (function=analyze). The function analyze has a sub option ability=0 to analyze the logs for disabled landmarks and ability =1 (default) to analyze log entries excluding those for disabled landmarks. The function landmarks has an additional parameter out (out=html (default) or out=csv), which can toggle output between html and csv. Using this link, one can easily have a look at landmarks present in the database, along with their most important properties, like tracerouteURL, pingURL, longitude, latitude, city, country and hostname.

The script uses asynchronous I/O to talk simultaneously with up to 20 landmarks. Up to 5 copies of reflector.cgi can be running simultaneously.

For the Planetlab landmarks an interpretive script (also see here for the original is supplied with the target ($target) and the number of pings ($ping) to make. For the SLAC ping servers and the Looking glass sites the landmark is accessed by a URL provided in the sites.txt file (see below) in the PingSites token (e.g. http://www.slac.stanford.edu/cgi-wrap/nph-traceroute.pl?choice=yes&function=trace&target=$target).

New XML landmarks file

TULIP has been refactored to use an XML file for its site/landmark information. This is generated from a MySQL database in the 'tulip' schema on pinger. The file can be generated with a perl script in the subversion repository, pinger/trunk/bin/create_sites-xml.pl and its associated template file, pinger/trunk /bin/sites-xml.tt2. Add/modify nodes in the sites list by connecting to the database.

Data can be converted from the previous sites.txt format using pinger/trunk/bin/insert_sites-xml.pl. This creates SQL insert statements which can then be run using the mysql command line.

Old landmarks file

The list of SLAC and Looking Glass landmarks is read from /afs/slac/www/comp/net/wan-mon/tulip/sites.txt. the format is space separated tokens:

```
SNo Site_name PingSite TraceSite Lat Long Reference_Site Alpha, for example (where \ means line broken for
viewing)
1  SLAC,Stanford_US http://www.slac.stanford.edu/cgi-wrap/traceroute.pl?function=ping&target=
\ http://www.slac.stanford.edu/cgi-wrap/nph-traceroute.pl?choice=yes&function=trace&target=
\  39.32  -122.04 www.slac.stanford.edu   73
```

The list of PlanetLab landmarks is read from: /afs/slac.stanford.edu/www/comp/net/wan-mon/tulip/TULIP/newsites.txt. It appears as:

```
Piscataway_UnitedStates_PL 128.6.192.158 40.5516 -74.4637 orbpl1.rutgers.edu northamerica
Aachen_Germany_PL 137.226.138.154 50.7708 6.1053 freedom.informatik.RWTH-Aachen.DE europe
Winnipeg_Canada_PL 198.163.152.230 49.8833 -97.1668 planetlab2.win.trlabs.ca northamerica 0
```

Anything following a # sign is ignored (it is a comment).

The tokens are space delimited, they are:

```
Location: in the form City_Country_Type (the only Type currently is PL=PlanetLab)
IP Address
Latitude
Longitude
IP Name
Region (possible regions with PlanetLab hosts are northamerica, eastasia, europe)
Tier: currently may be 0 or 1, if not provided tier 1 is implied
```

Conventions

We use the following conventions:

- Country names are defined by the Mapland database since this is used to produce our maps and we cannot modify it. Usually (but not always) it is in agreement with UN standards.
- The country names (and regions) in the PingER database can be found here.
- Regions are defined as given in the PingER NodeDetails Oracle database.
- A region to country mapping can be found here. This mapping is created using the script createCountryRegion.pl which is placed in /afs/slac.stanford.edu/www/comp/net/wan-mon/tulip/TULIP/. This script uses the NODE_DETAILS hash to determine the regions of various countries.

Deployment of Landmarks

There are about 60 SLAC/Looking Glass landmarks, and about 156 PlanetLabs landmarks. We filter the landmarks nightly using tulip-tuning.pl to disable non- or poorly responding landmarks, and re-enable them when they are working well. The PlanetLab landmarks send 10 pings very quickly, whereas the SLAC/Looking Glass landmarks send five 56 byte pings with one second between them, they will also wait for a deadline time of 30 seconds for pings to be replied to.

There is a Google map of PlanetLab and SLAC/Looking Glass landmarks.

Tiering

To reduce the network impact and reduce the initial rough estimate time, we also break the landmarks into two tiers. Tier0 landmarks are used to identify the region for the target. Then tier1 hosts for that region can be used to more exactly locate the target. Tier0 hosts are chosen as being at the edges of the region, being well connected, highly reliable and quick to respond. We currently only define tier0 hosts for North America and Europe. In other regions all the landmarks are regarded as tier0. There are about 8 tier0 hosts for North America and 4 for Europe. This reduces the number of landmarks to make measurements with a tier0 request since there are over 100 landmarks in either North America or Europe.When sending the query to the reflector we can characterize it into three tiers 0,1 and 2. These tiers would be matched with sites.xml nodes and relevant tiering information would be extracted from tag name "tier".

Responses

The responses appear as:

Landmark=http://128.6.192.158, Client=134.79.117.29, failed to connect response code 500
Landmark=http://141.149.218.208, Client=134.79.117.29, 10 packets transmitted, 0 received, 100% packet loss, rtt min/avg/max = 0/0/0
Landmark=http://128.193.33.7, Client=134.79.117.29, 10 packets transmitted, 10 received, 0% packet loss,\
rtt min/avg/max = 29.178/29.2495/29.316
Landmark=http://pinger.fnal.gov/cgi-pub/traceroute.pl?function=ping&target=134.79.16.9, Client=134.79.117.57, 5 packets transmitted,\
5 received, 0% packet loss, rtt min/avg/max = 52/52/53

The first 3 response are from PlanetLab landmarks and the latter is from a SLAC type landmark.

Nb. a negative loss is reported if the target provides duplicate responses (emplification), such as

```
3 packets transmitted, 5 packets received, 1.67 times amplification
```

PlanetLab Landmarks

To access the PlanetLab landmarks one needs a cookie that is associated with a subnet (in our case 134.79/16). In addition one needs a ruby script that is sent to the PlanetLab landmark to execute. These are put together by reflector.cgi to create a URL in hex form.

Errors Reported by PlanetLab

Failed to connect to http://129.22.150.90 response code 500
ERROR: you're (134.79.18.134) already running a measurement on socket 14. http://128.83.122.179
10 packets transmitted, 0 received, 100% packet loss, time 0 ms rtt min/avg/max = 0/0/0 http://141.149.218.208
Can't resolve DNS: submitted:6:in `ip_dst=': unable to resolve $target: running in a chroot without dns support (RuntimeError)
submitted:9: warning: didn't see packet 5 leave: pcap overloaded or server bound to incorrect interface?
To 134.79.16.9 timed out
Error connecting: Connection refused
ERROR: you need a valid scriptroute authentication cookie to use this server, or the cookie you used does not match\
your client IP 134.79.18.163; go to http://www.scriptroute.org/cookies.html to get one.
ERROR: you're (134.79.18.134) already running a measurement on socket 10.
PlanetLab Server Error: Received: IP (tos 0xc0, ttl 253, id 51592, offset 0, flags [none], length: 56)
192.70.187.218 > 198.82.160.220: icmp 36: time exceeded in-transit
Error connecting: No buffer space available
submitted:9:in `send_train': scriptrouted error: unable to send to 137.138.137.177: No buffer space available (ScriptrouteError)

Parsing the SLAC Landmarks

Unfortunately the SLAC landmark server is really designed to be executed from a browser which will render the output, see for example http://www.slac.stanford.edu/cgi-wrap/traceroute.pl?function=ping&target=www.fnal.gov and view the source. Thus not only is it verbose (~3.3KBytes per successful request) but also EventHandler.pm has to carefully parse this human readable output to find the relevant ping output lines with the RTTs and losses. In addition the server uses the standard system ping command and its output which varies in details between OS's and releases so this also has to be accomodated in the parsing.

Logging

In addition to the normal web server (Apache) logging, we use Log4perl for logging. The configuration file is very simple. The following types of error messages can be found in the log file (this is at /scratch/tulip_log on wanmon.slac.stanford.edu and is also available here) together with with time stamped records of all requests, the requesting host, and the target.
2007/09/03 20:02:25 ERROR> EventHandler.pm:70 EventHandler::on_failure - Landmark=http://128.6.192.158,\
Client=134.79.117.29, failed to connect response code 500
2007/09/03 20:02:34 ERROR> EventHandler.pm:142 EventHandler::parseData - Landmark=http://129.22.150.90,\
Client=134.79.117.29, 10 packets transmitted, 0 received, 100% packet loss, rtt min/avg/max = 0/0/0:
2007/09/03 20:09:09 ERROR> EventHandler.pm:115 EventHandler::parseData - Landmark=http://128.143.137.250,\
Client=134.79.117.29, request timed out: To 134.79.16.9 timed out

Plus Unusual PlanetLab errors of the form:

2007/09/03 20:02:58 ERROR> EventHandler.pm:125 EventHandler::parseData - Landmark=http://128.4.36.11,\
Client=134.79.117.29, <planetLab error message, see section "Errors Reported by PlanetLab">

There is a script at /afs/slac/package/netmon/tulip/tulip-log-analyze.pl to analyse the logs. Typical output appears as:

```
28cottrell@wanmon:~>bin/tulip-log-analyze.pl
===============Failure types by landmark ======================
           Landmark, Success,  100%_loss,connect_fail, not_sent, timeout, refused,  in_use,  no_name,
transit_exc, Totals,
    143.225.229.236, 100.0%,  0.0%, 0.0%, 0.0%, 0.0%, 0.0%, 0.0%, 0.0%,        0.0%           1,
      149.48.230.20, 40.0%,  37.8%, 2.2%, 0.0%, 11.1%, 0.0%, 0.0%,         8.9%,         0.0%          45,...
itchy.cs.uga.edu_PL, 0.0%,  15.4%, 84.6%, 0.0%, 0.0%, 0.0%, 0.0%,        0.0%,         0.0%          26,
           Landmark, Success,  100%_loss,connect_fail, not_sent, timeout, refused, in_use,  no_name,
transit_exc,  Totals,
             Totals,  2258, 422, 457, 11, 401, 0, 52, 287, 0, 3888
Wed Oct  3 14:58:38 2007 tulip-log-analyze.pl: took 38 seconds to analyze 4378 records for 323 requests.
Successful hosts=111, Failing hosts=108, PlanetLabs=128(100% success=26), SLACs=16(100% success=10)
```

The log is analyzed to understand usage, look for abusers etc.

Landmark Failures

The typical failure mechanisms for the target www.cern.ch with a timeout of 2 and 10 seconds made in the evening (PDT) of September 8th 2007 is seen in the table below. The multiple numbers in each cell are for different requests. It is seen that increasing the timeout from 2 to 10 seconds does not provide much, if any help. So we utilize a timeout of 2 seconds.

Timeout	2 secs	10 secs
100% loss	7, 10, 9	10, 11, 8
Success	22, 16, 17	20, 14, 21
Fail to connect	10, 8, 9	8, 6, 5
Timeout	45, 50, 49	46, 51, 46

Performance

Some spot measures of performance indicate that for 10 pings per target and 86 PlanetLab landmarks for region=northamerica as we vary the number of landmarks accessed simultaneously, the number of parallel requests per landmark, and the timeout for each request the duration is as follows (n.b. there is a timeout of 100 seconds on the complete process, and the default values are in boldface in the table below:

Simultaneous landmarks	Parallel requests / landmark	Request timeout	Duration (secs)
20	**5**	**2**	**50**
20	5	10	60
10	5	2	88
40	5	2	34
20	10	2	50

Performance Version=3.0

There was a problem found with reflector that the request timeout was fixed at 5sec and the parallel requests was fixed at 10. Modifying the parameters in the code didn't change anything. This was causing two problems, 1) 5 seconds was too less for most of the landmarks causing large number of timeouts and 2) The speed of the script was not fast enough. The timeout was changed to 10sec and the number of parallel requests was increased to 80. During the time of testing there were a total of 200 active landmarks of which 91 were in North America, 15 in South Asia and 49 in Europe.

Region	Tier	Time (seconds)
All	0	10
North America	1	23
Europe	1	35
South Asia	1	17
All	All	65

Testing

It can be tested from a web browser by entering the URL (e.g. from a browser or from wget), e.g.

```
http://www-wanmon.slac.stanford.edu/cgi-wrap/reflector.cgi?region=northamerica&target=134.79.16.9
```

It can also be called from the command line, e.g.
>setenv REMOTE_ADDR 134.79.18.134; perl -d -T bin/reflector.cgi "region=northamerica&target=134.79.16.9&tier=0&type=slac"
However unless you have a PlanetLab cookie for your host it will not fully work. To get around this one can use:

```
ssh www-wanmon /afs/slac/g/www/cgi-wrap-bin/net/shahryar/smokeping/reflector.cgi 'target=www.slac.stanford.edu'
```

Some hosts mis-identified by Geo IP tools and VisualRoute include: www.cst.edu.ve

Security

There are several issues related to security.

Landmark Server

The SLAC traceroute/landmark server that is frequently used by landmarks servers: rejects attempts to traceroute to a broadcast address; does not allow a remote host name to be greater than 255 characters to prevent buffer overflow attempts; does not allow a remote host in a different domain to do a traceroute to a host within the same domain as the web server; limits the maximum number of traceroute processes running in the server to reduce the chance of a denial of service request; starts the traceroute after 3 hops if the client/browser and server are in different domains in order to hide internal routing information from outsiders; has a blacklist of sites that are blocked.

Tulip Client

TULIP only allows one copy of the client to be running on a client host. TULIP also hides the URLs used for the landmarks to reduce the possibility of people gleaning the URLs for a denial of service attack. Editing the landmark URL's requires a password known only to the developers.

Scanning and Denial of Service

A major concern is that the target is pinged simultaneously from multiple landmarks. This can look like a scan of multiple hosts when the target host responds to the ping requests. It can also look like a denial of service attack, especially for hosts with limited available bandwdth, such as are found in developing countries. We thus limit the number of pings from a landmark to a target to 5.

I doubt the early version triggered the alert. It had < 60 landmarks, of these I am guessing (TULIP is down at the moment) about 10-20 did not work (i.e respond to the request to ping). However recently we added 149 PlanetLab hosts. The net result is that with the current version 39 PlanetLab landmarks answer 100% time, 39 answer sometimes, the rest are either not requested or never answer (as far as I can tell this means they are not pinging, i.e. they are not responding to the request to make the pings). The typical number of PlanetLab hosts trying to ping is about 60 (of these about 10 fail with 0 pings responding).

We are working on two things to reduce the number of landmarks pinging at a time.

1. Remove landmarks which are not 100% reliable and whose function is replicated by another landmark (e.g. a nearby working one).
2. We are also looking at tiering the landmarks (see tiering to tier the N. American and European hosts). The top tier will enable us to locate the region of the world and then the second tier can be used to find the location in that region. This reduces the number of landmarks used and divides them in time into two or more sets. Most landmarks are in N. America or Europe (136 out of 149 for PlanetLabs & 26 out of 63 for the SLAC type landmarks). So for tier0 landmarks we use 5 sites in North America, 3 in Europe and all 32 sites outside N. America and Europe. The tier0 sites are first requested to provide the area the host is in and a rough estimate of position. Thus there are currently 5+32+3 tier0 landmark

requests (thiw will be reduced when we remove unreliable landmarks, see above) The client can then request more detailed information of the host if it is in N. America or Europe.

Other Concerns

We have also considered whether the knowledge that a machine and possibly the usual owner can be accurately located may violate some privacy issue. This may require us to add some fuzz to results. So far this has not been done.

Sample Scripts

traceroute.pl: This script has been written with special security considerations so it will help in implementing reflector.cgi

topology.pm: This is a multi-threaded script written by Yee so this will help understand the threading issues in perl which are a bit complex.