2017 IPv6 for PingER

See INC0164829 (for pinger and INC0176266 for www3), also INC0191881 for fixes following upgrading the border.

pinger.slac.stanford.edu and www3.slac.stanford.edu

A quick test to see if the host has full IPv6 capability is to run traceroute6

```
453cottrell@www3:~$traceroute6 ipv6.google.com
traceroute to ipv6.google.com (2607:f8b0:4007:802::200e), 30 hops max, 80 byte packets
connect: Network is unreachable
Exit 1
```

To enable for IPv6:

- Had to get subnet enabled for IPv6 - Mark.
  - To find subnet for the host (e.g. pinger) use NetDB (or ping, or ifconfig on actual host) to find IPv4 address
  - Use http://network.slac.stanford.edu/subnets to find list of subnets
  - Enter first 3 octets of the host (e.g. 134.79.197.) in the filter
- Taylor blocked ipv6 by default for security reasons, so had to request to enable IPv6 for pinger in Taylor (INC0175010) - Karl
  - for taylor.opts on a linux server, I believe would look something like ipv6addr=2620:114:d000:25a1::80/64,2620:114:d000:25a1::1
- Modifications to the PAN rules - Kent
- It needed an IPv6 address in NetDB
  - It may seem to have an IPv6 address by looking at ifconfig you may see

    ```
    eth0      Link encap:Ethernet  HWaddr 00:50:56:BE:3D:4C
    inet addr:134.79.197.214 Bcast:134.79.197.255 Mask:255.255.255.128
    inet6 addr: 2620:114:d000:2716:250:56ff:febe:3d4c/64 Scope:Global
    inet6 addr: fe80::250:56ff:febe:3d4c/64 Scope:Link
    ```

    The global ipv6 address which is currently configured on this host is probably the slaac (auto-configured) from the router.
    If the last couple octets look like a MAC address (versus zero or a very small number), then the address was almost certainly auto-configured.

    The other way to convince yourself is to look at the DNS record for the given system (e.g. “host www3”). If there is no IPv6 address displayed, then the address got autoconfigured.

    ```
    447cottrell@rhel6-64f:~$host www3
    www3.slac.stanford.edu has address 134.79.197.214
    ```

  - If it does not have an IPv6 address then we have to assign one from the relevant range (e.g. for www3)

    ```
    ksa@cdlogin3 $ /afs/slac/g/scs/net/bin/subnet all | grep SERV01-DMZ-WEBSERV
    134.79.197.128    SERV01-DMZ-WEBSERV     134.79.197.129          255.255.255.128       Serv01 Web
    Servers DMZ (vlan 1814)
    2620:114:d000:2716:: SERV01-DMZ-WEBSERV      2620:114:d000:2716::1    ffff:ffff:ffff:ffff:: Serv01
    Web Servers DMZ (vlan 1814)
    ```

  - To test:

    ```
    ksa@www3 $ curl -I -v www3 * About to connect() to www3 port 80 (#0) * Trying 2620:114:d000:2716::
    8... connected * Connected to www3 (2620:114:d000:2716::8) port 80 (#0) [?]
    ```

Modifications Required

pinger2.pl

Modified to enable both the hostname and IPv6 address to be the same. It is integrated with the new xml files generated from NODEDETAILS. It is backward compatible. Version >= 3.0 is IPv6 version.

traceroute.pl

Modified to make work on Solaris and Linux:

my $version="7.3, 12/13/2017, Les Cottrell";
Added \[\] to untainting of dig command. Appears to be needed for IPv6.
Do not avoid testing internal domains if server is IPv6 host,
Added avoid calling gethostbyname6 if hostname is already an ipv6 address
Fixed how Solaris mis-interprets system(@args) sometimes (saw in IPv6)

create-scriptTable.pl

We added the option --inet4-only to the wget command, else the cromjob timed out after 8550 seconds . when the tokens ran out.

getdata.pl

Created nodes-ipv6.pl to add the PingER IPv6 MA 2001:da8:270:2018:f816:3eff:fef3:bd3. See here for addition.

Copied getdata.pl to getdata-ipv6.pl and modified to use the /afs/slac/package/pinger/nodes-ipv6.cf file and write to the /nfs/slac/g/net/pinger/pingerdata /ipv6/data/ directory (e.g. /nfs/slac/g/net/pinger/pingerdata/ipv6/data/2001:da8:270:2018:f816:3eff:fef3:bd3/ping-2017-12-19.txt.gz) when running getdata-ipv6.pl 2001:da8:270:2018:f816:3eff:fef3:bd3 2017-12-19 1. Also replaced IPv4 address checks with sub chck_ip{} to test for both IPv4 and IPv6 addresses.

We added the option --prefer-family=ipv4 to the wget options. This redcuced the time for */afs/slac/package/pinger/getdata_all.pl -h pinger.slac.stanford.edu -D 0* from 190 seconds to 10 10 seconds.

10 seconds.

wrap-analyze-hourly.pl

~~Call with:~~

~~wrap-analyze-hourly.pl --usemetric --dataset ipv6 --by by-node --size 100 --set_metric 3 --date 2017-12-19~~

~~This will use as input:~~

- ~~/nfs/slac/g/net/pinger/pingerdata/<dataset>/data/<nodename>/ping-<year>-<mm>-<dd>.txt.gz~~
- ~~or more specifically /nfs/slac/g/net/pinger/pingerdata/ipv6/data/2001:da8:270:2018:f816:3eff:fef3:bd3/ping-2017-12-19.txt.gz~~

~~Output will go to:~~

- ~~/nfs/slac/g/net/pinger/pingerreports/<dataset>/<metrics>/<metric>-<size>-<by-node>-<yyyy><mm><dd>.txt.gz~~
- ~~or more specifically /nfs/slac/g/net/pinger/pingerreports/ipv6/minimum_rtt/minimum_rtt-100-by-node-2017-12-19.txt.gz~~

We decided, initially to stick with a single data set (hep) and wrap-analyze-hourly now allows both IPv4 and IPv6 addresses by using the valid_ip function. Later we may also support a seprate ipv6 dataset.

The other analyze scripts (daily, monthly and yearly) did not need modification.

APEX/Oracle user interface to PingER NODEDETAILS database

This did not accept IPv6 Addresses, see INC0176849 and INC0176966. The NODEDETAILS Oracle Apex database now accepts IPv6 addresses as the name (in case there is no name for the host) and as the IP address. It is backward compatible and also (as before) accepts IPv4 addresses in both fields.

```
%NODE_DETAILS=(
"2001:da8:270:2018:f816:3eff:fef3:bd3" => [
"2001:da8:270:2018:f816:3eff:fef3:bd3",
"gzhu.edu.cn",
"CN.GZHU.EDU.IPV6",
"CERNET Cloud",
"Tsinghua University, Beijing",
"China",
"East Asia",
"23.037002 113.36777",
"M",
"http://[2001:da8:270:2018:f816:3eff:fef3:bd3]/cgi-bin/traceroute.pl?function=ping",
"http://[2001:da8:270:2018:f816:3eff:fef3:bd3]/cgi-bin/traceroute.pl?choice=yes",
"http://[2001:da8:270:2018:f816:3eff:fef3:bd3]/cgi-bin/ping_data.pl",
"http://www.gzhu.edu.cn/",
"",
"",
"",
"Saqib Ali <saqibutm\@outlook.com>, Prof. Guojun Wang, csgjwang\@gzhu.edu.cn; csgjwang\@gmail.com",
"Test record by Venkat based on add by Cottrell 12/19/2017.",
],
```

Nodename (the primary key)

APEX doesn't like colon(:) in primary key column values. Primary key value is passed in the URL as a parameter and APEX parameters are separated by colons (:), which is causing the issue. Venkat provided a workaround, which is backward compatible as far as the user is concerned.

ipaddress

Ipaddress is limited to 15 characters and expecting a format of ^[[:digit:]]{1,3}\.[[:digit:]]{1,3}\.[[:digit:]]{1,3}\.[[:digit:]]{1,3}$

If you want to increase the length or change the format, we need to modify the interface code. We extended the field to 100 characters (like the other fields) and not apply any filters. This was done.

Testing

The connection to the database is defined in: /afs/slac/package/pinger/oracleArchive/netratDb.pm

In that program the following lines contain slacprod/SLACPROD

- ian@rhel6-64e $ grep -i slacprod netratDb.pm
- # to all the scripts that store in the SLACPROD database for the user
- $ENV{'TWO_TASK'} = 'SLACPROD';
- $dbh = DBI->connect('dbi:Oracle:dbname=SLACPROD;host=slacprod;SID=SLACPROD', $usr, $pwd,

I made a copy of the script in /afs/slac/package/pinger/oracleArchive/netratDbdev2.pm and substituted slacdev2/SLACDEV2 for slacprod/SLACPROD and also made a copy of /afs/slac/package/pinger/guthrie/node.pl in /afs/slac/package/pinger/guthrie/node-ipv6.pl to call the new script. Then I saved the new file using /afs/slac/package/pinger/guthrie/node-ipv6.pl -o > /afs/slac/package/pinger/nodes-ipv6.cf

The password is kept in a protected file. The password on SLACDEV2 was changed to match the one on SLACPROD

Accessing SLACPROD and SLACDEV2 from the web as follows, they use the same windows userid/passwords

- To access the SLACPROD production copy of the NODEDETAILS database go to:
  - **https://oraweb.slac.stanford.edu/apex/slacprod/f?p=123:2:15892093406016::NO:2::**
- To access the SLACDEV2 test copy of the NODETATILS database go to:
  - https://oraweb.slac.stanford.edu/apex/slacdev2/f?p=123:2:15892093406016::NO:2::

write_offssitenodes.pl

The script write_offsitenodes.pl creates the configuration file /afs/slac.stanford.edu/package/pinger/pinger2/share/pinger/pinger.xml for the SLAC MA.

We modified NodeIPCheck-new.pl to add a subroutine to use dig to find the IPv6 address of a host and also to validate the hostname and Iv6 address. These were used in script write_offsitenodes.pl.

node.pl

There were no changes.

ping_data_plot.pl

Need to add a colon (:) to the list of characters allowed by cgi-wrap. This goes in the rules file.

Typical PingER Raw Data

ipv6.google.com 2607:f8b0:4007:802::200e 56 1524939147 10 10 9.632 9.690 9.860 1 2 3 4 5 6 7 8 9 10 9.63 9.63 9.74 9.86 9.66 9.65 9.70 9.67 9.64 9.67

IPv6 Targets

See Validating ICMP ping measurements against TCP nping measurements

One can use https://network-tools.webwiz.net/ping.htm to check if a target responds to IPv6 pings. As part of this we discovered that the SLAC border was blocking IPv6 pings (see INC0178575). This was fixed.

We started by adding ipv6.google.com(2607:f8b0:4007:802::200e) to NODEDETAILS in the SLACPROD database. After running

/afs/slac/package/pinger/guthrie/node.pl -o > /afs/slac/package/pinger/nodes.cf, we got the following in nodes.cf.

```
    "ipv6.google.com" => [
    "2607:f8b0:4007:802::200e",
    "Google.com",
    "COM.GOOGLE.IPV6",
    "GooglePlex",
    "Mountain View, California",
    "United States",
    "North America",
    "37.4220 -122.0841",
    "NOT-SET",
    "NOT-SET",
    "NOT-SET",
    "NOT-SET",
    "http://www.google.com",
    "",
    "",
    "",
    "",
    "Testing IPv6 support, add by Cottrell 1/31/2018.",
  ],
```

We also ran /afs/slac/package/pinger/write_offsitenodes.pl to generate /afs/slac.stanford.edu/package/pinger/pinger2/share/pinger/pinger.xml file for the SLAC MA. This works.

For the IPv6.google.com raw data from pinger.pl only reports min/avg/max RTTs, there are no sequence number or associated RTTs. This has to do with truncating the ping/icmp MTU between SLAC and ipv6.google.com and no RTT being reported for each of the pings. On the other hand pings are not truncated between SLAC and perfsonar-lt.cern.ch (2001:1458:301:a7bc::100:15) and many other IPv6 nodes .See below for examples. I was just unlucky in the host I chose to test against.

```
Examples (nb. –s defines the packet size)
123cottrell@pinger:~$/bin/ping6 -n -w 30 -c 2 -i 1 -s 64 ipv6.google.com
PING ipv6.google.com(2607:f8b0:4007:80d::200e) 64 data bytes
72 bytes from 2607:f8b0:4007:80d::200e: icmp_seq=1 ttl=51 time=9.88 ms
72 bytes from 2607:f8b0:4007:80d::200e: icmp_seq=2 ttl=51 time=9.78 ms

--- ipv6.google.com ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1011ms
rtt min/avg/max/mdev = 9.780/9.834/9.888/0.054 ms
124cottrell@pinger:~$/bin/ping6 -n -w 30 -c 2 -i 1 -s 65 ipv6.google.com
PING ipv6.google.com(2607:f8b0:4007:80d::200e) 65 data bytes
72 bytes from 2607:f8b0:4007:80d::200e: icmp_seq=1 ttl=51 (truncated)
72 bytes from 2607:f8b0:4007:80d::200e: icmp_seq=2 ttl=51 (truncated)

--- ipv6.google.com ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1011ms
rtt min/avg/max/mdev = 9.818/9.834/9.850/0.016 ms
125cottrell@pinger:~$/bin/ping6 -n -w 30 -c 2 -i 1 -s 1000 perfsonar-lt.cern.ch
PING perfsonar-lt.cern.ch(2001:1458:301:a7bc::100:15) 1000 data bytes
1008 bytes from 2001:1458:301:a7bc::100:15: icmp_seq=1 ttl=45 time=151 ms
1008 bytes from 2001:1458:301:a7bc::100:15: icmp_seq=2 ttl=45 time=152 ms

--- perfsonar-lt.cern.ch ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1152ms
rtt min/avg/max/mdev = 151.987/151.996/152.006/0.389 ms
```

This resulted in PingER's raw data measurements reporting:

```
pinger.slac.stanford.edu 2620:114:d000:25a1::80 ipv6.google.com 2607:f8b0:4007:802::200e 100 1517962635 10 10
9.617 9.704 9.780
pinger.slac.stanford.edu 2620:114:d000:25a1::80 2607:f8b0:4007:802::200e 2607:f8b0:4007:802::200e 100
1517962640 10 10 9.594 9.648 9.705
```

and automated daily email warnings of the form:

```
/afs/slac/package/pinger/smokeping/SelectConvSrcDest1.pl
--mon pinger.slac.stanford.edu #takes 520 mins (10/10/2011)
produced the following output:
decide_Action: create_File: start_Converting(37):
add_ToRrd: warning: only min/avg/max in pinger.slac.stanford.edu
2620:114:d000:25a1::80 2607:f8b0:4007:802::200e 2607:f8b0:4007:802::200e 100
1517788034 10 10 9.519 9.574 9.619 , skipping pair
```

Thus I added perfsonar-lt.cern.ch to NODEDETAILS. and got:

```
pinger.slac.stanford.edu 2620:114:d000:25a1::80 perfsonar-lt.cern.ch 2001:1458:301:a7bc::100:15 100 1518025122
10 10 151.850 152.004 152.372 1 2 3 4 5 6 7 8 9 10 152 151 151 151 152 151 151 151 152 151
```

For testing I also added 128.142.223.247 (the IPv4 address for perfsonar-lt.cern.ch) to NODEDETAILS. Finally I set the ping packet length for ipv6.google.com to 56Bytes in NODEDETAILS and now get the individual sequence of ping RTTs.

```
pinger.slac.stanford.edu 2620:114:d000:25a1::80 ipv6.google.com 2607:f8b0:4007:802::200e 56 1518450162 10 10
9.643 9.673 9.692 1 2 3 4 5 6 7 8 9 10 9.66 9.69 9.64 9.68 9.66 9.67 9.69 9.64 9.68 9.68
pinger.slac.stanford.edu 2620:114:d000:25a1::80 2607:f8b0:4007:802::200e 2607:f8b0:4007:802::200e 56 1518450162
10 10 9.586 9.674 9.755 1 2 3 4 5 6 7 8 9 10 9.58 9.68 9.70 9.66 9.64 9.71 9.72 9.75 9.62 9.65
```

XML file at Guangzhou

See here. Currently, it contains random pingable IPv6 addresses as Saqib took it from http://www.ipv6forum.com/ipv6%5fenabled/approval%5flist.php. He proceeded as follows:

1. I am trying to make a list of IPv6 pingable sites of the education institutes of the world. For this, I have taken the list of top 1000 universities of the world (http://cwur.org/2017.php)
2. Next I am validating each site on http://ipv6-test.com/validate.php for AAAA record.
3. When I get the IPv6 address, I checked its ping status on https://network-tools.webwiz.net/ping.htm . Currently, I do not have a direct access to IPv6 machine.
4. If the address is pingable, I add its details in the attached excel sheet.
5. Latitude and Longitude are calculated using https://www.iplocation.net/index.php
6. The XML file in use on the Measurement Agent is available.
7. I have added a few sites in the attached file. However, I am facing problems in finding Lat/log of the server as it seems that they are not located on the campus of the University. **Kindly check the remarks column of the US universities.**

East Carolina University list

Quality of IPv6 Enablement of Universities: An International Study by John Pickard and Anne Y Patric of East Carolina University has a list of about 1000 world universities and their IPv6 addresses if they have them. It uses IT sonar agents from Nephos6, to perform "user experience" measurements accessing select websites from multiple global geographic vantage points. About 125 hosts have IP addresses. We wrote a perl script (ping-ipv6.pl) to ping all those IPv6 addresses from an IPv6 host at SLAC and about 67 respond to pings. The script runs against the file of ipv6 universities exported from the Excel spreadsheet as a tab-delimited file (top-uni.txt). Unfortunately, many of these hosts were proxies at unknown (to us) places.

perfSONAR

Unfortunately it is very common that sites use a proxy for common services such as www. A possibility is to use perfSONAR. These are high-performance throughput monitoring sites mainly in US, and Europe. perfSONAR fully supports IPv6 and has a database that includes latitude and longitude. When we were working on TULIP (the use of pings from ping servers to find the location of targets) we used these perfSONAR ping/traceroute servers/landmarks to make the RTT measurements. There is some documentation at https://confluence.slac.stanford.edu/display/IEPM/Automated+PerfSONAR+Landmark+finding

There is a spreadsheet of active landmarks that can make pings at http://www-wanmon.slac.stanford.edu/cgi-wrap/reflector.cgi?function=landmarks&ability=1. I am not sure how current this is, the psonar_auto.pl job updates the web page information and runs monthly. Tulip has not been maintained for a couple of years. There is more documentation at https://confluence.slac.stanford.edu/display/IEPM/TULIP+Analysis and looking at http://www-iepm.slac.stanford.edu/pinger/crontab-slaconly.txt you can see the tulip jobs that are run by cron.

In the spreadsheet there is a column labelled Ping URL. It contains the URL of the ping server for the Measurement Agent (MA). However this only yielded a few (53) hosts.

We, therefore, wrote a perl script json-to-xml.pl to read the perfSONAR JSON configuration (see information) and produce in STDOUT a CSV file of hosts with IPv4 and IPv6 addresses (e.g. ps-v4-lc.cf, lc stands for created by Les Cottrell, as opposed to sq created by Saqib Ali).These files were used as config files inputted to ping-vs-tcp-ps.pl that measured the ICMP and TCP RTTs to each of the hosts and outputted the information to STDOUT (e.g. ping-vs-tcp-ps.pl --conf ps-v4-lc.cf | tee ps-v4-sl, sl stands for the MA at SLAC as opposed to ch for China, th for Thailand) that were filtered using for example grep summary ps-v4-sl.txt >! ps-v4-sl.csv and then imported to Excel analyzed and saved as ps-v4-sl.xlsx

SLAC IPv4 Pinger hosts

Using %NODE_DETAILS we wrote a script ping-vs-tcp.pl to read a configuration file of hosts to ping and nping (measures TCP RTT for SYN and return ACK). This yielded about 53 successful hosts for which we provided the name(ipv6 address), lat long, country.

SLAC Hosts with IPv6 support

Pinger (Linux), www1 (Solaris), www3 (Linux), nospam2-out (Linux), pinger

```
Looks like nospam2 is working for me.
this is the ipv6 address for nospam2:
```ksa@nospam2 $ host nospam2-out
nospam2-out.slac.stanford.edu has IPv6 address 2620:114:d000:2598::10
ksa@nospam2 $ ```

```ksa@nospam2 $ host google.com | grep IPv6
google.com has IPv6 address 2607:f8b0:4005:808::200e
ksa@nospam2 $ ping6 2607:f8b0:4005:808::200e
PING 2607:f8b0:4005:808::200e(2607:f8b0:4005:808::200e) 56 data bytes
64 bytes from 2607:f8b0:4005:808::200e: icmp_seq=1 ttl=53 time=2.34 ms
64 bytes from 2607:f8b0:4005:808::200e: icmp_seq=2 ttl=53 time=2.24 ms
64 bytes from 2607:f8b0:4005:808::200e: icmp_seq=3 ttl=53 time=2.27 ms
^C
--- 2607:f8b0:4005:808::200e ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2638ms
rtt min/avg/max/mdev = 2.248/2.290/2.347/0.057 ms
ksa@nospam2 $
```

www1 also works:
435cottrell@www1:~>ping -s -A inet6 -a 2607:f8b0:4005:805::200e
PING 2607:f8b0:4005:805::200e: 56 data bytes
64 bytes from 2607:f8b0:4005:805::200e: icmp_seq=0. time=2.58 ms
64 bytes from 2607:f8b0:4005:805::200e: icmp_seq=1. time=2.37 ms
64 bytes from 2607:f8b0:4005:805::200e: icmp_seq=2. time=2.46 ms
^C
----2607:f8b0:4005:805::200e PING Statistics----
3 packets transmitted, 3 packets received, 0% packet loss
round-trip (ms) min/avg/max/stddev = 2.37/2.47/2.58/0.10

However ns-ext1, pinger do not work:
436cottrell@ns-ext1:~$ping6 2607:f8b0:4005:805::200e
PING 2607:f8b0:4005:805::200e(2607:f8b0:4005:805::200e) 56 data bytes

--- 2607:f8b0:4005:805::200e ping statistics ---
12 packets transmitted, 0 received, 100% packet loss, time 10999ms

Exit 1
436cottrell@pinger:~$ping6 2607:f8b0:4005:808::200e
connect: Network is unreachable
Exit 2

Guangzhou hosts

The MA at Guangzhou (installed July 2017) is not working since November due to security concerns. Saqib has an IPv6 MA at CERNET in Beijing (2001:da8:270:2018:f816:3eff:fef3:bd3) however it is not pingable from outside China or from the IHEP host at Beijing.