Dr. Thomas Dreibholz
Computer Networking Technology Group
Institute for Experimental Mathematics
Ellernstraße 29, 45326 Essen, Germany
☎ +49-201-183-7637
+49-201-183-7673
dreibh@iem.uni-due.de
☞ http://www.iem.uni-due.de/~dreibh

UNIVERSITÄT DUISBURG ESSEN

June 8, 2010

SCTP and Reliable Server Pooling

A Practical Exercise

SAMPLE SOLUTION

Abstract

The intention of this exercise is to obtain basic knowledge of the SCTP protocol [Ste07] and the configuration and application of Reliable Server Pooling (RSerPool) [LOTD08, Dre07, DR08b]. This exercise covers SCTP association setup, data transport, association teardown and multi-homing as well as setting up an RSerPool scenario with the protocols ASAP [SXST08a], ENRP [XSS$^+$08] and example applications.

Contents

1 Lab Setup and Preparations

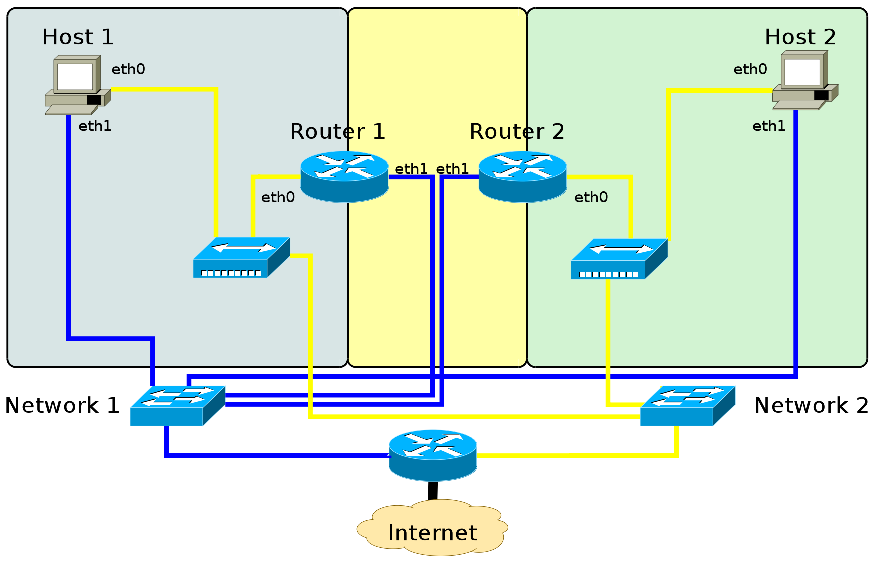


Figure 1: The Basic Networking Lab Setup

Figure 1 illustrates the networking lab setup. The lab PCs are connected to two independent networks: network 1 (blue cables, interface *eth1*) and network 2 (yellow cables, interface *eth0*). The router provides IPv4 (see [Pos81]) and IPv6 (see [DH98]) connectivity to the Internet. IPv4 addresses are provided by a DHCP server, IPv6 prefixes are provided by the router (see [CDG06, NNS98]).

Please take care of the following rules:

- Do not reboot or turn off the PCs. They may be used for simulation runs in background.
- For the same reason, do not disconnect or reconfigure network 1.
- At the end of the exercise, please reconfigure the PCs to the basic setup.

2 The SCTP Protocol

The SCTP protocol [Ste07] is the foundation of Reliable Server Pooling. Therefore, we will have a look at the basics of this protocol first.

2.1 Preparations

To get a practical insight into the functionalities of SCTP, the tool `sctp_darn` will be used in this exercise. Before applying the tool, a few basic settings should be performed. The Linux kernel SCTP module should already be loaded on the lab PCs. If it is not loaded, this can be done manually by:

```
sudo modprobe sctp
```

The default SCTP heartbeat interval is 30000ms. For our exercise, this default is a little bit too large (and the tool cannot set the interval by itself). Therefore, we set the heartbeat interval to 3000ms by:

```
sudo sysctl net.sctp.hb_interval=3000
```

After these settings, the PC is ready for some tests with `sctp_darn`. Using the command

```
man sctp_darn
```

you can get a description of the parameters for `sctp_darn`.

2.2 Association Setup and Data Transmission

At first, run WIRESHARK to capture all traffic on the *any* pseudo-interface. You can use "sctp" as filter rule to see the SCTP traffic only. After having started WIRESHARK, run an SCTP receiver on port 1234 by:

```
sctp_darn -H :: -P 1234 -l
```

The receiver will accept association requests on any of its network interfaces and receive messages.

On another PC, start a sender using the following command:

```
sctp_darn -H :: -P 2345 -h <Remote IP> -p 1234 -s -I
```

The parameter "-I" denotes the interactive mode. In this mode, you can interactively call commands (like sending data) or change parameters (like the primary path). Using "?" as command, you can get an overview of all possible commands.

Now, let the sender transmit a 10,000 bytes message by the following command:

```
snd=10000
```

Since there is no association established yet, SCTP will establish an association first. After that, the message is sent.

Question 1:
Which type of SCTP chunks can you observe on the WIRESHARK trace?

Solution: First, the association is established using the 4-way handshake. You can observe INIT, INIT_ACK, COOKIE_ECHO and COOKIE_ACK chunks for this purpose. After that, data is sent by DATA chunks and acknowledged by SACK chunks.

Question 2:
What are the main differences to a comparable TCP session for the data transport?

Solution: TCP only uses a 3-way handshake, the data stream simply goes into the payload of the TCP packet and the acknowledgement number is simply a field of the header. SCTP uses a 4-way handshake, there are DATA chunks for the payload data and SACK chunks for the acknowledgements.

In particular: Using SCTP, the message borders are preserved. Using TCP, you only get a byte stream.

Example SCTP 4-way handshake:

```
Internet Protocol, Src: 132.252.156.11 (132.252.156.11),
                   Dst: 132.252.156.10 (132.252.156.10)
Stream Control Transmission Protocol, Src Port: 5678 (5678),
                                      Dst Port: 1234 (1234)
    Source port: 5678
    Destination port: 1234
    Verification tag: 0x00000000
    Checksum: 0xd331c2ac [correct CRC32C]
    INIT chunk (Outbound streams: 10, inbound streams: 65535)
        Chunk type: INIT (1)
        Chunk flags: 0x00
        Chunk length: 100
        Initiate tag: 0x7a7d4573
        Advertised receiver window credit (a_rwnd): 55296
        Number of outbound streams: 10
        Number of inbound streams: 65535
        Initial TSN: 4260146167
        IPv6 address parameter
           (Address: 2001:638:501:4ef3:211:43ff:febb:cd56)
        IPv6 address parameter
           (Address: 2001:638:501:4ef4:201:2ff:fe1f:505)
        IPv4 address parameter (Address: 132.252.156.141)
        IPv4 address parameter (Address: 132.252.156.11)
        Supported address types parameter (Supported types: IPv4, IPv6)
        ECN parameter
        Forward TSN supported parameter
        Adaptation Layer Indication parameter (Indication: 0)

Internet Protocol, Src: 132.252.156.10 (132.252.156.10),
                   Dst: 132.252.156.11 (132.252.156.11)
Stream Control Transmission Protocol, Src Port: 1234 (1234),
                                      Dst Port: 5678 (5678)
    Source port: 1234
    Destination port: 5678
    Verification tag: 0x7a7d4573
    Checksum: 0x8011602b [correct CRC32C]
    INIT_ACK chunk (Outbound streams: 10, inbound streams: 10)
        Chunk type: INIT_ACK (2)
        Chunk flags: 0x00
        Chunk length: 388
        Initiate tag: 0xff9a2bae
        Advertised receiver window credit (a_rwnd): 55296
        Number of outbound streams: 10
        Number of inbound streams: 10
        Initial TSN: 4131966131
        IPv4 address parameter (Address: 132.252.156.10)
        IPv6 address parameter
           (Address: 2001:638:501:4ef3:211:43ff:febb:cebd)
        IPv6 address parameter
           (Address: 2001:638:501:4ef4:250:4ff:fe4b:bb79)
        IPv4 address parameter (Address: 132.252.156.140)
        State cookie parameter (Cookie length: 292 bytes)
        ECN parameter
        Forward TSN supported parameter
        Adaptation Layer Indication parameter (Indication: 0)

Internet Protocol, Src: 132.252.156.11 (132.252.156.11),
                   Dst: 132.252.156.10 (132.252.156.10)
Stream Control Transmission Protocol, Src Port: 5678 (5678),
                                      Dst Port: 1234 (1234)
    Source port: 5678
    Destination port: 1234
    Verification tag: 0xff9a2bae
    Checksum: 0x7d571650 [correct CRC32C]
    COOKIE_ECHO chunk (Cookie length: 292 bytes)
        Chunk type: COOKIE_ECHO (10)
        Chunk flags: 0x00
        Chunk length: 296
        Cookie: 9937AA255AFAF47503C19540F202AEFA0000000000000000...

Internet Protocol, Src: 132.252.156.10 (132.252.156.10),
                   Dst: 132.252.156.11 (132.252.156.11)
Stream Control Transmission Protocol, Src Port: 1234 (1234),
                                      Dst Port: 5678 (5678)
    Source port: 1234
    Destination port: 5678
    Verification tag: 0x7a7d4573
    Checksum: 0x56d4d990 [correct CRC32C]
    COOKIE_ACK chunk
        Chunk type: COOKIE_ACK (11)
        Chunk flags: 0x00
        Chunk length: 4
```

Question 3:

Have a look into the DATA and SACK chunks! What is the difference between Stream Sequence Number (SSN) and Transport Sequence Number (TSN)? Why is there no SSN or Stream ID necessary in the SACK chunk?

Solution: The TSN is a sequence number for a transmitted DATA chunk, the SSN is the sequence number within a stream. In this example, only stream #0 is used. For reordering the packets within their stream, the receiver needs the SSN. For the detection of packet losses – and therefore for the acknowledgement mechanism – only the TSN is necessary: it identifies a DATA chunk, which can be lost or acknowledged.

Example DATA and SACK messages:

```
Internet Protocol, Src: 132.252.156.11 (132.252.156.11),
                   Dst: 132.252.156.10 (132.252.156.10)
Stream Control Transmission Protocol, Src Port: 5678 (5678),
                                      Dst Port: 1234 (1234)
    Source port: 5678
    Destination port: 1234
    Verification tag: 0xff9a2bae
    Checksum: 0x6a77a710 [correct CRC32C]
    DATA chunk(ordered, last segment, TSN: 4260146174, SID: 0,
               SSN: 0, PPID: 0, payload length: 176 bytes)
        Chunk type: DATA (0)
            0... .... = Bit: Stop processing of the packet
            .0.. .... = Bit: Do not report
        Chunk flags: 0x01
            .... ...1 = E-Bit: Last segment
            .... ..0. = B-Bit: Subsequent segment
            .... .0.. = U-Bit: Ordered deliviery
        Chunk length: 192
        TSN: 4260146174
        Stream Identifier: 0x0000
        Stream sequence number: 0
        Payload protocol identifier: not specified (0)
Data (10000 bytes)
    Data: 2122232425262728292A2B2C2D2E2F303132333435363738...

Internet Protocol, Src: 132.252.156.10 (132.252.156.10),
                   Dst: 132.252.156.11 (132.252.156.11)
Stream Control Transmission Protocol, Src Port: 1234 (1234),
                                      Dst Port: 5678 (5678)
    Source port: 1234
    Destination port: 5678
    Verification tag: 0x7a7d4573
    Checksum: 0x53173b31 [correct CRC32C]
    SACK chunk (Cumulative TSN: 4260146174, a_rwnd: 55296,
                gaps: 0, duplicate TSNs: 0)
        Chunk type: SACK (3)
```

```
            0... .... = Bit: Stop processing of the packet
            .0.. .... = Bit: Do not report
        Chunk flags: 0x00
            .... ...0 = Nounce sum: 0
        Chunk length: 16
        Cumulative TSN ACK: 4260146174
        Advertised receiver window credit (a_rwnd): 55296
        Number of gap acknowledgement blocks: 0
        Number of duplicated TSNs: 0
```

2.3 Multi-Homing

One of the most interesting features of SCTP is the multi-homing (see also [Jun05]). Since all lab PCs have two network interfaces – with each one having an IPv4 as well as an IPv6 address – there are four different paths in each direction.

Question 4:
How are the possible paths signalled at the association setup?

Solution: The paths are signalled by the INIT chunk (in direction from sender node to receiver node) and INIT_ACK chunk (in direction from receiver node to sender node). Both types of chunks contain the chunk sender's list of IPv4 and IPv6 addresses.

Question 5:
Observe the SCTP association in WIRESHARK for some seconds. How is the usability of each path checked by SCTP?

Solution: There are HEARTBEAT chunks transmitted over each path. They are answered using HEARTBEAT_ACK chunks. After a configurable threshold is reached for the number of missing answers on a path, the path is assumed to be unusable.

Ensure that the primary path goes over network 2 (yellow cable). You can explicitly set the primary path by `primary=<Remote IP>` and test this setting by sending a few more messages. After that, unplug the yellow cable and again send some messages.

Question 6:
What can you observe in the WIRESHARK trace?

Solution: First, the PC tries to send the data (in DATA chunks) via network 2. However, there is no SACK chunk to acknowledge (since the network link is broken). Therefore, SCTP decides to use another path. By sending over network 1 (blue cable), the data will be successfully transmitted and acknowledged.

2.4 Association Teardown

In order to finally perform a shutdown of the association, you can e.g. stop the sender process by typing <Ctrl>+C.

Question 7:
How is the association teardown signalled by SCTP?

Solution: The association teardown is signalled by three chunks: SHUTDOWN, SHUTDOWN_ACK and SHUTDOWN_COMPLETE.

3 The Reliable Server Pooling Framework

After learning the basics of the SCTP transport protocol, we will now have a look at an important SCTP application: Reliable Server Pooling (RSerPool), which is described in [LOTD08, Dre07]. As implementation, we use the Open Source RSPLIB [Dre10c, Dre07] package. It is already installed on the lab PCs.

3.1 Setting Up a Basic Scenario

The first thing we need is a registrar[1] (PR). Negotiate with the other participants which PC in the lab should become the first PR. Run WIRESHARK on this PC (sniffing on the *any* pseudo-interface) and start the PR process by:

```
registrar
```

After starting the PR, choose another PC for running a PE process. Also, run WIRESHARK and start a PE for the Fractal Generator Service [Dre07, section 5.7]:

```
server -fractal
```

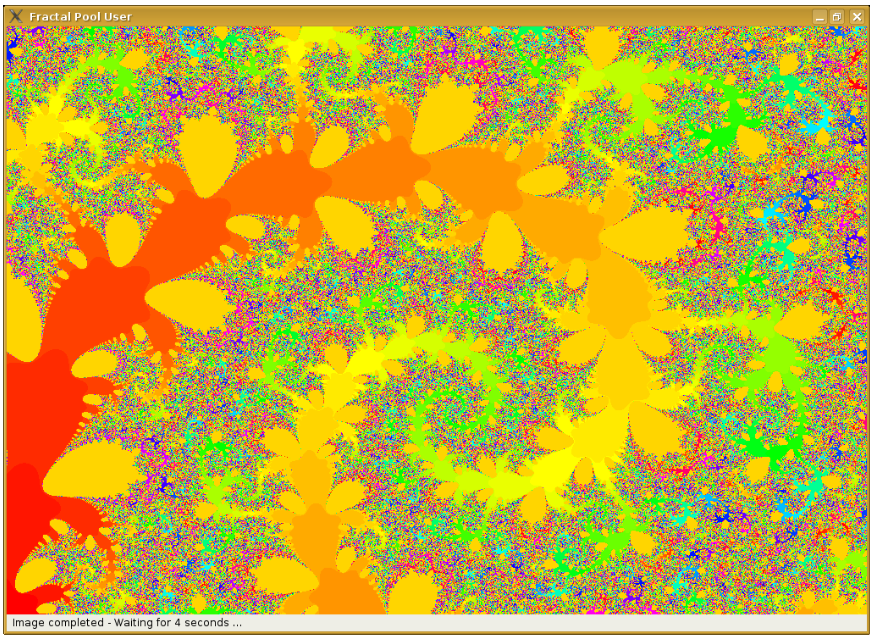


Figure 2: Screenshot of the Fractal Pool User

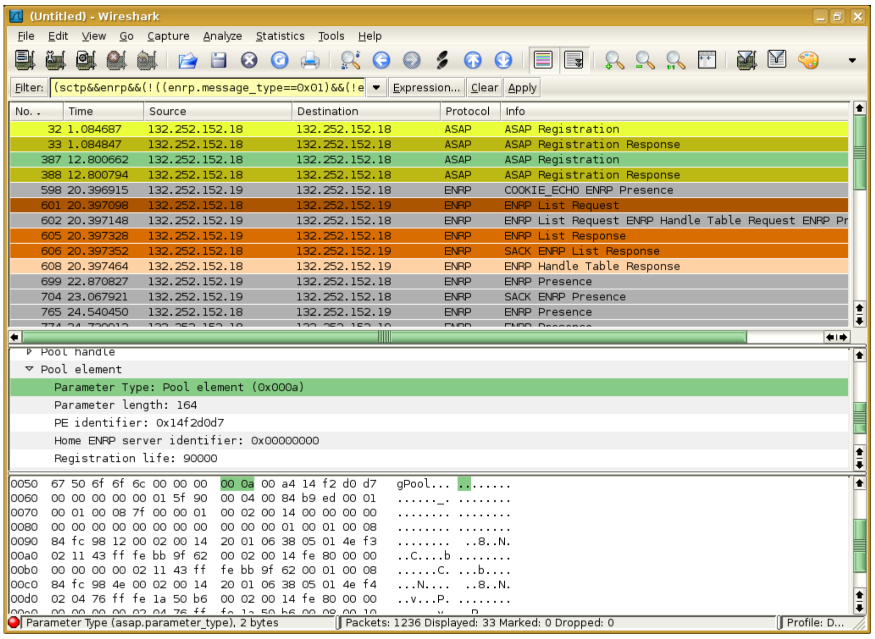
Finally, choose a third PC to run WIRESHARK and the Fractal Generator Service PU:

```
fractalpooluser
```

Note: There are manual pages for all RSPLIB programs, describing their possible options. Just have a look using `man <program>`!

[1] In the drafts, "registrar" is denoted as ENRP server. This terminology is in fact wrong – since ENRP has peers, but no designated clients or servers. However, due to "standardization tactics", the term "registrar" – which is also used in SIP signalling – had to be avoided for this reason.

You should now observe that the initial scenario is running and the PU should display the progress of the image calculation, as illustrated in figure 2. Also, on the WIRESHARK outputs, you should see the protocols ASAP [SXST08a], ENRP [XSS$^+$08] and Fractal Generator Protocol (FGP). Use the pre-defined filters to select specific types of packets. Also, useful colouring rules are provided to make observing the RSerPool and application traffic illustrative.



```
(sctp&&enrp&&(!((enrp.message_type==0x01)&&(!e
```

```
No. .  Time       Source          Destination     Protocol  Info
   32  1.084687   132.252.152.18  132.252.152.18  ASAP      ASAP Registration
   33  1.084847   132.252.152.18  132.252.152.18  ASAP      ASAP Registration Response
  387 12.800662   132.252.152.18  132.252.152.18  ASAP      ASAP Registration
  388 12.800794   132.252.152.18  132.252.152.18  ASAP      ASAP Registration Response
  598 20.396915   132.252.152.19  132.252.152.18  ENRP      COOKIE_ECHO ENRP Presence
  601 20.397098   132.252.152.18  132.252.152.19  ENRP      ENRP List Request
  602 20.397148   132.252.152.19  132.252.152.18  ENRP      ENRP List Request ENRP Handle Table Request ENRP Pr
  605 20.397328   132.252.152.19  132.252.152.18  ENRP      ENRP List Response
  606 20.397352   132.252.152.18  132.252.152.19  ENRP      SACK ENRP List Response
  608 20.397464   132.252.152.18  132.252.152.19  ENRP      ENRP Handle Table Response
  699 22.870827   132.252.152.19  132.252.152.18  ENRP      ENRP Presence
  704 23.067921   132.252.152.19  132.252.152.18  ENRP      SACK ENRP Presence
  765 24.540450   132.252.152.18  132.252.152.19  ENRP      ENRP Presence
```

```
 ▷ Pool handle
 ▽ Pool element
     Parameter Type: Pool element (0x000a)
     Parameter length: 164
     PE identifier: 0x14f2d0d7
     Home ENRP server identifier: 0x00000000
     Registration life: 90000
```

```
0050  67 50 6f 6f 6c 00 00 00  00 0a 00 a4 14 f2 d0 d7   gPool... ........
0060  00 00 00 00 00 01 5f 90  00 04 00 84 b9 ed 00 01   ......_. ........
0070  00 01 00 08 7f 00 00 01  00 02 00 14 00 00 00 00   ........ ........
0080  00 00 00 00 00 00 00 00  00 00 00 01 00 01 00 08   ........ ........
0090  84 fc 98 12 00 02 00 14  20 01 06 38 05 01 4e f3   ........  ..8..N.
00a0  02 11 43 ff fe bb 9f 62  00 02 00 14 fe 80 00 00   ..C....b ........
00b0  00 00 00 00 02 11 43 ff  fe bb 9f 62 00 01 00 08   ......C. ...b....
00c0  84 fc 98 4e 00 02 00 14  20 01 06 38 05 01 4e f4   ...N....  ..8..N.
00d0  02 04 76 ff fe 1a 50 b6  00 02 00 14 fe 80 00 00   ..v...P. ........
```

```
Parameter Type (asap.parameter_type), 2 bytes | Packets: 1236 Displayed: 33 Marked: 0 Dropped: 0 | Profile: D...
```

Figure 3: Observing RSerPool Traffic with WIRESHARK

Note: The filters and colouring rules are provided as part of the RSPLIB source package. Just copy the files `colorfilters`, `dfilters` and optionally `preferences` from `rsplib/wireshark` to your WIRESHARK configuration directory (usually: `~/.wireshark` or `/root/.wireshark`). The output should look similar to the example in figure 3.

3.2 Keeping an Overview of the Scenario

Before starting further components, it is useful no ensure not loosing the overview of running components. For this reason, the RSPLIB components support the Component Status Protocol[2] (CSP), which provides regular status information over UDP to a monitoring component. First, choose another PC to provide the monitoring output and start the monitor program:

```
cspmonitor
```

Now, restart the other components with two additional parameters, i.e.:

```
registrar -cspserver=<Monitor Address> -cspinterval=500
server -fractal -cspserver=<Monitor Address> -cspinterval=500
fractalpooluser -cspserver=<Monitor Address> -cspinterval=500
```

The interval gives the inter-report time in milliseconds. 500ms should be useful for our scenario.

Now, you should be able to keep an overview of your RSerPool scenario. Start a second PR and some additional PEs and PUs to test your setup.

[2]CSP is not part of RSerPool itself, but a quite useful tool provided by RSPLIB.

3.3 Automatic Configuration

Have a look at the ASAP and ENRP traffic over UDP. You can apply the filter rule “(asap||enrp)&&udp” in WIRESHARK to view exactly this kind of packets.

Question 8:
Which ASAP message type can you see here?

Solution: You can see ASAP Server Announces.

Example ASAP Server Announce:

```
Internet Protocol, Src: 132.252.156.18 (132.252.156.18),
                   Dst: 239.0.0.50 (239.0.0.50)
User Datagram Protocol, Src Port: 15893 (15893), Dst Port: 3863 (3863)
Aggregate Server Access Protocol
    Type: ASAP Server Announce (10)
    Flags: 0x00
    Length: 8
    Server identifier: 0x36acceb7
```

Question 9:
What can you say about the destination address of these ASAP messages?

Solution: It is a multicast address.

Question 10:
Can you imagine the reason why SCTP cannot be used for these messages?

Solution: SCTP is a unicast protocol, so multicast cannot be used with SCTP. UDP is the only transport protocol which can be used for this purpose (at least, it is the only multicast-capable protocol supported by “normal” systems).

Question 11:
Where can you find the IP address and SCTP port number of the PR sending these messages?

Solution: If there are no addresses provided in the message body itself (which is optionally possible, see [SXST08a, SXST08b]), the IP address is the sender’s IP address and the SCTP port number is equal to the sender’s UDP port number.

Question 12:
Can you also find out the sender’s PR ID?

Solution: Yes, it is provided within the ASAP Server Announce message.

Question 13:
Have a look at the ENRP Presence messages. What is the difference to the ASAP Server Announces?

Solution: The ENRP Presence also contains the sender and receiver PR ID, the sender’s checksum of its owned PE identities and a Server Information Parameter. This Server Information Parameter contains a description of the ENRP endpoint: its SCTP port number and all IP addresses.

```
Internet Protocol, Src: 132.252.156.19 (132.252.156.19),
                   Dst: 132.252.156.18 (132.252.156.18)
Stream Control Transmission Protocol, Src Port: 14736 (14736),
                                      Dst Port: 57129 (57129)
Endpoint Handlespace Redundancy Protocol
    Type: ENRP Presence (1)
    Flags: 0x01
        .... ...1 = R bit: Reply required
    Length: 160
    Sender server’s ID: 0x278ca457
    Receiver server’s ID: 0x36acceb7
    PE checksum (0xffff)
        Parameter Type: PE checksum (0x000f)
        Parameter length: 6
        PE checksum: 0xffff
        Padding: 0000
    Server Information
        Parameter Type: Server Information (0x000b)
        Parameter length: 140
        Server identifier: 0x278ca457
        SCTP transport address
            Parameter Type: SCTP transport address (0x0004)
            Parameter length: 132
            Port: 14736
            Transport use: Data only (0)
            IPV4 address (127.0.0.1)
                Parameter Type: IPV4 address (0x0001)
                Parameter length: 8
                IP Version 4 address: 127.0.0.1 (127.0.0.1)
            IPV6 address (::1)
                Parameter Type: IPV6 address (0x0002)
                Parameter length: 20
                IP Version 6 address: ::1 (::1)
            IPV4 address (132.252.156.149)
                Parameter Type: IPV4 address (0x0001)
                Parameter length: 8
                IP Version 4 address: 132.252.156.149 (132.252.156.149)
            IPV6 address (2001:638:501:4ef4:201:2ff:fef1:8409)
                Parameter Type: IPV6 address (0x0002)
                Parameter length: 20
                IP Version 6 address: 2001:638:501:4ef4:201:2ff:fef1:8409
            IPV6 address (fe80::201:2ff:fef1:8409)
                Parameter Type: IPV6 address (0x0002)
                Parameter length: 20
                IP Version 6 address: fe80::201:2ff:fef1:8409
            IPV4 address (132.252.156.19)
                Parameter Type: IPV4 address (0x0001)
                Parameter length: 8
                IP Version 4 address: 132.252.156.19 (132.252.156.19)
            IPV6 address (2001:638:501:4ef3:211:43ff:febb:cee0)
                Parameter Type: IPV6 address (0x0002)
                Parameter length: 20
                IP Version 6 address: 2001:638:501:4ef3:211:43ff:febb:cee0
            IPV6 address (fe80::211:43ff:febb:cee0)
                Parameter Type: IPV6 address (0x0002)
                Parameter length: 20
                IP Version 6 address: fe80::211:43ff:febb:cee0
```

Question 14:

Now, have a look at the messages sent over SCTP by applying the filter "(asap||enrp)&&sctp". Can you imagine why you cannot see ASAP Server Announces but only ENRP Presences via SCTP?

Solution: ASAP Server Announces are only used to announce the existence of a PR to PEs and PUs. If the PR is gone, PEs and PUs can detect the failure by missing answers to their requests.

For ENRP, a SCTP association will be established to all other PRs. There must be a keep-alive mechanism among the PRs to detect failures – since in the usual case there is no request-response communication. This keep-alive mechanism is realized by ENRP Presences over the SCTP associations.

3.4 Pool Management

Now, have a look at the WIRESHARK output at a PE's PR-H. In particular, observe the ASAP Registration and ENRP Update messages.

Question 15:

Which two parameters can you observe in ASAP Registration and ENRP Update messages?

Solution: You can see a Pool Handle parameter (containing the PH of the pool) and a Pool Element Parameter (containing all information about the PE to register). They are described in [SXST08b].

Example ASAP Registration:

```
Internet Protocol, Src: 132.252.156.19 (132.252.156.19),
                   Dst: 132.252.156.18 (132.252.156.18)
Stream Control Transmission Protocol, Src Port: 33125 (33125),
                                      Dst Port: 15893 (15893)
Aggregate Server Access Protocol
    Type: ASAP Registration (1)
    Flags: 0x00
    Length: 184
    Pool handle
        Parameter Type: Pool handle (0x0009)
        Parameter length: 24
        Pool handle: 4672616374616C47656E657261746F72506F6F6C
    Pool element
        Parameter Type: Pool element (0x000a)
        Parameter length: 156
        PE identifier: 0x336e014d
        Home ENRP server identifier: 0x00000000
        Registration life: 90000
        SCTP transport address
            Parameter Type: SCTP transport address (0x0004)
            Parameter length: 132
            Port: 57378
            Transport use: Data plus control (1)
            IPV4 address (127.0.0.1)
                Parameter Type: IPV4 address (0x0001)
                Parameter length: 8
                IP Version 4 address: 127.0.0.1 (127.0.0.1)
            IPV6 address (::1)
                Parameter Type: IPV6 address (0x0002)
                Parameter length: 20
                IP Version 6 address: ::1 (::1)
            IPV4 address (132.252.156.149)
```

```
                Parameter Type: IPV4 address (0x0001)
                Parameter length: 8
                IP Version 4 address: 132.252.156.149 (132.252.156.149)
            IPV6 address (2001:638:501:4ef4:201:2ff:fef1:8409)
                Parameter Type: IPV6 address (0x0002)
                Parameter length: 20
                IP Version 6 address: 2001:638:501:4ef4:201:2ff:fef1:8409
            IPV6 address (fe80::201:2ff:fef1:8409)
                Parameter Type: IPV6 address (0x0002)
                Parameter length: 20
                IP Version 6 address: fe80::201:2ff:fef1:8409
            IPV4 address (132.252.156.19)
                Parameter Type: IPV4 address (0x0001)
                Parameter length: 8
                IP Version 4 address: 132.252.156.19 (132.252.156.19)
            IPV6 address (2001:638:501:4ef3:211:43ff:febb:cee0)
                Parameter Type: IPV6 address (0x0002)
                Parameter length: 20
                IP Version 6 address: 2001:638:501:4ef3:211:43ff:febb:cee0
            IPV6 address (fe80::211:43ff:febb:cee0)
                Parameter Type: IPV6 address (0x0002)
                Parameter length: 20
                IP Version 6 address: fe80::211:43ff:febb:cee0
        Pool member selection policy
            Parameter Type: Pool member selection policy (0x0008)
            Parameter length: 8
            Policy type: Round robin (1)
```

Example ENRP Update:

```
Internet Protocol, Src: 132.252.156.18 (132.252.156.18),
                   Dst: 132.252.156.19 (132.252.156.19)
Stream Control Transmission Protocol, Src Port: 57129 (57129),
                                      Dst Port: 26843 (26843)
Endpoint Handlespace Redundancy Protocol
    Type: ENRP Handle Update (4)
    Flags: 0x00
    Length: 192
    Sender server’s ID: 0x36acceb7
    Receiver server’s ID: 0x081c671f
    Update action: Add pool element (0)
    Reserved: 0x0000
    Pool handle
        Parameter Type: Pool handle (0x0009)
        Parameter length: 24
        Pool handle: 4672616374616C47656E657261746F72506F6F6C
    Pool element
        Parameter Type: Pool element (0x000a)
        Parameter length: 152
        PE identifier: 0x336e014d
        Home ENRP server identifier: 0x36acceb7
        Registration life: 90000
        SCTP transport address
            Parameter Type: SCTP transport address (0x0004)
            Parameter length: 64
            Port: 57378
            Transport use: Data plus control (1)
            IPV4 address (132.252.156.149)
```

```
                Parameter Type: IPV4 address (0x0001)
                Parameter length: 8
                IP Version 4 address: 132.252.156.149 (132.252.156.149)
            IPV6 address (2001:638:501:4ef4:201:2ff:fef1:8409)
                Parameter Type: IPV6 address (0x0002)
                Parameter length: 20
                IP Version 6 address: 2001:638:501:4ef4:201:2ff:fef1:8409
            IPV4 address (132.252.156.19)
                Parameter Type: IPV4 address (0x0001)
                Parameter length: 8
                IP Version 4 address: 132.252.156.19 (132.252.156.19)
            IPV6 address (2001:638:501:4ef3:211:43ff:febb:cee0)
                Parameter Type: IPV6 address (0x0002)
                Parameter length: 20
                IP Version 6 address: 2001:638:501:4ef3:211:43ff:febb:cee0
        Pool member selection policy
            Parameter Type: Pool member selection policy (0x0008)
            Parameter length: 8
            Policy type: Round robin (1)
        SCTP transport address
            Parameter Type: SCTP transport address (0x0004)
            Parameter length: 64
            Port: 33125
            Transport use: Data only (0)
            IPV4 address (132.252.156.19)
                Parameter Type: IPV4 address (0x0001)
                Parameter length: 8
                IP Version 4 address: 132.252.156.19 (132.252.156.19)
            IPV4 address (132.252.156.149)
                Parameter Type: IPV4 address (0x0001)
                Parameter length: 8
                IP Version 4 address: 132.252.156.149 (132.252.156.149)
            IPV6 address (2001:638:501:4ef4:201:2ff:fef1:8409)
                Parameter Type: IPV6 address (0x0002)
                Parameter length: 20
                IP Version 6 address: 2001:638:501:4ef4:201:2ff:fef1:8409
            IPV6 address (2001:638:501:4ef3:211:43ff:febb:cee0)
                Parameter Type: IPV6 address (0x0002)
                Parameter length: 20
                IP Version 6 address: 2001:638:501:4ef3:211:43ff:febb:cee0
```

Question 16:
Which information about the PE to register can be found in an ASAP Registration message's parameters?

Solution: You can find the following information:

- PE ID,
- Home-PR ID,
- Registration lifetime,
- Endpoint addresses of the *application* socket and
- Pool Member Selection Policy paramter (describing the PE's selection policy).

Question 17:

Does the ENRP Update contain exactly the same information about the PE as the ASAP Registration message? Why or why not?

Solution: No, the information in the ENRP Update is different. The PR-H may filter out addresses – which are either not reachable or have a too-limited scope (default for RSPLIB: scope must be at least site-local). Furthermore, after the policy parameter, there is the specification of the ASAP endpoint addresses. They are directly filled in at the PR-H (known from the association to the PE), so it is *not* necessary to specify them in the ASAP Registration message.

Question 18:

What is the ASAP response to an ASAP Registration message? Can you imagine why there is no Pool Element Parameter included in this type of message?

Solution: The response is an ASAP Registration Response. It is already sufficient to only specify the PE ID instead of giving a full description of the PE: the PE itself already has this information – and only the ID is needed to differentiate among multiple simultaneous registrations.

Example ASAP Registration Response:

```
Internet Protocol, Src: 132.252.156.18 (132.252.156.18),
                   Dst: 132.252.156.19 (132.252.156.19)
Stream Control Transmission Protocol, Src Port: 15893 (15893),
                                      Dst Port: 33125 (33125)
Aggregate Server Access Protocol
    Type: ASAP Registration Response (3)
    Flags: 0x00
        .... ...0 = R bit: Accepted
    Length: 36
    Pool handle
        Parameter Type: Pool handle (0x0009)
        Parameter length: 24
        Pool handle: 4672616374616C47656E657261746F72506F6F6C
    Pool Element identifier (0x336e014d)
        Parameter Type: Pool Element identifier (0x000e)
        Parameter length: 8
        PE identifier: 0x336e014d
```

Question 19:

Now, try registering a new PE into the existing pool, using `-policy=LeastUsed` as additional parameter. This parameter sets the pool policy (see also [DT08]) to Least Used (LU). Why is the registration not successful? Also have a look at the response message!

Solution: The ASAP Registration Response now also contains an Error Parameter (see [SXST08b] for the definition), giving the reason for the rejected registration: an incompatible pool policy. Each pool has exactly one policy! Since the pool is already existing, all new PEs must have the policy of the existing PEs.

Question 20:

Stop all PEs of the pool. After the pool is completely empty, restart them with the Least Used policy. Does the pool now have the desired policy (check the WIRESHARK output)? Why or why not?

Solution: Since the pool was empty (i.e. it was not existing in the handlespace anymore), it got the policy of the first PE registering into the pool: LU. Therefore, all other PEs registering to the pool and using LU were also able to join, too.

Question 21:

Now, deregister one of the PEs and monitor the message sequence. Which message types do you see? What is the difference in the ENRP Update message(s)?

Solution: A deregistration is performed by an ASAP Deregistration. It is confirmed by an ASAP Deregistration Response. The ENRP Update is the same as for the registration, except for the "Update Action" field: it is now "Remove pool element (1)" instead of "Add pool element (0)".

Note that the ENRP Update contains a full Pool Element Parameter, not only the PE ID as for the ASAP Deregistration Response. Reason: just a design flaw, but no technical reason.

Example ASAP Deregistration:

```
Internet Protocol, Src: 132.252.156.19 (132.252.156.19),
                   Dst: 132.252.156.18 (132.252.156.18)
Stream Control Transmission Protocol, Src Port: 33125 (33125),
                                      Dst Port: 15893 (15893)
Aggregate Server Access Protocol
    Type: ASAP Deregistration (2)
    Flags: 0x00
    Length: 36
    Pool handle
        Parameter Type: Pool handle (0x0009)
        Parameter length: 24
        Pool handle: 4672616374616C47656E657261746F72506F6F6C
    Pool Element identifier (0x336e014d)
        Parameter Type: Pool Element identifier (0x000e)
        Parameter length: 8
        PE identifier: 0x336e014d
```

Example ENRP Update:

```
Internet Protocol, Src: 132.252.156.18 (132.252.156.18),
                   Dst: 132.252.156.19 (132.252.156.19)
Stream Control Transmission Protocol, Src Port: 57129 (57129),
                                      Dst Port: 26843 (26843)
Endpoint Handlespace Redundancy Protocol
    Type: ENRP Handle Update (4)
    Flags: 0x00
    Length: 192
    Sender server’s ID: 0x36acceb7
    Receiver server’s ID: 0x081c671f
    Update action: Delete pool element (1)
    Reserved: 0x0000
    Pool handle
        Parameter Type: Pool handle (0x0009)
        Parameter length: 24
        Pool handle: 4672616374616C47656E657261746F72506F6F6C
    Pool element
        Parameter Type: Pool element (0x000a)
        Parameter length: 152
        PE identifier: 0x336e014d
        Home ENRP server identifier: 0x36acceb7
        Registration life: 90000
        SCTP transport address
            Parameter Type: SCTP transport address (0x0004)
            Parameter length: 64
            Port: 57378
            Transport use: Data plus control (1)
            IPV4 address (132.252.156.149)
                Parameter Type: IPV4 address (0x0001)
                Parameter length: 8
                IP Version 4 address: 132.252.156.149 (132.252.156.149)
            IPV6 address (2001:638:501:4ef4:201:2ff:fef1:8409)
                Parameter Type: IPV6 address (0x0002)
                Parameter length: 20
                IP Version 6 address: 2001:638:501:4ef4:201:2ff:fef1:8409
            IPV4 address (132.252.156.19)
                Parameter Type: IPV4 address (0x0001)
                Parameter length: 8
                IP Version 4 address: 132.252.156.19 (132.252.156.19)
            IPV6 address (2001:638:501:4ef3:211:43ff:febb:cee0)
                Parameter Type: IPV6 address (0x0002)
                Parameter length: 20
                IP Version 6 address: 2001:638:501:4ef3:211:43ff:febb:cee0
        Pool member selection policy
            Parameter Type: Pool member selection policy (0x0008)
            Parameter length: 8
            Policy type: Round robin (1)
        SCTP transport address
            Parameter Type: SCTP transport address (0x0004)
            Parameter length: 64
            Port: 33125
            Transport use: Data only (0)
            IPV4 address (132.252.156.19)
                Parameter Type: IPV4 address (0x0001)
                Parameter length: 8
                IP Version 4 address: 132.252.156.19 (132.252.156.19)
            IPV4 address (132.252.156.149)
                Parameter Type: IPV4 address (0x0001)
                Parameter length: 8
                IP Version 4 address: 132.252.156.149 (132.252.156.149)
            IPV6 address (2001:638:501:4ef4:201:2ff:fef1:8409)
                Parameter Type: IPV6 address (0x0002)
                Parameter length: 20
                IP Version 6 address: 2001:638:501:4ef4:201:2ff:fef1:8409
            IPV6 address (2001:638:501:4ef3:211:43ff:febb:cee0)
                Parameter Type: IPV6 address (0x0002)
                Parameter length: 20
                IP Version 6 address: 2001:638:501:4ef3:211:43ff:febb:cee0
```

Example ASAP Deregistration Response:

```
Internet Protocol, Src: 132.252.156.18 (132.252.156.18),
                   Dst: 132.252.156.19 (132.252.156.19)
Stream Control Transmission Protocol, Src Port: 15893 (15893),
                                      Dst Port: 33125 (33125)
Aggregate Server Access Protocol
    Type: ASAP Deregistration Response (4)
    Flags: 0x00
    Length: 36
    Pool handle
        Parameter Type: Pool handle (0x0009)
        Parameter length: 24
        Pool handle: 4672616374616C47656E657261746F72506F6F6C
    Pool Element identifier (0x336e014d)
        Parameter Type: Pool Element identifier (0x000e)
        Parameter length: 8
        PE identifier: 0x336e014d
```

3.5 Server Selection

After the pool management, we will now have a look at the PU side. Observe the message flow at one of the lab PCs running the PU process.

Question 22:
Which type of ASAP message is used to request a server selection? What are its contents?

Solution: An ASAP Handle Resolution is used to request the server selection. It contains the PH of the pool.

Example ASAP Handle Resolution:

```
Internet Protocol, Src: 132.252.156.149 (132.252.156.149),
                   Dst: 132.252.156.149 (132.252.156.149)
Stream Control Transmission Protocol, Src Port: 56159 (56159),
                                      Dst Port: 12653 (12653)
Aggregate Server Access Protocol
    Type: ASAP Handle Resolution (5)
    Flags: 0x00
    Length: 36
    Pool handle
        Parameter Type: Pool handle (0x0009)
        Parameter length: 24
        Pool handle: 4672616374616C47656E657261746F72506F6F6C
    Unknown parameter (type 32831 and 4 bytes value)
        Parameter Type: Unknown (0x803f)
        Parameter length: 8
        Parameter value: 00000001
```

Note: the “unknown parameter” is the handle resolution option, defined in [Dre10b].

Question 23:
What is the response?

Solution: The response is an ASAP Handle Resolution Response. It contains the following information:

- The PH of the pool,
- The pool policy and
- A list of PE identities (in form of Pool Element Parameters).

Example ASAP Handle Resolution Response:

```
Internet Protocol, Src: 132.252.156.149 (132.252.156.149),
                   Dst: 132.252.156.149 (132.252.156.149)
Stream Control Transmission Protocol, Src Port: 12653 (12653),
                                      Dst Port: 56159 (56159)
Aggregate Server Access Protocol
    Type: ASAP Handle Resolution Response (6)
    Flags: 0x00
    Length: 124
    Pool handle
        Parameter Type: Pool handle (0x0009)
        Parameter length: 24
        Pool handle: 4672616374616C47656E657261746F72506F6F6C
    Pool member selection policy
        Parameter Type: Pool member selection policy (0x0008)
        Parameter length: 8
        Policy type: Round robin (1)
    Pool element
        Parameter Type: Pool element (0x000a)
        Parameter length: 88
        PE identifier: 0x7faab6d8
        Home ENRP server identifier: 0x36acceb7
        Registration life: 90000
        SCTP transport address
            Parameter Type: SCTP transport address (0x0004)
            Parameter length: 64
            Port: 51074
            Transport use: Data plus control (1)
            IPV4 address (132.252.156.18)
                Parameter Type: IPV4 address (0x0001)
                Parameter length: 8
                IP Version 4 address: 132.252.156.18 (132.252.156.18)
            IPV6 address (2001:638:501:4ef3:211:43ff:febb:9f62)
                Parameter Type: IPV6 address (0x0002)
                Parameter length: 20
                IP Version 6 address: 2001:638:501:4ef3:211:43ff:febb:9f62
            IPV4 address (132.252.156.148)
                Parameter Type: IPV4 address (0x0001)
                Parameter length: 8
                IP Version 4 address: 132.252.156.148 (132.252.156.148)
            IPV6 address (2001:638:501:4ef4:204:76ff:fe1a:50b6)
                Parameter Type: IPV6 address (0x0002)
                Parameter length: 20
                IP Version 6 address: 2001:638:501:4ef4:204:76ff:fe1a:50b6
        Pool member selection policy
            Parameter Type: Pool member selection policy (0x0008)
            Parameter length: 8
            Policy type: Round robin (1)
```

Question 24:

What happens upon request for a non-existing (equal to “empty”) pool? Try to run the PU for a non-existing PH using `-poolhandle=<Name>`!

Solution: The ASAP Handle Resolution Response contains – instead of the PE identities – an appropriate error parameter: “unknown pool handle”.

3.6 Session Layer

After the basic functionalities, we will have a look at the Session Layer functionality of RSerPool. On a PC running a PU, filter for all SCTP traffic and observe the protocol flow of an image calculation using the Fractal Generator Service. Also, during calculation, shut down the PE the PU is connected to in order to observe the failover. Alternatively, you can turn on the “failure mode” for the PEs using the parameter `-fgpfailureafter=<Packets>`, which turns on breaking the association after the given number of FGP Data packets.

Question 25:
How is a session failover handled for the Fractal Generator Service?

Solution: It uses client-based state sharing: the PE sends its state as state cookie to the PU using ASAP Cookie messages. The PU stores the latest cookie. On failover, it sends the stored cookie to the new PE using an ASAP Cookie Echo message.

You can tell the PU program to use multiple sessions simultaneously, using the PU's command line parameter `-threads=<Sessions>`. Each PE by default accepts four sessions simultaneously. You can change this behaviour by `-fgpmaxthreads=<Sessions>`.

Question 26:
What do you observe on WIRESHARK when you start more sessions than there is PE capacity to process them? Is this a problem?

Solution: A fully-loaded PE rejects any further session by simply performing a shutdown of the association. The PU tries again – by performing another handle resolution and contacting the newly chosen PE – after a short timeout. So, there is no real problem. Some time later, a usable PE will be found and the session can be processed.

Of course, the administrator of an RSerPool setup should dimension the system appropriately – so that in the usual case there is sufficient capacity to handle all requests without rejections.

3.7 ENRP Handlespace Synchronization

In the following, we are going to make some tests with the ENRP protocol. First, we would like to observe the PR initialization. Therefore, run WIRESHARK and start a new PR. The new PR will detect the existence of the other PRs by the ENRP Presences over UDP multicast and establish associations.

Question 27:
How is the initial sequence of the ENRP messages over SCTP?

Solution: The new PR asks for another PR's Peer Table (using ENRP List Request) and handlespace copy (ENRP Handle Table Request). Usually, these requests are bundled within a single SCTP packet.

Example ENRP List Request/ENRP Handle Table Request:

```
Internet Protocol, Src: 132.252.156.19 (132.252.156.19),
                   Dst: 132.252.156.18 (132.252.156.18)
Stream Control Transmission Protocol, Src Port: 14736 (14736),
                                      Dst Port: 57129 (57129)
Endpoint Handlespace Redundancy Protocol
    Type: ENRP List Request (5)
    Flags: 0x00
    Length: 12
    Sender server's ID: 0x278ca457
    Receiver server's ID: 0x36acceb7
Stream Control Transmission Protocol
Endpoint Handlespace Redundancy Protocol
    Type: ENRP Handle Table Request (2)
    Flags: 0x00
        .... ...0 = W bit: Information for all PEs
    Length: 12
    Sender server's ID: 0x278ca457
    Receiver server's ID: 0x36acceb7
```

Question 28:

Which type of message contains the Peer List?

Solution: The Peer List is contained within the ENRP List Response.

Example ENRP List Response:

```
Internet Protocol, Src: 132.252.156.18 (132.252.156.18),
                   Dst: 132.252.156.19 (132.252.156.19)
Stream Control Transmission Protocol, Src Port: 57129 (57129),
                                      Dst Port: 14736 (14736)
Endpoint Handlespace Redundancy Protocol
    Type: ENRP List Response (6)
    Flags: 0x00
        .... ...0 = R bit: Accepted
    Length: 84
    Sender server’s ID: 0x36acceb7
    Receiver server’s ID: 0x278ca457
    Server Information
        Parameter Type: Server Information (0x000b)
        Parameter length: 72
        Server identifier: 0x278ca457
        SCTP transport address
            Parameter Type: SCTP transport address (0x0004)
            Parameter length: 64
            Port: 14736
            Transport use: Data only (0)
            IPV4 address (132.252.156.149)
                Parameter Type: IPV4 address (0x0001)
                Parameter length: 8
                IP Version 4 address: 132.252.156.149 (132.252.156.149)
            IPV6 address (2001:638:501:4ef4:201:2ff:fef1:8409)
                Parameter Type: IPV6 address (0x0002)
                Parameter length: 20
                IP Version 6 address: 2001:638:501:4ef4:201:2ff:fef1:8409
            IPV4 address (132.252.156.19)
                Parameter Type: IPV4 address (0x0001)
                Parameter length: 8
                IP Version 4 address: 132.252.156.19 (132.252.156.19)
            IPV6 address (2001:638:501:4ef3:211:43ff:febb:cee0)
                Parameter Type: IPV6 address (0x0002)
                Parameter length: 20
                IP Version 6 address: 2001:638:501:4ef3:211:43ff:febb:cee0
```

Question 29:

What type of information is transported by the ENRP Handle Table Response? Can you imagine the meaning of the “M” bit in the flags field?

Solution: The ENRP Handle Table Response contains a copy of the handlespace (or a subset of it). If the “M” (i.e. “More”) bit is set, there is more handlespace data available than possible to fit in the ENRP Handle Table Response. A subsequent ENRP Handle Table Request can retrieve the next block.

Example ENRP Handle Table Response:

```
Internet Protocol, Src: 132.252.156.18 (132.252.156.18),
                   Dst: 132.252.156.19 (132.252.156.19)
Stream Control Transmission Protocol, Src Port: 57129 (57129),
                                      Dst Port: 14736 (14736)
Endpoint Handlespace Redundancy Protocol
    Type: ENRP Handle Table Response (3)
    Flags: 0x00
        .... ..0. = M bit: All information included
        .... ...0 = R bit: Accepted
    Length: 368
    Sender server’s ID: 0x36acceb7
    Receiver server’s ID: 0x278ca457
    Pool handle
        Parameter Type: Pool handle (0x0009)
        Parameter length: 17
        Pool handle: 536372697074696E67506F6F6C
        Padding: 000000
    Pool element
        Parameter Type: Pool element (0x000a)
        Parameter length: 160
        PE identifier: 0x14f2d0d7
        Home ENRP server identifier: 0x36acceb7
        Registration life: 90000
        SCTP transport address
            Parameter Type: SCTP transport address (0x0004)
            Parameter length: 64
            Port: 47597
            Transport use: Data plus control (1)
            IPV4 address (132.252.156.18)
                Parameter Type: IPV4 address (0x0001)
                Parameter length: 8
                IP Version 4 address: 132.252.156.18 (132.252.156.18)
            IPV6 address (2001:638:501:4ef3:211:43ff:febb:9f62)
                Parameter Type: IPV6 address (0x0002)
                Parameter length: 20
                IP Version 6 address: 2001:638:501:4ef3:211:43ff:febb:9f62
            IPV4 address (132.252.156.148)
                Parameter Type: IPV4 address (0x0001)
                Parameter length: 8
                IP Version 4 address: 132.252.156.148 (132.252.156.148)
            IPV6 address (2001:638:501:4ef4:204:76ff:fe1a:50b6)
                Parameter Type: IPV6 address (0x0002)
                Parameter length: 20
                IP Version 6 address: 2001:638:501:4ef4:204:76ff:fe1a:50b6
        Pool member selection policy
            Parameter Type: Pool member selection policy (0x0008)
            Parameter length: 16
            Policy type: Least used with degradation (1073741826)
            Policy load: 0
            Policy degradation: 4294967295
        SCTP transport address
            Parameter Type: SCTP transport address (0x0004)
            Parameter length: 64
            Port: 53161
            Transport use: Data only (0)
            IPV4 address (132.252.156.18)
                Parameter Type: IPV4 address (0x0001)
                Parameter length: 8
                IP Version 4 address: 132.252.156.18 (132.252.156.18)
            IPV6 address (2001:638:501:4ef3:211:43ff:febb:9f62)
                Parameter Type: IPV6 address (0x0002)
                Parameter length: 20
                IP Version 6 address: 2001:638:501:4ef3:211:43ff:febb:9f62
            IPV4 address (132.252.156.148)
                Parameter Type: IPV4 address (0x0001)
                Parameter length: 8
                IP Version 4 address: 132.252.156.148 (132.252.156.148)
            IPV6 address (2001:638:501:4ef4:204:76ff:fe1a:50b6)
                Parameter Type: IPV6 address (0x0002)
                Parameter length: 20
                IP Version 6 address: 2001:638:501:4ef4:204:76ff:fe1a:50b6
    Pool handle
        Parameter Type: Pool handle (0x0009)
        Parameter length: 24
        Pool handle: 4672616374616C47656E657261746F72506F6F6C
    Pool element
        Parameter Type: Pool element (0x000a)
        Parameter length: 152
        PE identifier: 0x7faab6d8
        Home ENRP server identifier: 0x36acceb7
        Registration life: 90000
        SCTP transport address
            Parameter Type: SCTP transport address (0x0004)
            Parameter length: 64
            Port: 51074
            Transport use: Data plus control (1)
            IPV4 address (132.252.156.18)
                Parameter Type: IPV4 address (0x0001)
                Parameter length: 8
                IP Version 4 address: 132.252.156.18 (132.252.156.18)
            IPV6 address (2001:638:501:4ef3:211:43ff:febb:9f62)
                Parameter Type: IPV6 address (0x0002)
                Parameter length: 20
                IP Version 6 address: 2001:638:501:4ef3:211:43ff:febb:9f62
            IPV4 address (132.252.156.148)
                Parameter Type: IPV4 address (0x0001)
                Parameter length: 8
                IP Version 4 address: 132.252.156.148 (132.252.156.148)
            IPV6 address (2001:638:501:4ef4:204:76ff:fe1a:50b6)
                Parameter Type: IPV6 address (0x0002)
                Parameter length: 20
                IP Version 6 address: 2001:638:501:4ef4:204:76ff:fe1a:50b6
        Pool member selection policy
            Parameter Type: Pool member selection policy (0x0008)
            Parameter length: 8
            Policy type: Round robin (1)
        SCTP transport address
            Parameter Type: SCTP transport address (0x0004)
            Parameter length: 64
            Port: 53324
            Transport use: Data only (0)
            IPV4 address (132.252.156.18)
                Parameter Type: IPV4 address (0x0001)
                Parameter length: 8
                IP Version 4 address: 132.252.156.18 (132.252.156.18)
            IPV6 address (2001:638:501:4ef3:211:43ff:febb:9f62)
                Parameter Type: IPV6 address (0x0002)
                Parameter length: 20
                IP Version 6 address: 2001:638:501:4ef3:211:43ff:febb:9f62
```

```
             IPV4 address (132.252.156.148)
                 Parameter Type: IPV4 address (0x0001)
                 Parameter length: 8
                 IP Version 4 address: 132.252.156.148 (132.252.156.148)
             IPV6 address (2001:638:501:4ef4:204:76ff:fe1a:50b6)
                 Parameter Type: IPV6 address (0x0002)
                 Parameter length: 20
                 IP Version 6 address: 2001:638:501:4ef4:204:76ff:fe1a:50b6
```

4 Application of Reliable Server Pooling

In the last part of this exercise, we have a look at an RSerPool application: the RSPLIB Scripting Service (SS). This service can e.g. be used to process OMNeT+ [Var09] simulation runs (see [DR08a, DZR09] for details). In the following, we utilize it to perform ray-tracing image calculations using the ray-tracer POV-Ray [POV10].

4.1 The RSPLIB Scripting Service

The Scripting Service works as follows: the Scripting Service PE accepts a Tar/GZip file ("work package"), which is unpacked into a temporary directory. Within the archive, there is a script named `ssrun`. `ssrun` is executed with the name of an output archive as its first argument. The `ssrun` script may do something useful with the data provided within the archive (e.g. processing a simulation run) and finally write a Tar/GZip archive as output ("results package") – using the provided output archive name. If `ssrun` returns 0, the processing has been successful. Otherwise, there has been a problem and a failover should be performed.

A user simply has to provide the work package to the Scripting Service PU and eventually gets back the results package. Using a pool of multiple PEs and starting several PU sessions in parallel, RSerPool can be used for efficient load balancing and pool management (see also [Dre10a]).

4.2 The POV-RAY Ray-Tracer

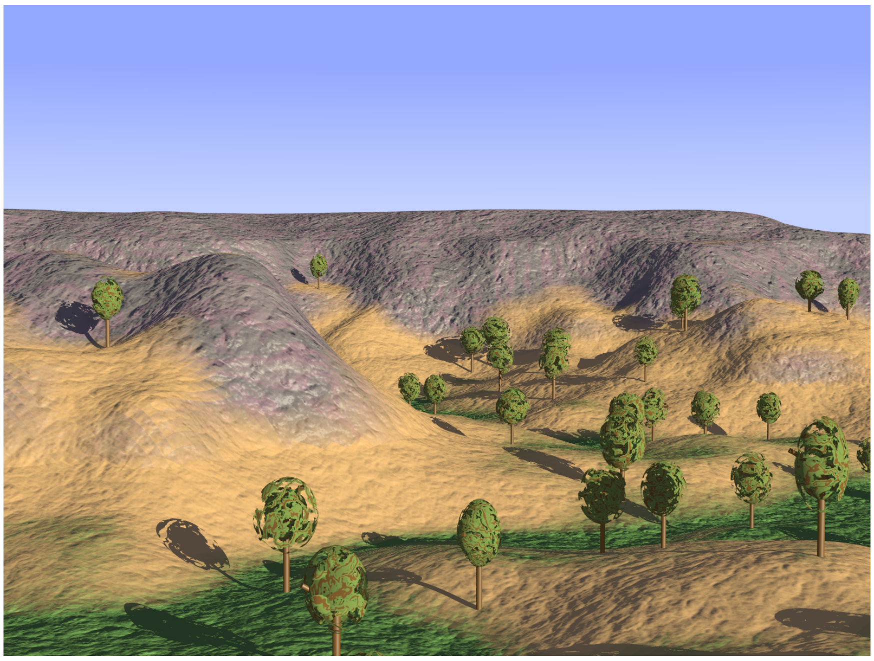


Figure 4: The POV-RAY Example "`landscape.pov`"

The POV-RAY [POV10] ray-tracer provides the command-line program `povray`, which takes an input file (`.pov`) and calculates the resulting image. For us, the following parameters are relevant:

-w Image width (e.g. `-w1024`),

-h Image height (e.g. `-h768`),

+a Use anti-alias (e.g. `+a0.3`),

+FN8 Use PNG output format (8 bits per colour, i.e. 24 bits for RGB),

+I Specifies input file name (e.g. `+Iinput.pov`),

+O Specifies output file name (e.g. `+Ooutput.png`),

-D Turns off X11 preview.

There are various other options described in the manual page of `povray`. The output of the example `landscape.pov` is shown in figure 4.

The directory `/usr/share/doc/povray/examples/advanced` contains a set of advanced example `.pov` files (GZip-compressed). For convenience reasons, copy them to a new directory and unpack them:

```
mkdir raytracing-images
cd raytracing-images
find /usr/share/doc/povray/examples/advanced | xargs -i§ cp § .
gzip -d *.gz
```

Now, we would like to calculate wallpapers (e.g. 1024x768) of all `.pov` files in the new directory. Of course, we would like to utilize the computation power of the complete lab pool for this task.

4.3 Applying the Scripting Service for POV-RAY

We now write a script `povray-distribute`, which performs the workload distribution task. This new script takes image width and height as well as a `.pov` file name as arguments. First, the arguments have to be processed:

```
#!/bin/bash
# ====== Get arguments ==============================================
if [ $# -lt 3 ] ; then
   echo >&2 "ERROR: Usage $0 [Width] [Height] [Input POV]"
   exit 1
fi
WIDTH=$1
HEIGHT=$2
INPUT=$3
OUTPUT="`echo $INPUT | sed -e "s/.pov/-$WIDTH-$HEIGHT.png"/g`"
```

For the new image calculation task, we create a temporary directory and store the input `.pov` file as `input.pov` into this directory. Include files (`*.inc`) are also copied.

```
# ====== Create temporary directory =================================
TEMPDIR="temp-$INPUT-$WIDTH-$HEIGHT"
umask 077
rm -rf $TEMPDIR
mkdir $TEMPDIR
cp $INPUT $TEMPDIR/input.pov
find -name "*.inc" | xargs --no-run-if-empty -n1 -i§ cp § $TEMPDIR
```

Furthermore, we need a `ssrun` file, which we write using `echo` commands. `ssrun` is also stored into the temporary directory. The `ssrun` script will call `povray` on the input file, with the appropriate parameters. The resulting output image will be called `image.png`. The variable `SUCCESS` contains the result of `ssrun`. It is set to 1 (i.e. “failed”) if something goes wrong. Furthermore, the text output of `povray` is written to `output.txt` for debugging in the case of something going wrong.

```
# ====== Write ssrun script =========================================
(
   echo "#!/bin/sh"
   echo "OUTPUT_ARCHIVE=\$1"
   echo "SUCCESS=1"
   echo -n "povray -w$WIDTH -h$HEIGHT +a0.3 -D +FN8 +Ooutput.png "
   echo    "+Iinput.pov >output.txt 2>&1 || SUCCESS=0"
   echo "tar czvf \$OUTPUT_ARCHIVE output.png output.txt || SUCCESS=0"
   echo "exit $SUCCESS"
) >"$TEMPDIR/ssrun"
chmod +x "$TEMPDIR/ssrun"
```

We can now create the work package Tar/GZip file `input.tar.gz` – containing `input.pov`, includes and `ssrun`. This work package can be processed by the Scripting Service PU, i.e. `scripting-client`. The output archive will be written to `output.tar.gz`.

```
# ====== Create and distribute work package =========================
cd "$TEMPDIR"
find . -name "ssrun" -or -name "input.pov" -or -name "*.inc" | \
   xargs tar czf input.tar.gz
cd ..
scriptingclient -quiet -input=$TEMPDIR/input.tar.gz \
                       -output=$TEMPDIR/output.tar.gz
```

If an output file has been written, we can unpack it. If there is also a PNG file, our run has succeeded. Otherwise, there should be the log output of `povray`, which can be printed for debugging:

```
if [ -e "$TEMPDIR/output.tar.gz" ] ; then
   cd "$TEMPDIR"
   tar xzf output.tar.gz
   cd ..
   if [ -e "$TEMPDIR/output.png" ] ; then
     mv $TEMPDIR/output.png $OUTPUT
     rm -rf $TEMPDIR
   else
      echo >&2 "ERROR: No image has been created. Check log:"
      echo "------ LOG -------------------------------------------"
      cat "$TEMPDIR/output.txt"
      echo "----------------------------------------------------"
   fi
fi
```

In order to perform workload distribution, some PEs have to be in the Scripting Service pool. That is, start some scripting PEs by:

```
server -scripting -policy=LeastUsedDegradation:0.5 \
       -ssmaxthreads=2 ...
```

The policy is set to Least Used with Degradation (LUD). The load degradation by accepting a new request is 50%. Up to two sessions are processed simultaneously – since our lab PCs are dual-core machines.

Test your new script with a small test run, e.g. computing `landscape.pov` (see figure 4) in the resolution 128x96. Also record the session's message flow using WIRESHARK.

Question 30:
How is the message flow of the Scripting Service Protocol (SSP)?

Solution: The message flow is as follows:

1. SSP Ready from the PE,
2. A sequence of SSP Uploads to upload the work package to the PE,
3. SSP Keep-Alives and SSP Keep-Alive Acks in regular interval,
4. SSP Status with the result of the processing and
5. A Sequence of SSP Downloads to download the results package.

Question 31:
Can you imagine how failovers are handled?

Solution: Unfortunately, failovers are handled by just retrying the whole computation at a new PE. Already performed – but not completed – work is lost. That is, the session failover strategy is "abort and restart" [DR09]. Application checkpointing – which allows for session resumption from defined checkpoints – is not yet supported by the Scripting Service.

4.4 Parallelizing Image Computation

Now, we would like to process all `.pov` files in parallel. Therefore, we use another script – named `run-povray-for-files` which simply calls `povray-distribute` in background for each file provided as argment:

```
#!/bin/bash
POV_FILE_LIST=`\
( \
while [ x$1 != "x" ] ; do \
   echo $1 && \
   shift ; \
done \
) | sort -u`

for POV_FILE in $POV_FILE_LIST ; do
   ./povray-distribute 1024 768 $POV_FILE &
done
```

Execute this script in the `.pov` files directory by:

```
./run-povray-for-files *.pov
```

It will start processing all files in parallel. Not having enough PE capacity is no problem. A PU will try another PE after some delay. However, it is recommended to add all lab PCs to the Scripting Service pool. Otherwise, the computation can take a long time. . . .

References

[CDG06] A. Conta, S. Deering, and M. Gupta, *Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification*, Standards Track RFC 4443, IETF, March 2006, http://www.ietf.org/rfc/rfc4443.txt.

[DH98] S. Deering and R. Hinden, *Internet Protocol, Version 6 (IPv6)*, Standards Track RFC 2460, IETF, December 1998, http://www.ietf.org/rfc/rfc2460.txt.

[DR08a] T. Dreibholz and E. P. Rathgeb, *A Powerful Tool-Chain for Setup, Distributed Processing, Analysis and Debugging of OMNeT++ Simulations*, Proceedings of the 1st ACM/ICST OMNeT++ Workshop (Marseille/France), March 2008, http://tdrwww.iem.uni-due.de/dreibholz/rserpool/rserpool-publications/OMNeT++Workshop2008.pdf.

[DR08b] ______, *Towards the Future Internet – An Overview of Challenges and Solutions in Research and Standardization*, Proceedings of the 2nd GI/ITG KuVS Workshop on the Future Internet (Karlsruhe/Germany), November 2008, http://tdrwww.iem.uni-due.de/dreibholz/rserpool/rserpool-publications/FutureInternet2008.pdf.

[DR09] ______, *Overview and Evaluation of the Server Redundancy and Session Failover Mechanisms in the Reliable Server Pooling Framework*, International Journal on Advances in Internet Technology (IJAIT) **2** (2009), no. 1, 1–14, http://tdrwww.iem.uni-due.de/dreibholz/rserpool/rserpool-publications/IJAIT2009.pdf.

[Dre07] T. Dreibholz, *Reliable Server Pooling – Evaluation, Optimization and Extension of a Novel IETF Architecture*, Ph.D. thesis, University of Duisburg-Essen, Faculty of Economics, Institute for Computer Science and Business Information Systems, March 2007, http://duepublico.uni-duisburg-essen.de/servlets/DerivateServlet/Derivate-16326/Dre2006-final.pdf.

[Dre10a] ______, *Applicability of Reliable Server Pooling for Real-Time Distributed Computing*, Internet-Draft Version 08, IETF, Individual Submission, January 2010, http://www.watersprings.org/pub/id/draft-dreibholz-rserpool-applic-distcomp-08.txt.

[Dre10b] ______, *Handle Resolution Option for ASAP*, Internet-Draft Version 06, IETF, Individual Submission, January 2010, http://www.watersprings.org/pub/id/draft-dreibholz-rserpool-asap-hropt-06.txt.

[Dre10c] ———, *Thomas Dreibholz's RSerPool Page*, 2010, http://tdrwww.exp-math.uni-essen.de/dreibholz/rserpool.

[DT08] T. Dreibholz and M. Tüxen, *Reliable Server Pooling Policies*, RFC 5356, IETF, September 2008, http://www.ietf.org/rfc/rfc5356.txt.

[DZR09] T. Dreibholz, X. Zhou, and E. P. Rathgeb, *SimProcTC – The Design and Realization of a Powerful Tool-Chain for OMNeT++ Simulations*, Proceedings of the 2nd ACM/ICST OMNeT++ Workshop (Rome/Italy), March 2009, http://tdrwww.iem.uni-due.de/dreibholz/rserpool/rserpool-publications/OMNeT++Workshop2009.pdf.

[Jun05] A. Jungmaier, *Das Transportprotokoll SCTP*, Ph.D. thesis, Universität Duisburg-Essen, Institut für Experimentelle Mathematik, August 2005, http://miless.uni-duisburg-essen.de/servlets/DocumentServlet?id=12152.

[LOTD08] P. Lei, L. Ong, M. Tüxen, and T. Dreibholz, *An Overview of Reliable Server Pooling Protocols*, Informational RFC 5351, IETF, September 2008, http://www.ietf.org/rfc/rfc5351.txt.

[NNS98] T. Narten, E. Nordmark, and W. Simpson, *Neighbor Discovery for IP Version 6 (IPv6)*, Standards Track RFC 2461, IETF, December 1998, http://www.ietf.org/rfc/rfc2461.txt.

[Pos81] J. Postel, *Internet Protocol*, Standards Track RFC 791, IETF, September 1981, http://www.ietf.org/rfc/rfc791.txt.

[POV10] POV-Team, *POV-Ray – The Persistence of Vision Ray Tracer*, 2010, http://www.povray.org.

[Ste07] R. Stewart, *Stream Control Transmission Protocol*, Standards Track RFC 4960, IETF, September 2007, http://www.ietf.org/rfc/rfc4960.txt.

[SXST08a] R. Stewart, Q. Xie, M. Stillman, and M. Tüxen, *Aggregate Server Access Protcol (ASAP)*, RFC 5352, IETF, September 2008, http://www.ietf.org/rfc/rfc5352.txt.

[SXST08b] ———, *Aggregate Server Access Protocol (ASAP) and Endpoint Handlespace Redundancy Protocol (ENRP) Parameters*, RFC 5354, IETF, September 2008, http://www.ietf.org/rfc/rfc5354.txt.

[Var09] A. Varga, *OMNeT++ Discrete Event Simulation System*, 2009, http://www.omnetpp.org.

[XSS+08] Q. Xie, R. Stewart, M. Stillman, M. Tüxen, and A. Silverton, *Endpoint Handlespace Redundancy Protocol (ENRP)*, RFC 5353, IETF, September 2008, http://www.ietf.org/rfc/rfc5353.txt.