OpenDiameter User Guide

Project: Diameter + PANA + EAP-FAST

8/25/2014

List of Authors and Changes

Version	Author	Date	Comments
0.9	Dr. Bing Li	2013-10-25	Initial version
1.0	Ron Brash	2014-08-25	Updated document to include missing information

Contents

Terminology and Vocabulary

SSL	Transport Layer Security (TLS) and its predecessor, Secure Sockets Layer (SSL), are cryptographic protocols designed to provide communication security over the Internet.They use X.509 certificates and hence asymmetric cryptography to authenticate the counterparty with whom they are communicating, and to exchange a symmetric key.
Certificate	In cryptography, a public key certificate (also known as a digital certificate or identity certificate) is an electronic document used to prove ownership of a public key.
CA	In cryptography, a certificate authority or certification authority (CA) is an entity that issues digital certificates.
TLS	See SSL
PANA	PANA is an acronym for Protocol for carrying Authentication for Network Access (PANA). It provides network access authentication by working as an EAP lower layer for transmitting EAP packets. PANA encapsulates EAP authentication methods inside EAP packets between a PANA enabled client (PaC) and a PANA Authentication Agent (PAA).
PaC	See PANA.
PAA	See PANA.
PAC	PAC is an acronym for Protected Access Credential.
EP	EP is an acronym for Enforcement Point when used in relation to PANA. It acts as a filter of the packets which source is an authenticated PaC. Basically, an EP is a network node which drops packets according to some parameters provided as results of the authentication processes.
AS	AS is an acronym for Authentication Server when used in relation to PANA.
NAS	NAS is an acronym for network Authentication Server
AAA	In computer security, AAA commonly stands for authentication, authorization and accounting. It refers to security architecture for distributed systems for controlling which users are allowed access to which services, and tracking which resources they have used.
EAP	Extensible Authentication Protocol, or EAP, is an authentication framework frequently used in wireless networks and point-to-point connections. EAP is also used for providing for the transport and usage of keying material and parameters generated by EAP methods.
Network	A computer network or data network is a telecommunications network that allows computers to exchange data.
Localhost	In computer networking, localhost means this computer. It is a hostname that the computer's software and users may employ to access the computer's own network services via its loopback network interface. Using the loopback interface bypasses local network interface hardware.
MD5	The MD5 message-digest algorithm is a widely used cryptographic hash function producing a 128-bit (16-byte) hash value, typically expressed in text format as a 32 digit hexadecimal number.
Ping	Ping is a computer network administration utility used to test the reachability of a host on an Internet Protocol (IP) network and to measure the round-trip time for messages sent from the originating host to a destination computer.

Executive Summary

This project aims to provide documentation and the source code with modifications to the following protocols:

- Diameter
- PANA
- EAP-FAST

Diameter

Diameter is a computer network protocol used to provide authentication, authorization and accounting (AAA). It has roots with the RADIUS protocol and is intended to be RADIUS'S eventual replacement using a server and client architecture.

The Diameter base protocol is defined by (IETF RFC 6733)and defines the minimum requirements for an AAA protocol. Diameter applications such as NASREQ or Mobile IPv4 can extend the base DIAMETER protocol by adding new commands, attributes and/or both.

EAP

Extensible Authentication Protocol (EAP) provides an authentication framework for support of multiple authentication methods.

EAP works by creating an exchange between an authenticator and a peer (another name for client). The authenticator uses one or more EAP methods in sequence to authenticate the peer.

Instead of requiring that the authenticator support all authentication methods, EAP permits the use of a backend authentication server which can implement EAP methods with the authenticator acting as a pass-through. When using this deployment method, a backend authentication server is connected with the authenticator; the actual authentication will be performed by the backend authentication server or software. The authenticator will forward the received EAP packets from the peer to the backend authentication server; alternatively, the replies from the backend server will be forwarded back to the peer being authenticated.

EAP is not directly implemented on top of the IP layer. EAP is carried by both Diameter (placeholder) and the PANA protocol.

EAP-FAST is one of the many possible EAP authentication methods, but for the purpose of this project – it will be introduced into the Open Diameter project.

The details of the EAP architecture and its deployment are specified in RFC 3748 (IETF RFC 3748).

PANA

PANA is an acronym for Protocol for carrying Authentication for Network Access (PANA). It provides network access authentication by working as an EAP lower layer for transmitting EAP packets. PANA encapsulates EAP authentication methods inside EAP packets between a PANA enabled client (PaC) and a PANA Authentication Agent (PAA).

The PaC using PANA interacts with the PAA which then interrogates an Authentication Server (AS) for the authorization and authentication of a PaC.

There are two potential implementations of an AS:

- Located on the same host as the PAA
- Located elsewhere on the network

If the AS is located on the same network node as the PAA, an authentication API is sufficient for relaying communications. Alternatively, if the AS is separated from the PAA, an AAA protocol such as Diameter will be required for network communications. The AS is deployed as a conventional backend Authorization, Authentication and Accounting Server (AAAS) to terminate both EAP and EAP methods.

PANA can also use a PANA Enforcement Point (EP) which can be used as a checkpoint to effectively block or allow network traffic from authorized or unauthorized PaCs. The EP can also reside either on the same host as the PAA or another node, however, communication will require the use of another protocol such as SNMP.

The details of the PANA architecture and operation are specified in RFC 5191 (IETF 5191) and RFC 5193 (IETF 5193).

Introduction to EAP-FAST Open Diameter Source Code

Source Code Background

Most of the source code used in this project has been reused from the Open Diameter project (OpenDiameter.org). The Open Diameter project provides the complete source code of Diameter and PANA, but an incomplete source code implementation for EAP. In the openly available (and under GPL), only two authentication methods are supported: MD5 and Archie.

For the PANA implementation, it now supports draft-ieft-pana-pana-07 instead of RFC 5191 (IETF 5191). In addition, there are additional fixes for errors and warnings as well as a new standalone PAC generation program.

Source Code Development

Much of the source code is based on the source code provided by the opendiameter-1.0.7-I release that was provided by the Open Diameter project. However, the latest code opendiameter-1.0.8-b release will take the code from Dr. Bing Li's research and be forward ported to the later developed code located within the project's SVN repository.

This code provides support for EAP-FAST in addition to Diameter and PANA. It also provides three stable applications: aaad, nasd, pacd, generate_pac

- Aaad supplies the Diameter server and EAP backend authentication server
- Nasd supplies the Diameter client, EAP pass-through server and PANA Agent (PAA)
- Pacd supplies the PANA Client (PaC) and EAP client
- Generate_pac provides a pre-generated opaque PAC

This source code is considered to be a new version, although a fork of the package provided by Dr. Li Bing will also be made available.

Note: In order to implement EAP-FAST, a modified version of OpenSSL must be installed – the patch is provided within the patch directory located in the root of the cplusplus directory and is known to work against OpenSSL 1.0.0m. This patch provides an extension of the TLS handshake protocol to use Protected Access Credentials (PAC) that is missing from the OpenSSL library. However, this implementation of EAP-FAST does not include support for RFC 5422 (RFC 5422)**which indicates that an EAP backend AS can provide a PAC dynamically; it only manually generates and is used by the configured pacd application.**

Source Code Interactions

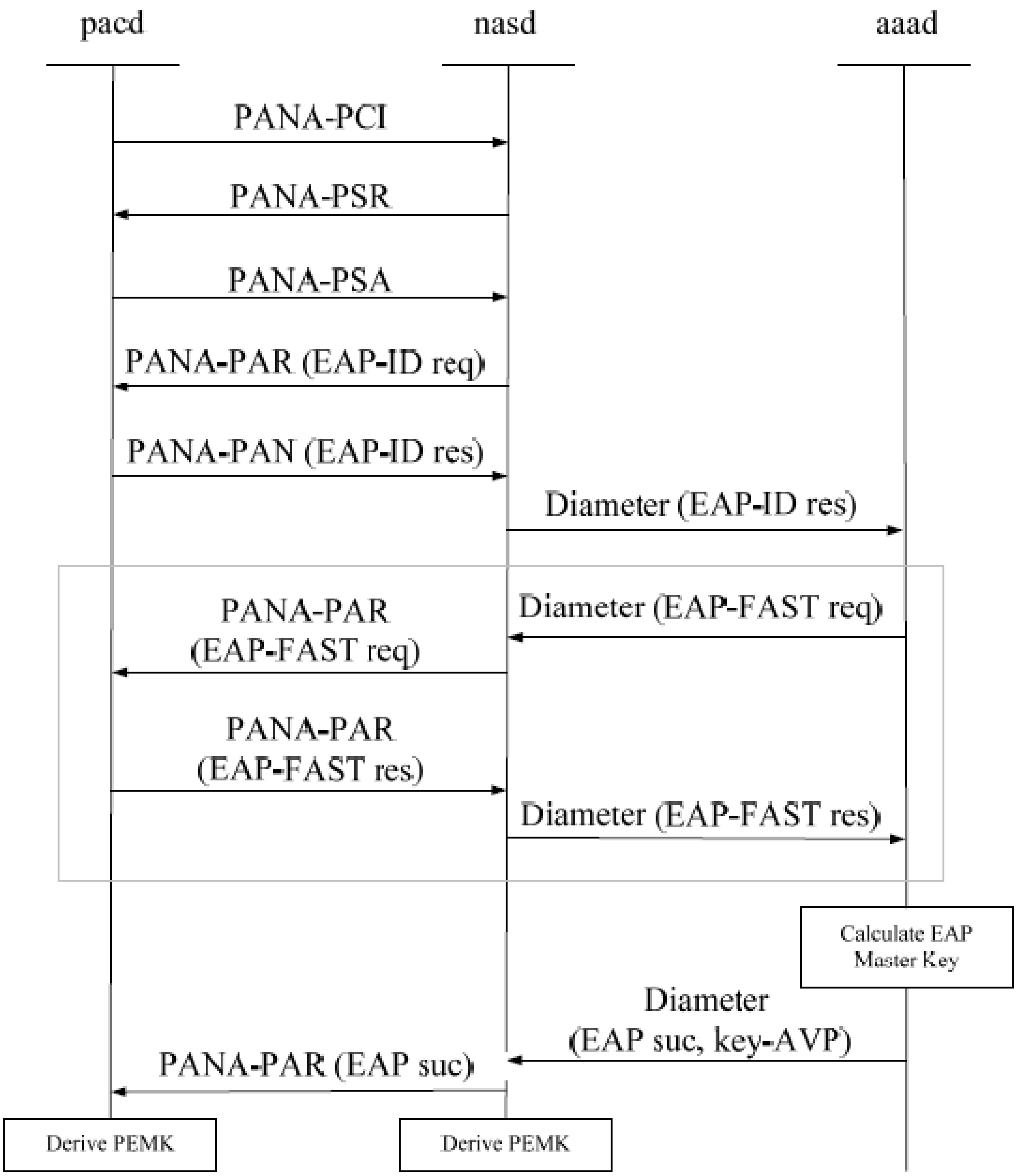

Figure 1 Interaction among the three daemons

Installing Prerequisite Software

In order to install the Open Diameter source code, the system must have the following software installed as well as the modified/required libraries.

System Preparations

At a minimum, a good development system should have at least the following packages already installed (on an Ubuntu based system).

```
sudo apt-get install wget build-essential asciidoc binutils bzip2 gawk gettext
libncurses5-dev libz-dev patch unzip zlib1g-dev lib32gcc1 libc6-dev-i386
subversion libtool autoconf python python-dev libbz2-dev geany
```

OpenDiameter

Retrieve the OpenDiameter source code from SVN, but don't attempt to install or compile it yet.

```
svn checkout svn://svn.code.sf.net/p/diameter/svn/ diameter-svn
```

OpenSSL (with Modifications)

The OpenSSL Project is a collaborative effort to develop a robust, commercial-grade, full-featured, and Open Source toolkit implementing the Secure Sockets Layer (SSL v2/v3) and Transport Layer Security (TLS v1) protocols as well as a full-strength general purpose cryptography library. The project is managed by a worldwide community of volunteers that use the Internet to communicate, plan, and develop the OpenSSL toolkit and its related documentation.

There are older (and newer) versions of OpenSSL (Openssl.org) that have vulnerabilities, but it appears that 1.0.0m is compatible with this version of OpenDiameter.

```
wget https://www.openssl.org/source/openssl-1.0.0m.tar.gz

tar -xzvf openssl-1.0.0m.tar.gz

cd openssl-1.0.0m/

Copy the SSL patch into the directory

cp <WHEREVER THIS IS>/diameter-svn/cplusplus/SSLpatch/OpenSSL-1.0.0m-TLS.patch
.
```

Try a dry-run first before patching.

```
patch -p2 --dry-run < OpenSSL-1.0.0m-TLS.patch
```

If the patch is successful, apply the patch

```
patch -p2 < OpenSSL-1.0.0m-TLS.patch
```

Continue configuration and compilation of the OpenSSL library.

```
./config --prefix=/usr --openssldir=/etc/ssl --libdir=lib shared

make

make test

sudo make install
```

ACE

The ADAPTIVE Communication Environment (ACE) is a freely available, open-source object-oriented (OO) framework that implements many core patterns for concurrent communication software. ACE provides a rich set of reusable C++ wrapper facades and framework components that perform common communication software tasks across a range of OS platforms. The communication software tasks provided by ACE include event de-multiplexing and event handler dispatching, signal handling, service initialization, inter-process communication, shared memory management, message routing, dynamic (re)configuration of distributed services, concurrent execution and synchronization. For more information go to (ACE).

```
wget http://download.dre.vanderbilt.edu/previous_versions/ACE-5.7.6.tar.gz

tar -xzvf ACE-5.7.6.tar.gz

cd ACE_wrappers

mkdir build

cd build

../configure --prefix=/usr

make

sudo make install
```

BOOST

Boost provides free portable peer-reviewed C++ libraries. The emphasis is on portable libraries which work well with the C++ Standard Library. See (Boost.org) for more information.

Previously Boost 1.41.0 worked, however, you may use the newest version (1.55.0 with some tweaks to allow for the "Classic" API (Opendiameter 1.0.7-j has these tweaks).

```
wget
http://downloads.sourceforge.net/project/boost/boost/1.55.0/boost_1_55_0.tar.gz

tar -xzvf boost_1_55_0.tar.gz

cd boost_1_55_0/

./bootstrap.sh --prefix=/usr --libdir=/usr/lib --includedir=/usr/include

./bjam

sudo ./bjam install
```

Xerces

Xerces-C++ is a validating XML parser written in a portable subset of C++. Xerces-C++ makes it easy to give your application the ability to read and write XML data. A shared library is provided for parsing, generating, manipulating, and validating XML documents. Xerces-C++ is faithful to the XML 1.0 and 1.1 recommendations and many associated standards.

The parser provides high performance, modularity, and scalability. Source code, samples and API documentation are provided with the parser. For portability, care has been taken to make minimal use of templates, no RTTI, and minimal use of #ifdefs. For more information go to the Xerces website (Xerces).

```
wget http://apache.mirror.gtcomm.net//xerces/c/3/sources/xerces-c-3.1.1.tar.gz
```

```
tar -xzvf xerces-c-3.1.1.tar.gz

cd xerces-c-3.1.1/

./configure --prefix=/usr --libdir=/usr/lib --includedir=/usr/include

make

sudo make install
```

Compiling OpenDiameter

The Makefles are generated by the Autotools/automake utilities. To compile and install OpenDiameter, perform the following:

```
Cd diameter-svn/cplusplus

aclocal -f

autoconf -f

automake -f

./configure --enable-eap-md5 --with-eap-tls --with-eap-fast

make

make install
```

Configuring OpenDiameter

PAC Generation

To generate the pac file, execute “generate_pac” and follows the step according to the prompt.

To create a valid PAC, set the encryption key value with pac_opaque_encryption value in file server.eap-fast.xml and set the life-time to a decent value.

```
cd applications/eap_fast_generate_pac/

./generate_pac

Input the A-ID:1

Input the I-ID:1

Input the A-ID_INFO:1

Input the encryption key (16 BYTE) with hex string(
000102030405060708090a0b0c0d0e0f ):000102030405060708090a0b0c0d0e0f

Input the lifetime of PAC (min):1000

Set Pac type:1
```

```
Generating random PAC key

       Generated random PAC and time is 1409188001

Building generic PAC struct with user input

       Built generic PAC struct with user input

Building EAP Fast Opaque structure

       Built EAP Fast Opaque structure

Building EAP Fast Info structure

       Built EAP Fast Info structure

Writing PAC to file

opaque - hexdump(len=48): d2 7f 94 35 0b ec e1 b8 bb 68 80 18 5d 65 8b 65 fc 83
2e 3c 4b b0 99 af fa 3c 00 83 06 a9 e0 84 8f e2 13 99 80 cc fc 1f 94 13 5e ec
70 fa df 3e

       Opened file: pac

       Attempting to close file

       Closed file

EAP-FAST: Wrote 1 PAC entries into 'pac' (bin)
```

To create an invalid PAC, set the encryption key with other value or set the life time short.

The output of the application will be located within a PAC file named “pac”

```
sudo cp pac /etc/opendiameter/pac/config/
```

Generate Certificates for aaad and pacd

EAP-FAST requires the certificates. When the PAC is invalid, the EAP-FAST downgrades to EAP-TLS where the certificates are used to authenticate EAP peer and EAP authenticator. OpenSSL utilities are used to generate certificates for EAP peer (client) and EAP authenticator (server).

Generate CA Certificates

Generate the CA SSL certificates using the following commands after creating the directory “demoCA”.

```
mkdir demoCA/;

cd demoCA;

mkdir private crl certs newcerts;

echo '01' > serial; touch index.txt;

openssl genrsa -out private/ca-key.pem -des3 1024

openssl req -new -x509 -key private/ca-key.pem -out ca-cert.pem –pass:123456 -
subj "/C=CA/ST=BC/L=Nanaimo/O=TestLtd./OU=IT/CN=Ron Brash"
```

Generate Server Certificates

Generate the server SSL certificates using the following commands:

```
openssl genrsa -out private/srv-key.pem -des3 1024 -pass:123456

openssl req -new -key private/srv-key.pem -out crl/srv-csr.pem
```

Here the password is needed to input and some base information is required:

```
"Enter pass phrase for private/srv-key.pem:

You are about to be asked to enter information that will be incorporated

into your certificate request.

What you are about to enter is what is called a Distinguished Name or a DN.

There are quite a few fields but you can leave some blank

For some fields there will be a default value,

If you enter '.', the field will be left blank.

-----

Country Name (2 letter code) [AU]:CA

State or Province Name (full name) [Some-State]:BC

Locality Name (eg, city) []:Nanaimo

Organization Name (eg, company) [Internet Widgits Pty Ltd]: TestLtd.

Organizational Unit Name (eg, section) []:IT

Common Name (eg, YOUR name) []:server

Email Address []:ron.brash@gmail.com

Please enter the following 'extra' attributes

to be sent with your certificate request

A challenge password []:123456

An optional company name []:server"
```

NOTE: NOTICE THAT THE CHALLENGE PASSWORD HAS BEEN SET.

```
cd ../

openssl ca -in demoCA/crl/srv-csr.pem -out demoCA/certs/srv-cert.pem –
pass:123456 -cert demoCA/ca-cert.pem -keyfile demoCA/private/ca-key.pem
```

Generate Client Certificates

Generate the client SSL certificates.

```
cd demoCA/

openssl genrsa -out private/clt-key.pem -des3 1024 –pass:123456

openssl req -new -key private/clt-key.pem -out crl/clt-csr.pem
```

Here the password is needed to input and some base information is required:

```
“Enter pass phrase for private/clt-key.pem:
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:CA
State or Province Name (full name) [Some-State]:BC
Locality Name (eg, city) []:Nanaimo
Organization Name (eg, company) [Internet Widgits Pty Ltd]: TestLtd.
Organizational Unit Name (eg, section) []:IT
Common Name (eg, YOUR name) []:client
Email Address []:ron.brash@gmail.com
Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:123456
An optional company name []:client”
```

NOTE: NOTICE THAT THE CHALLENGE PASSWORD HAS BEEN SET.

```
cd ../
```

Now generate the client certificate using the following command:

```
openssl ca -in demoCA/crl/clt-csr.pem -out demoCA/certs/clt-cert.pem –
pass:123456 -cert demoCA/ca-cert.pem -keyfile demoCA/private/ca-key.pem
```

AAAD Certificates

Create the SSL certificate directory if it has not already been created. Then copy the SSL certificates to the correct location.

```
sudo mkdir -p /etc/opendiameter/aaa/config/fast/

cd demoCA/private

sudo cp ca-key.pem /etc/opendiameter/aaa/config/fast/

sudo cp srv-key.pem /etc/opendiameter/aaa/config/fast/

cd ../demoCA/certs/

sudo cp srv-cert.pem /etc/opendiameter/aaa/config/fast/
```

PACD Certificates and Config

Copy the SSL certificates and configuration file for the PaC daemon.

```
sudo cp ca-cert.pem /etc/opendiameter/pac/config/fast/

sudo cp certs/clt-cert.pem /etc/opendiameter/pac/config/fast/

sudo cp private/clt-key.pem /etc/opendiameter/pac/config/fast/
```

Edit the configuration file to look like the below entry.

```
sudo geany /etc/opendiameter/aaa/config/aaad_user_db.xml

       <user_entry>

       <name_match>Server</name_match>

       <eap_method>md5</eap_method>

       <md5>

          <password_type>flat</password_type>

          <secret>12345</secret>

       </md5>

    </user_entry>
```

Configure Daemons

Copy over and edit the server EAP configuration file.

```
sudo cp /etc/opendiameter/eap/server.eap-fast.xml /etc/opendiameter/aaa/config/

sudo geany /etc/opendiameter/aaa/config/server.eap-fast.xml
```

Verify that the PAC value is the same - otherwise it will be invalid.

```
<pac_opaque_encr>000102030405060708090a0b0c0d0e1f</pac_opaque_encr>
```

```
sudo geany /etc/opendiameter/aaa/config/aaad_diameter_server.xml
```

Edit real/identity fields for network names.

```
<transport_mngt>
         <identity>localaaa.localdomain.net</identity>
      <realm>localdomain.net</realm>
      <tcp_listen_port>1812</tcp_listen_port>
      <sctp_listen_port>1813</sctp_listen_port>
      <use_ipv6>0</use_ipv6>
      <watchdog_timeout>4</watchdog_timeout>
      <reconnect_interval>30</reconnect_interval>
      <reconnect_max>3</reconnect_max>
      <request_retransmission_interval>10</request_retransmission_interval>
      <max_request_retransmission_count>3</max_request_retransmission_count>
      <receive_buffer_size>2048</receive_buffer_size>
      <advertised_hostname>localaaa.localdomain.net</advertised_hostname>
      <peer_table>
          <expiration_time>1</expiration_time>
          <peer>
              <hostname>localnas.localdomain.net</hostname>
              <port>1810</port>
              <use_sctp>0</use_sctp>
              <tls_enabled>0</tls_enabled>
          </peer>
      </peer_table>
```

```
sudo geany /etc/opendiameter/nasd/config/nasd_diameter_eap.xml
```

Edit host name fields

```
   <transport_mngt>
      <identity>localnas.localdomain.net</identity>
      <realm>localdomain.net</realm>
      <tcp_listen_port>1810</tcp_listen_port>
```

```
<sctp_listen_port>1811</sctp_listen_port>

<use_ipv6>0</use_ipv6>

<watchdog_timeout>3</watchdog_timeout>

<reconnect_interval>30</reconnect_interval>

<reconnect_max>3</reconnect_max>

<request_retransmission_interval>10</request_retransmission_interval>

<max_request_retransmission_count>3</max_request_retransmission_count>

<receive_buffer_size>2048</receive_buffer_size>

<peer_table>

    <expiration_time>1</expiration_time>

    <peer>

        <hostname>localnas.localdomain.net</hostname>

        <port>1812</port>

        <use_sctp>0</use_sctp>

        <tls_enabled>0</tls_enabled>

    </peer>

</peer_table>
```

Edit route information:

```
<route_table>

    <expire_time>0</expire_time>

    <route>

       <realm>localdomain.net</realm>

       <role>1</role>

       <redirect_usage>0</redirect_usage>

       <application>

          <application_id>5</application_id>

          <vendor_id>0</vendor_id>

          <peer_entry>

              <server>localaaa.localdomain.net</server>

              <metric>1</metric>
```

```
                    </peer_entry>

                 </application>
```

```
    sudo geany /etc/opendiameter/pac/config/pana_pac.xml
```

Verify host set to either localhost or 127.0.0.1.

```
            <paa_ip_address>localhost</paa_ip_address>
```

Edit hosts file to know where these domain names are located.

```
    sudo vi /etc/hosts

    127.0.0.1 localaaa localaaa.localdomain.net

    127.0.0.1 localnas localnas.localdomain.net
```

Testing

The following sections outline two scenarios in which this project was designed to operate. The documentation in the previous sections assumes testing on a single machine. However, testing on multiple machines can be performed via section Multiple Machines/Over Network

Single Machine

This is the simple test where three applications are implemented and ran using a single machine.

Flushing iptables on the test machine to avoid any filtering is an optional step

```
    sudo iptables -F
```

Run the three applications in three terminals respectively. The commands are:

```
    sudo applications/aaa/aaad

    sudo applications/nas/nasd

    sudo applications/pana/pacd
```

NOTE: PLEASE RUN AAAD AT FIRST. RUN NASD AFTER THE APPLICATION OF AAAD PROMPTS WAITING FOR CONNECTING. WHEN THE "WATCHDOG MSG" HAS BEEN SEEN AT AAAD AND NASD, IT INDICATES THAT NASD AND AAAD HAVE WORKED NORMALLY. AFTER THAT, RUN THE PACD APPLICATION. (I SUGGEST CHECKING THE EAP INFORMATION HAS BEEN OUTPUT FROM AAAD THEN WAITING AT LEAST TWO MINUTES TO START PACD AFTER AAAD AND NASD STARTS.)

Multiple Machines/Over Network

Instead of using a single machine, this describes the procedure to setup a test over multiple machines or a network.

Flushing iptables on the test machine to avoid any filtering is an optional step

```
sudo iptables -F
```

Name the two machines using a schema similar to the below example:

```
The machine that runs aaad: aaa.aaadomain.net

The machine that runs nasd: nas.nasdomain.net
```

Setup a DNS server to provide the machine names to other systems and verify that the DNS names are resolvable.

```
In the machine that runs pacd: ping nas.nasdomain.net

In the machine that runs nasd: ping aaa.aaadomain.net
```

Modify the configuration file “aaad_diameter_server.xml” in reference to the machine name.

Modify the configuration file “nasd_diameter_eap.xml” in reference to the machine name.

Modify the configure file “pana_pac.xml” in reference to the machine name.

Run the three applications in three terminals respectively. The commands are:

```
sudo applications/aaa/aaad

sudo applications/nas/nasd

sudo applications/pana/pacd
```

Note: Please run aaad at first. Run nasd after the application of aaad prompts waiting for connecting. When the “watchdog msg” has been seen at aaad and nasd, it indicates that nasd and aaad have worked normally. After that, run the pacd application. (I suggest checking the EAP information has been output from aaad then waiting at least two minutes to start pacd after aaad and nasd starts.)

RFCs and References

ACE. (n.d.). Retrieved from http://www.cs.wustl.edu/~schmidt/ACE.html

Boost.org. (n.d.). Retrieved from http://www.boost.org

IETF 5191. (n.d.). Retrieved from http://tools.ietf.org/html/rfc5191

IETF 5193. (n.d.). Retrieved from http://tools.ietf.org/html/rfc5193

IETF RFC 3748. (n.d.). Retrieved from http://tools.ietf.org/html/rfc3748

IETF RFC 6733. (n.d.). *IETF RFC 6733*. Retrieved from RFC 6733 Diameter Base Protocol: http://tools.ietf.org/html/rfc6733

OpenDiameter.org. (n.d.). Retrieved from http://www.opendiameter.org

Openssl.org. (n.d.). Retrieved from https://www.openssl.org/

RFC 5422. (n.d.). Retrieved from http://tools.ietf.org/html/rfc5422

Xerces. (n.d.). Retrieved from http://xerces.apache.org/xerces-c/