University College London
Department of Computer Science
M.Sc Data Communications, Network and Distributed Systems

Extensible Secure Event and Report Toolkit

User Manual

Group Members
Shaohua Yan
Yang Wang
Xiaoyue She
Ping Liang

Supervisor
Dr. Saleem Bhatti

30 Aug 2005

1 System Requirement

A set of software and third party libraries are required before running this toolkit. They are listed as follows:

1. Python 2.3.5 or above.
2. OpenSSL 0.9.7E or above.
3. Twisted (third party library of SSL)
4. MySQL 4.1 or above
5. Port 5555 must be available for PoDRegistry.

2 Installation

Once all the items in section A.1 have been installed, the first step is to copy and uncompress the source file. The following command is how to do this uncompressing:

tar -xvvf ExSERT.tar

After uncompressing the source file, a number of directories and files are generated, the directory structure is listed below:

Compiler/			
	Compiler.py		
	Configuration.ini		
	Report.xml		
Lib/			
	__init__.py		
	Base/		
		__init__.py	
		Base.dtd	
		Day.py	
		Hour.py	
		IPv4Address.py	
		IPv6Address.py	
		Integer.py	
		MACAddress.py	
		Minute.py	
		Month.py	
		NumOfBytes.py	
		PacketLoss.py	
		Second.py	
		String.py	
		RTT.py	
		Year.py	
	Function/		

		__init__.py	
		BinaryEncoding.py	
		LinkStatus.py	
		LinkUtilization.py	
		PoDServer.py	
		StatisticRTT.py	
		ThresholdRTT.py	
PoDRegistry/			
	Configuration.py		
	PoDRegistry.py		
Demo/			
	LinkMeasure/		
		__init__.py	
		Class/	
		Common/	
		Glue/	
		PoD/	
	Client/		
		Cert/	
			ClientCert.pem
			ClientKey.pem
		Configuration.ini	
		Handlers.py	
		VFClient.py	
		XMLValidator.py	
		XmlToHTML	

In the Demo folder, a sample RMF (LinkMeasure) is created for references to the end user. An example client has also been implemented to test the system. Note that the toolkit only produces XML document to the client, which then can handle, validate, or display the XML document by whatever ways they prefer.

3 RMF Toolkit Creation

The following procedures explain how to create a new RMF toolkit using the compiler and provided API. Firstly, it details the steps taken to generate common code for system components using the compiler. Then it includes a description on how to create key pairs and certificates using OpenSSL. Finally, it illustrates the user on how to modify the generated code in order to complete the creation of the toolkit. An example has been used throughout the description these steps, and this typical

RMF is named LinkMeasure. Before doing all these steps, the python path that enables the program to locate the path of the API needs to be set up like below:

export PYTHONPATH=”~/ExSERT”

3.1 Common Code Generation

1. Move to the Compiler directory.
2. Create a report format for the toolkit, and sample report format can be found in this directory.
3. Configure the compiler by modifying Compiler/Configuration.ini file in which the toolkit name, report name and the location of the library can be specified.
4. Run the compiler by typing

   python Compiler.py
5. The common code for the system components of this example have been generated and structured as below:

LinkMeasure/		Root of this RMF
	__init__.py	
	Class/	Contains the user-defined types
	Common/	Contains RMF configuration file and DTD file
	Glue/	Contains generated code for Glue
	PoD/	Contains generated code for PoD

3.2 Key Pairs and Certificate Creation

1. Modify file openssl.cnf to substitute default directory using your own specific directory.
2. Running the openSSL program by typing *openSSL* on the command line.
3. Create the self-signed CA certificate, which can sign the certificates for communication parties.

   openssl> req –new –x509 –keyout $dir/private/CAKey.pem –out $dir/private/CACert.pem –config /ssl/openssl.cnf
4. Create a server key file and certificate request file.

   openssl> req –new –keyout ServerKey.pem –out ServerReq.pem.
5. Concatenate these two files for the signing.

   Cat ServerReq.pem ServerKey.pem > Server.pem

6. Sign the server certificate.

   openssl> ca –policy policy_anything –out ServerCert.pem –config
   $dir/ssl/openssl.cnf –infiles Server.pem

7. Create and sign the client certificate in the same way.

   Openssl> req –new –keyout ClientKey.pem –out ClientReq.pem
   Cat ClientReq.pem Client.pem > Client.pem
   Openssl> ca –policy policy_anything –out ClientCert.pem –config
   $dir/ssl/openssl.cnf –infiles Client.pem

8. Copy CACert.pem ServerKey.pem and ServerCert.pem to ExSERT/PoD/Cert directory.
9. Copy ClientKey.pem and ClientCert.pem to Client/Cert.
10. Example certificates are included in Demo/LinkMeasure/PoD/Cert and Demo/Client/Cert folders for both the server and client respectively. The passwords for these two example certificates are “111111”.

3.3 Code Modification

1. Edit GlueLinkMeasureRS.py and copy it to the Glue directory. A sample implementation can be found in the Demo/LinkMeasure/Glue/.
2. Edit PoDDataAnalysier.py and copy it the PoD directory. For data analysing functions of the API, see Section 5.8 and the documentation. A sample implementation can be found in the Demo/LinkMeasure/PoD/.

4 Running the Program

1. Start the PoDRegistry in PoDRegistry directory by typing:

   python PoDRegistry.py

2. Configure the PoDServer by modifying the configuration.ini in LinkMeasure/PoD.
3. Start the PoDServer in /LinkMeasure/PoD directory by typing

   python PoDServer.py

   The password should be the one entered in A.2.2. For the example certificates, passwords are 111111 for both the client and server.

4. Run the sample client program in Demo/Client directory by typing

   python VFClient.py