Securing Communication using MRS Server

Introduction

MRS, Media Relay System, is an architecture and communication protocol in which “capture devices” stream video, audio and location information to “monitors” through the internet and/or other networks. An MRS Server is software which runs on a server computer and connects the capture devices with the monitors while applying access permission rules. In an MRS Server based architecture, the devices and monitors are all considered clients. Since the internet is very common, it is a natural choice for hosting the MRS communication between the devices, the server and the monitors. However, this makes it also possible for others to listen to the traffic along the way and possibly steal confidential material if it is not properly encrypted.

Encryption

MRS protocol allows, when selected in the client, to encrypt the traffic between the client and the server. Starting from version 5.4.0.0 it uses TLS protocol version 1.0 with a given server RSA asymmetric key and random AES symmetric keys encryption. The operator of the MRS Server is required to generate the RSA asymmetric key and a signed certificate (can be self-signed) and place it in the server in the “Server Authentication” web page under 'Administrator' in the main server administration web UI.

Key Generation Process

Step 1: Download and install Open SSL

Open SSL is a free tool for various encryption related purposes. Users of Linux or Unix have this tool already or can download and install this tool using their operating system package management tools. Developers can download and build this tool using this link: https://www.openssl.org/source/. Users of Windows can download and install this tool using this link: http://gnuwin32.sourceforge.net/packages/openssl.htm

Step 2: Open a terminal window

Open SSL receives commands via a terminal window. Users of Linux or Unix have many terminals to choose from. Users of Windows can click “Start”, then “Run” and then type “cmd” and click enter. In the open command window (Windows) go to the directory where the Open SSL was installed, for example: “**cd C:\Program Files\GnuWin32\bin**”.

Step 3: Generate an RSA key

In the terminal window, write: **openssl genrsa -out key1.pem 2048**
This command will create an RSA key of size 2048 bits and place it in the file “key1.pem”. Copy-paste the **complete** text from this file to the “Private Key” box in the “Server Authentication” page. Warning: this file is the key protecting all server communications. It must not be available to

anyone but the server itself. It is best to generate this file on the server itself and not allow access to it to anyone other the the administrator of the server.

Step 4: Generate a certificate signing request

In the terminal window, write: **openssl req -new -key key1.pem -out server.csr**
This command will create a signing request for the server key (key1.pem). It will store the output in the file "server.csr". The user will be asked to fill out a number of details. The important one, "Common Name", should be server's host name, such as "mrs2.infodraw.com", as it should appear in the browser or in the client server address box.

Note: Some users will encounter the following error: "Unable to load config info...". They will have to configure SSL for certificates. Here are instructions for SSL configuration as taken from http://www.flatmtn.com/article/setting-openssl-create-certificates:
Create 3 directories: sslcert sslcert/certs sslcert/private. The sslcert directory can be anywhere, but it will be simpler under the current directory of the command terminal.
Create file "serial" containing **'100001'** inside sslcert.
Create empty file "certindex.txt" inside sslcert.
Create file sslcert/openssl.cnf with the following text, change the text that appears in Bold Green:

```
# OpenSSL configuration file.
#

# Establish working directory.

dir                                     = .

[ ca ]
default_ca                              = CA_default

[ CA_default ]
serial                                  = $dir/serial
database                                = $dir/certindex.txt
new_certs_dir                           = $dir/certs
certificate                             = $dir/cacert.pem
private_key                             = $dir/private/cakey.pem
default_days                            = 365
default_md                              = md5
preserve                                = no
email_in_dn                             = no
nameopt                                 = default_ca
certopt                                 = default_ca
policy                                  = policy_match

[ policy_match ]
countryName                             = match
stateOrProvinceName                     = match
organizationName                        = match
organizationalUnitName                  = optional
commonName                              = supplied
emailAddress                            = optional

[ req ]
default_bits                            = 1024                  # Size of keys
default_keyfile                         = key.pem               # name of generated keys
default_md                              = md5                           # message digest algorithm
string_mask                             = nombstr               # permitted characters
distinguished_name                      = req_distinguished_name
req_extensions                          = v3_req

[ req_distinguished_name ]
# Variable name                         Prompt string
```

```
#------------------------        ---------------------------------
0.organizationName                      = Organization Name (company)
organizationalUnitName                  = Organizational Unit Name (department, division)
emailAddress                            = Email Address
emailAddress_max                        = 40
localityName                            = Locality Name (city, district)
stateOrProvinceName                     = State or Province Name (full name)
countryName                             = Country Name (2 letter code)
countryName_min                         = 2
countryName_max                         = 2
commonName                              = Common Name (hostname, IP, or your name)
commonName_max                          = 64

# Default values for the above, for consistency and less typing.
# Variable name                         Value
#------------------------        ------------------------------
0.organizationName_default              = My Company
localityName_default                    = My Town
stateOrProvinceName_default             = State or Providence
countryName_default                     = US

[ v3_ca ]
basicConstraints                        = CA:TRUE
subjectKeyIdentifier                    = hash
authorityKeyIdentifier                  = keyid:always,issuer:always

[ v3_req ]
basicConstraints                        = CA:FALSE
subjectKeyIdentifier                    = hash
```

set the configuration file with the command: **set OPENSSL_CONF=sslcert/openssl.cnf**
Repeat the command to generate the certificate signing request at the top of this section.

Step 5: Self sign the certificate

In the terminal window, write: **openssl x509 -req -in server.csr -signkey key1.pem -out server.crt**
This command will sign the request from “server.csr” with key from “key1.pem” and generate the certificate in the file “server.crt”. Copy-paste the **complete** text from “server.crt” file to the “Certificate” box in the “Server Authentication” page. This file is not confidential. Send it to the administrator's personal computer for server management (see step 6).

Step 6: Add certificate for browser access to the server

In the Windows machine double click “server.crt” file in the Windows Explorer Window. Click on “Install Certificate”, click “Next”, click “Finish”, click “Yes”. That will enable you to securely access the server from this machine, by using http**s**://server_ip:12654/admin, rather than plain insecure http.