Implementing ECC+PFS on LineRate (Part 2/3): Configuring SSL Offload on LineRate

Andrew Ragone, 2014-26-08

Overview

In case you missed it, Why ECC and PFS Matter: SSL offloading with LineRate details some of the reasons why ECC-based SSL has advantages over RSA cryptography for both performance and security.

We have already discussed how to pick an ECC curve and generate the appropriate certificates for it. This article will take these certificates and implement them in a LineRate System so that your organization can take advantage of the benefits that come along with SSL Offloading with these cryptology technologies.

Configuration: Adding the SSL Information to the LineRate System

There are a few aspects of the LineRate System that need to be configured in order for SSL/TLS Offloading to happen. Namely, the following items need to be done:

- Add the Private Key
- Add the Primary Certificate
- Configure an SSL Profile
- Attach the SSL Profile to a Virtual IP (i.e. Enable SSL Offloading)

Let's add the private key and primary certificates for our domain to the LineRate System. The certificate will be named "offloadCert" with a key of "offloadKey" and an SSL profile named "offload". The **key** and **certificate** commands are used for this as follows:

Private Key Configuration

```
LROS(config)# key offloadKey
*LROS(config-key:offloadKey)# pem-format
Enter a PEM format key followed by 'quit' on a line by itself.
-----BEGIN EC PARAMETERS-----
BgUrgQQAIg==
-----END EC PARAMETERS-----
-----BEGIN EC PRIVATE KEY-----
MIGkAgEBBDD1Kx9hghSGCTujAaqlnU2hs/spEOhfpKY9EO3mYTtDmKqkuJLKtv1P
1/QINzAU7JigBwYFK4EEACKhZANiAASLp1bvf/VJBJn4kgUFundwvBv03Q7c3tlX
kh6Jfdo3lpP2Mf/K09bpt+4RlDKQynajq6qAJ1tJ6Wz79EepLB2U40fC/3OBDFQx
5gSjRp8Y6aq8c+H8gs0RKAL+I0c8xDo=
-----END EC PRIVATE KEY-----
quit
*LROS(config-key:offloadKey)#
```

Primary Certificate Configuration

```
*LROS(config)# certificate offloadCert
*LROS(config-certificate:offloadCert)# pem-format
Enter a PEM format certificate followed by 'quit' on a line by itself.
-----BEGIN CERTIFICATE----
MIIC5jCCAm2gAwIBAgIETUKHWzAKBggqhkjOPQQDAzBtMQswCQYDVQQGEwJVUzEW
MBQGA1UEChMNRW50cnVzdCwgSW5jLjEfMB0GA1UECxMWRm9yIFRlc3QgUHVycG9z
ZXMgT25seTElMCMGA1UEAxMcRW50cnVzdCBFQ0MgRGVtb25zdHJhdGlvbiBDQTAe
Fw0xNDA4MTExODQ3MTZaFw0xNDEwMTAxOTE3MTZaMGkxHzAdBgNVBAsTFkZvciBU
ZXN0IFB1cnBvc2VzIE9ubHkxHTAbBgNVBAsTFFBlcnNvbmEgTm90IFZlcmlmaWVk
```

```
MScwJQYDVQQDEx5zc2xvZmZsb2FkLmxpbmVyYXRlc3lzdGVtcy5jb20wdjAQBgcq
hkjOPQIBBgUrgQQAIgNiAASLp1bvf/VJBJn4kgUFundwvBv03Q7c3tlXkh6Jfdo3
lpP2Mf/K09bpt+4RlDKQynajq6qAJ1tJ6Wz79EepLB2U40fC/3OBDFQx5gSjRp8Y
6aq8c+H8gs0RKAL+I0c8xDqjgeEwgd4wDgYDVR0PAQH/BAQDAgeAMB0GA1UdJQQW
MBQGCCsGAQUFBwMBBggrBgEFBQcDAjA3BgNVHR8EMDAuMCygKqAohiZodHRwOi8v
Y3JsLmVudHJ1c3QuY29tL0NSTC9lY2NkZW1vLmNybDApBgNVHREEIjAggh5zc2xv
ZmZsb2FkLmxpbmVyYXRlc3lzdGVtcy5jb20wHwYDVR0jBBgwFoAUJAVL4WSCGvgJ
zPt4eSH6cOaTMuowHQYDVR0OBBYEFESqK6HoSFIYkItcfekqqozX+z++MAkGA1Ud
EwQCMAAwCgYIKoZIzj0EAwMDZwAwZAIwXWvK2++3500EVaPbwvJ39zp2IIQ98f66
/7fgroRGZ2WoKLBzKHRljVd1Gyrl2E3BAjBG9yPQqTNuhPKk8mBSUYEi/CS7Z5xt
dXY/e7ivGEwi65z6iFCWuliHI55iLnXq7OU=
-----END CERTIFICATE-----
quit
*LROS(config-certificate:offloadCert)#
```

Remember to save the changes to memory!

```
*LROS(config)# write
Building configuration...
[OK]
LROS(config)#
```

Configuration: Confirming the Certificate has been added successfully

Now the SSL information needed to negotiate SSL/TLS session has been added to LineRate. To use these credentials, an SSL Profile needs to be created that points to them. Before we do so, let's perform a quick sanity check to ensure the certificate was loaded onto the system as we expect:

```
*LROS# show certificate brief
Certificate Subject Common Name (CN)
--------------------------------------------
offload sslloffload.lineratesystems.com
self-signed lros-default-host

Certificate Bundle Subject Common Name (CN)
--------------------------------------------
```

Looks good! Now to configure the SSL Profile

Configuration: Create a SSL Profile on the LineRate System

An SSL base profile will be created which will allow common parts of the SSL profile to be used among several Virtual Servers in the LineRate System. This may be useful, for instance, in cases where the same certificate applies to multiple sub-domains that are handled via different Virtual Servers on the LineRate System.

```
*LROS(config)# ssl profile base offloadBase
*LROS(config-ssl-profile:offloadBase)# attach primary-certificate offloadCert
*LROS(config-ssl-profile:offloadBase)# attach private-key offloadKey
*LROS(config-ssl-profile:offloadBase)# ecc-curve-list openssl secp384r1
*LROS(config-ssl-profile:offloadBase)# cipher-list openssl "ECDH:!RC4"
```

The specific offload SSL profile can then be configured to use the base SSL profile's configuration. Attributes of the offloadBase profile are inherited and can be overwritten in the SSL offload profile, if needed:

```
*LROS(config-ssl-profile:offloadBase)# ssl profile offload
*LROS(config-ssl-profile:offload)# base offloadBase
```

In the above configuration, you may have noticed something new: the "ECDH:!RC4" cipher-list configuration. Here, we are removing the more insecure options from the ECDH cipher suite. This is a quick way to intersect the ECC cipher suites with the High-Security cipher suite set configured on LineRate. To demonstrate this, here is breakout of what each cipher-list configuration option provides:

High Security Cipher Suites	All ECDH Cipher Suites	Removed ECDH Cipher Suites	High-Security ECC+PFS Cipher Suites
cipher-list openssl "HIGH"	*cipher-list openssl "ECDH"*		*cipher-list openssl "ECDH:!RC4"*
ECDHE-ECDSA-AES256-SHA384	ECDHE-ECDSA-AES256-SHA384	ECDHE-RSA-RC4-SHA	ECDHE-ECDSA-AES256-SHA384
ECDHE-ECDSA-AES256-SHA	ECDHE-ECDSA-AES256-SHA	ECDHE-ECDSA-RC4-SHA	ECDHE-ECDSA-AES256-SHA
AES256-GCM-SHA384	ECDHE-ECDSA-DES-CBC3-SHA	AECDH-RC4-SHA	ECDHE-ECDSA-DES-CBC3-SHA
AES256-SHA256	ECDHE-ECDSA-AES128-SHA256	ECDH-RSA-RC4-SHA	ECDHE-ECDSA-AES128-SHA256
AES256-SHA	ECDHE-ECDSA-AES128-SHA	ECDH-ECDSA-RC4-SHA	ECDHE-ECDSA-AES128-SHA
ECDHE-ECDSA-DES-CBC3-SHA	ECDHE-RSA-RC4-SHA		
DES-CBC3-SHA	ECDHE-ECDSA-RC4-SHA		
ECDHE-ECDSA-AES128-SHA256	AECDH-RC4-SHA		
ECDHE-ECDSA-AES128-SHA	ECDH-RSA-RC4-SHA		
AES128-GCM-SHA256	ECDH-ECDSA-RC4-SHA		
AES128-SHA256			
AES128-SHA			

A comprehensive list of SSL cipher-suites supported on LineRate can be found on the online documentation.

To ensure the SSL profile has been configured correctly, let's verify it:

```
LROS# show ssl profile offload
Configuration
  Primary Cert Name:           offloadCert inherited from offloadBase
  Private Key Name:            offloadKey inherited from offloadBase
  Chained Cert Name:

  Primary Cert and Key Match:  yes
  Cipher List:                 ECDH:!RC4 inherited from offloadBase
  ECC Curve List:              secp384r1 inherited from offloadBase
  SSL Session Cache Mode:      auto size default
  SSL Session Cache Size:      10 Mi default
  SSL Session Tickets Mode:    enabled default
Ordered cipher list
  ECDHE-ECDSA-AES256-SHA384
  ECDHE-ECDSA-AES256-SHA
  ECDHE-ECDSA-DES-CBC3-SHA
  ECDHE-ECDSA-AES128-SHA256
  ECDHE-ECDSA-AES128-SHA
```

Looks great! Let's now attach the SSL Profile to our VIP so that SSL/TLS Offloading may begin.

Enable SSL Offloading: Attach the SSL Profile to a Virtual IP

In order to have the SSL Profile begin to terminate SSL sessions on the LineRate System, it must be attached to a Virtual IP. This is simply done by running the following commands:

```
LROS(config)# virtual-ip mainServer
*LROS(config-virtual-ip:mainServer)# attach ssl profile offload
```

While we are here, let's update the VIP to listen on the default HTTPS port (versus the currently configured HTTP default port). In the current configuration, the LineRate System will start processing SSL requests that come into https://10.10.11.11:80, but this isn't as nice for native SSL traffic.

```
*LROS(config-virtual-ip:mainServer)# ip address 10.10.11.11 443
```

One more update: Updating the Virtual Server hostname

Our SSL certificates specify that they will serve secure content to a domain of sslloffload.lineratesystems.com. Let's update the Virtual Server to process requests coming into this domain:

```
LROS(config)# virtual-server ProdEnv
*LROS(config-vserver:ProdEnv)# service http
*LROS(config-vserver-http:ProdEnv)# hostname sslloffload.lineratesystems.com
```

And, finally, commit all changes to memory!

```
*LROS(config)# write
Building configuration...
[OK]
LROS(config)#
```

The LineRate system is now taking SSL/TLS requests going to https://sslloffload.lineratesystems.com terminating them. The requests are forwarded via HTTP to the internal application server. Any responses are then encrypted by LineRate and sent back to the end-user. To confirm this works as intended, let's run a few experiments:

Consideration: Recommended Header Insertion

In an environment where the LineRate system is communicating to real servers via an unencrypted protocol (i.e. HTTP), the application processing the requests may attempt to redirect traffic to a secure channel. Therefore, it is recommended that the *X-Forwarded-Proto: https* header is added to Real Server requests so that unnecessary redirects are avoided. Though out-of-scope of this article, a simple Node.js script can be written to add the following header to HTTP requests proxied to the Real Servers.

Verify the Configuration

Thus far, you should have a good understanding of Elliptic Curve Cryptography and Perfect Forward Secrecy and why it is important to your organization. An SSL Offload system has now been successfully implemented as well. However, maybe you are curious if ECC and PFS is actually functioning properly in the SSL session setup. Or perhaps you are not convinced that SSL Offloading is actually happening on the LineRate system itself but rather is being passed through to the Web Server. Next week, a demonstration on how to verify a correct implementation of SSL with ECC+PFS on LineRate will make a debut on DevCentral. The article will detail how to check for ECC SSL on the wire via WireShark and in the browser.

In case you missed any content, or would like to reference it again, here are the articles related to implementing SSL Offload with ECC and PFS on LineRate:

- Why ECC and PFS Matter: SSL offloading with LineRate
- Implementing ECC+PFS on LineRate (Part 1/3): Choosing ECC Curves and Preparing SSL Certificates
- **Implementing ECC+PFS on LineRate (Part 2/3): Configuring SSL Offload on LineRate**
- Implementing ECC+PFS on LineRate (Part 3/3): Confirming the Operation of SSL Offloading

Move over RSA: ECC crypto is here to stay! From the demonstration thus far, it is easy to see that LineRate is a great way to quickly and easily deploy better performance and security with SSL. Take LineRate and test out its SSL Offloading capabilities for a spin!

Ready to try LineRate? Visit https://linerate.f5.com/try
Want to learn more about LineRate? Visit https://linerate.f5.com/learn

F5 Networks, Inc. | 401 Elliot Avenue West, Seattle, WA 98119 | 888-882-4447 | f5.com

F5 Networks, Inc.
Corporate Headquarters
info@f5.com

F5 Networks
Asia-Pacific
apacinfo@f5.com

F5 Networks Ltd.
Europe/Middle-East/Africa
emeainfo@f5.com

F5 Networks
Japan K.K.
f5j-info@f5.com