Adding TLS to OpenLI

OpenLI Training: Chapter Eighteen

Shane Alcock
University of Waikato
New Zealand
shane.alcock@waikato.ac.nz

Benefits of TLS

- OpenLI communications contain sensitive information
    - Intercept target identities
    - IP addresses for core internal infrastructure
    - Intercepted customer communications

Benefits of TLS

- OpenLI communications should be internal
  - But still vulnerable to inspection from insider threats

Example

- Stop the collector service on collector container

```
/home/openli-coll#  stop_collector.sh
```

Example

- Run tracepktdump on eth1 on your provisioner container

```
/home/openli-prov#  tracepktdump ring:eth1
```

Example

- Now restart the collector

```
/home/openli-coll#  service openli-collector start
```

Example

- Look at the second-to-last packet in the tracepktdump output

```
...
unknown protocol tcp/8080
 Unknown Protocol: 8080
  5c 4c 6c 5c 00 1f 00 06 00 00 00 00 00 00 00 00 \Ll\............
  00 00 00 04 01 00 00 00 00 01 00 0a 31 37 32 2e ............172.
  31 39 2e 30 2e 33 00 02 00 05 31 32 30 30 31 5c 19.0.3....12001\
  4c 6c 5c 00 20 00 12 00 00 00 00 00 00 00 00 00 Ll\. ...........
  0e 00 01 01 00 0f 00 0f 32 30 33 2e 31 32 32 2e ........203.122.
  32 35 35 2e 31 34 30 00 10 00 04 31 36 34 35 5c 255.140....1645\
  4c 6c 5c 00 20 00 12 00 00 00 00 00 00 00 00 00 Ll\. ...........
  0e 00 01 01 00 0f 00 0f 32 30 33 2e 31 32 32 2e ........203.122.
  32 35 35 2e 31 34 30 00 10 00 04 31 36 34 36 5c 255.140....1646\
  4c 6c 5c 00 1d 00 12 00 00 00 00 00 00 00 00 00 Ll\.............
  0e 00 01 03 00 0f 00 0c 31 30 2e 31 30 30 2e 35 ........10.100.5
  30 2e 36 35 00 10 00 04 35 30 36 30 5c 4c 6c 5c 0.65....5060\Ll\
  00 5f 00 02 00 00 00 00 00 00 00 00 00 05 00 09 ._..............
  53 54 41 54 49 43 30 30 32 00 06 00 02 4e 5a 00 STATIC002....NZ.
  07 00 02 4e 5a 00 03 00 07 73 61 6c 63 6f 63 6b ...NZ....salcock
  00 09 00 07 6d 6f 63 6b 6c 65 61 00 14 00 04 06 ....mocklea.....
...
```

Example

- Sensitive information relayed in plain text
    - Usernames
    - IP addresses of servers
    - Agency IDs

Benefits of TLS

- Encryption with TLS offers extra protection
  - Even if OpenLI instructions are seen, they are not readable
    - (assuming you secure the encryption keys!)

SSL Certificates

- Required to enable encryption in OpenLI
  - Allows components to trust each other
  - Establish an encrypted channel for communication

- You will need one certificate per component

Generating SSL Certificates

- For a real deployment…
  - Generate a Certificate Signing Request on your OpenLI component
  - Pass the CSR on to a trusted CA for signing
    - Pay the required fee ($$)
  - Install received certificate on your OpenLI component

Generating SSL Certificates

- Let’s Encrypt is also an option
  - Remember that your OpenLI components are internal
  - HTTP challenge won’t work in that case
  - DNS challenge + adding a TXT record for your components
    - Exercise left to the deployer

Generating SSL Certificates

- Self-signed certificates
    - Create your own untrusted CA and sign certs yourself
        - OK for solely internal use
        - Tools and browsers will complain

- For simplicity, we’re using self-signed for the training lab
    - Consider the other options for your real deployment
    - Otherwise, use at your own risk!

SSL Certificates for the Lab

- I’ve already generated certificates for the lab containers
  - Also created a corresponding CA certificate
  - DO NOT use these in production!

- For a real deployment, you’ll need to:
  - Create and sign your own certificates
  - Copy them onto your component hosts
  - Set appropriate permissions to secure them

Enabling TLS

- To enable TLS, we just need to update OpenLI config
  - Let’s start with the provisioner

Enabling TLS

- Open up the provisioner config file in your text editor
  - Make sure you're logged in to the provisioner container

```
/home/openli-prov# vim /etc/openli/provisioner-config.yaml
```

Enabling TLS

- Look for the `tlscert`, `tlskey` and `tlsca` options

```
#tlscert: <TLSCERT>
#tlskey: <TLSKEY>
#tlsca: <TLSCA>
```

Enabling TLS

- Update tlscert with the signed certificate for the provisioner
    - This is located in /etc/openli/ssl/provisioner-crt.pem

```
#tlscert: /etc/openli/ssl/provisioner-crt.pem
#tlskey: <TLSKEY>
#tlsca: <TLSCA>
```

Enabling TLS

- Update tlskey with the private key for the provisioner certificate
    - This is located in /etc/openli/ssl/provisioner-key.pem

```
#tlscert: /etc/openli/ssl/provisioner-crt.pem
#tlskey: /etc/openli/ssl/provisioner-key.pem
#tlsca: <TLSCA>
```

Enabling TLS

- Update tlsca with the certificate for the issuing CA
  - This is located in /etc/openli/ssl/ca-crt.pem

```
#tlscert: /etc/openli/ssl/provisioner-crt.pem
#tlskey: /etc/openli/ssl/provisioner-key.pem
#tlsca: /etc/openli/ssl/ca-crt.pem
```

Enabling TLS

- Uncomment the TLS options so that they are applied

```
tlscert: /etc/openli/ssl/provisioner-crt.pem
tlskey: /etc/openli/ssl/provisioner-key.pem
tlsca: /etc/openli/ssl/ca-crt.pem
```

Enabling TLS

- Restart your provisioner and check the logs

```
# stop_provisioner.sh

# service openli-provisioner start

# less /var/log/openli/provisioner.log
```

Failure??

- These error messages don’t look great

```
openliprovisioner[567]: OpenLI: SSL Handshake failed for collector 172.19.0.4-51862
openliprovisioner[567]: OpenLI: SSL handshake for mediator 172.19.0.3-58354 is pending...
openliprovisioner[567]: OpenLI: Pending SSL Handshake for mediator 172.19.0.3-58354 failed
```

Next Time

- Fixing the errors!
  - Complete our deployment of TLS throughout OpenLI
  - Confirm that our messages are now encrypted
  - Use the new HTTPS version of the REST API