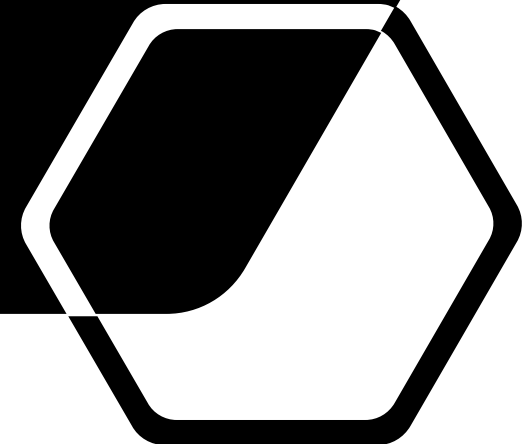
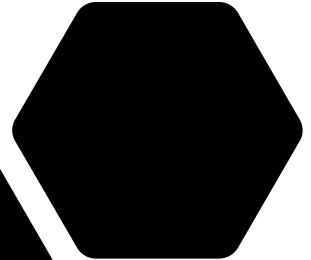


EPICS Security Technical Plan



Osprey DCS
George McIntyre

- A presentation of the implementation plan for EPIC Security, being carried out by SLAC, Osprey DCS and ORNL





Agenda

- **Planned Features**

- Implementation of basic TLS
- Certificate Management
- TLS Session Status & Management
- Enhanced Client Authorization

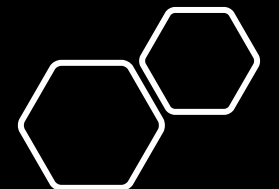
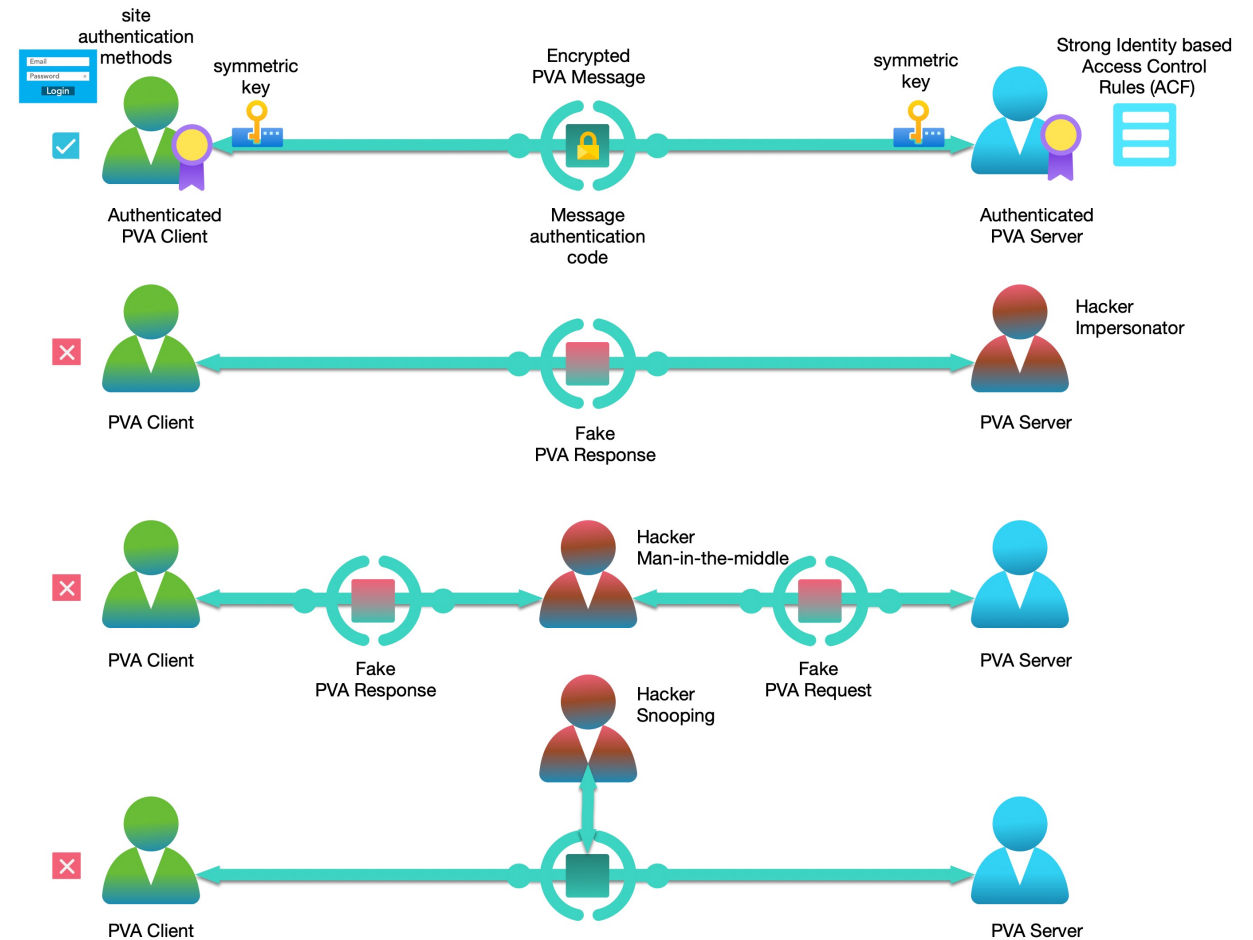
- **TLS Demo**

- Manually Create Certificates
- Configure Wireshark
- Non-secure PVA communications
- Configure secure PVA communications
- Demo Secure PVA communications

What this implementation will get us?

Benefits

- **Server Certificates** →
 - **Authentication:**
 - Prevent **Service Impersonation**
 - Prevent **Man-in-the-Middle** attacks
 - **PVA Message Encryption** →
 - Cipher suite **Message Authentication Codes** →
 - Guarantee **Data Integrity**
 - Securely shared **Symmetric Session Keys** →
 - Prevent **Packet Snooping**
 - **Client Certificates & Site Authentication integration** →
 - **Authorization:**
 - Provide a mechanism for **Service Access Control**
 - **Protect Data** by allowing Services to Restrict Access



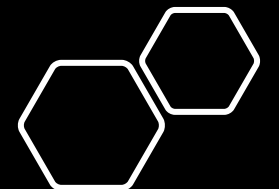
What this implementation will get us?

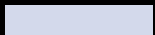
Benefits

- **Server Certificates** →
 - **Authentication:**
 - Prevent **Service Impersonation**
 - Prevent **Man-in-the-Middle** attacks
 - **PVA Message Encryption** →
 - Cipher suite **Message Authentication Codes** →
 - Guarantee **Data Integrity**
 - Securely shared **Symmetric Session Keys** →
 - Prevent **Packet Snooping**
 - **Client Certificates & Site Authentication integration** →
 - **Authorization:**
 - Provide a mechanism for **Service Access Control**
 - **Protect Data** by allowing Services to Restrict Access

Will Not

- Prevent **PV Impersonation** in a mixed TLS/TCP network
- Prevent discovery of **Service Endpoint** or **PV name**
- Prevent discovery of **Encryption Type**
- Prevent discovery of **Data Transmission Frequency**
- Prevent discovery of approximate **Amount of data transmitted**
- **Change Site Security Policies** – you need to implement these on top of the technical solution presented here



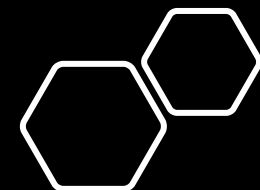


Basic TLS Implementation



Basic TLS Implementation

- Allow “tls” in search request
- Allow “tls” in search response
- Initiate TLS handshake if server certificate is configured
- Pass client certificate in handshake if client certificate is configured
- Encapsulate and encrypt PVA protocol messages
- Maintain backwards compatibility



TLS in EPICS



Programmatic Interface changes

Possibility of specifying "tls" as a protocol

New PV Access Server configuration options for TLS



Network Management Impact

Install Server Certificates

Note: TLS and legacy clients/servers use the same ports and may interoperate on the same network without any changes to legacy clients and servers



New EPICS Features

Verified identity of EPICS channel servers

Guaranteed data integrity for EPICS services/channels

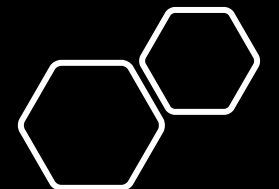
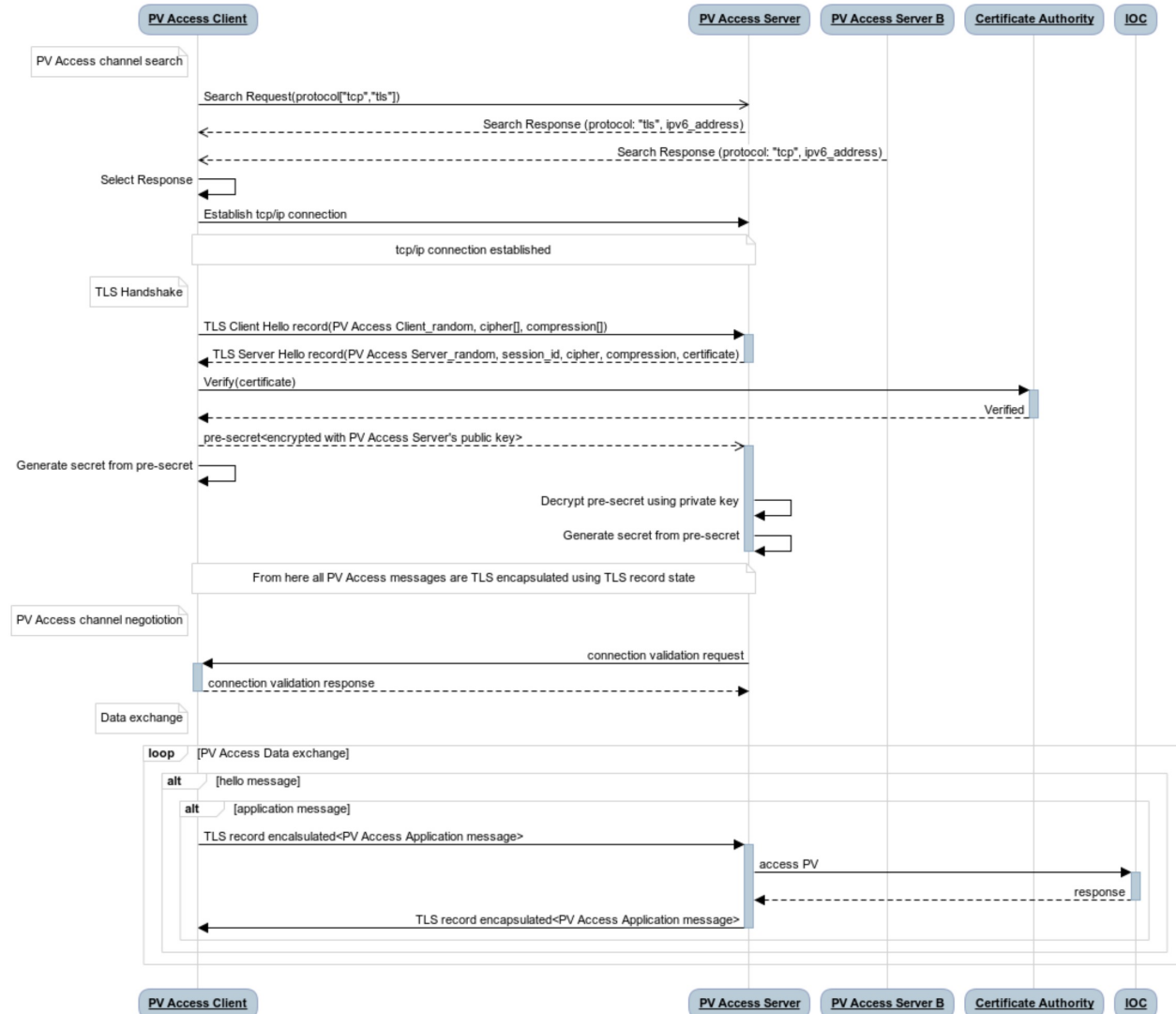
Encrypted EPICS data packets



Protocol Changes

New client-initiated TLS handshake phase before connection validation

Encapsulation of PV Access Messages



Implementation repositories

Java implementation – *maintainer Kay Kasemir*

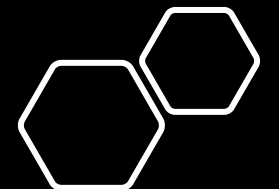
- <https://github.com/ControlSystemStudio/phoebus/tree/master/core/pva>

C++ implementation – *maintainer Michael Davidsaver*

- <https://mdavidsaver.github.io/pvxs> branch TLS

Documentation

- <https://github.com/epics-base/pvAccessCPP/wiki/protocol>



Out of scope

Features

Add TLS to Channel Access

UDP Broadcast search

UDP response

Beacon messages

Add additional TLS beacon messages for servers supporting both TLS and TCP

Any changes to support TLS in Gateways

Any changes to support TLS in EPICS Python (pvaPy)

Any changes to support TLS in PV Database

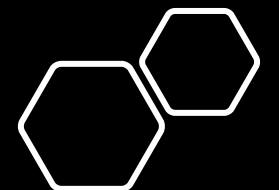
Repositories

EPICS base Java

- <https://github.com/epics-base/epicsCoreJava>
- <https://github.com/epics-base/pvaClientJava>

EPICS base C++

- <https://github.com/epics-base/pvAccessCPP>
- <https://github.com/epics-base/pvaClientCPP>



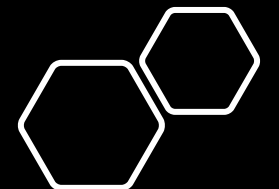
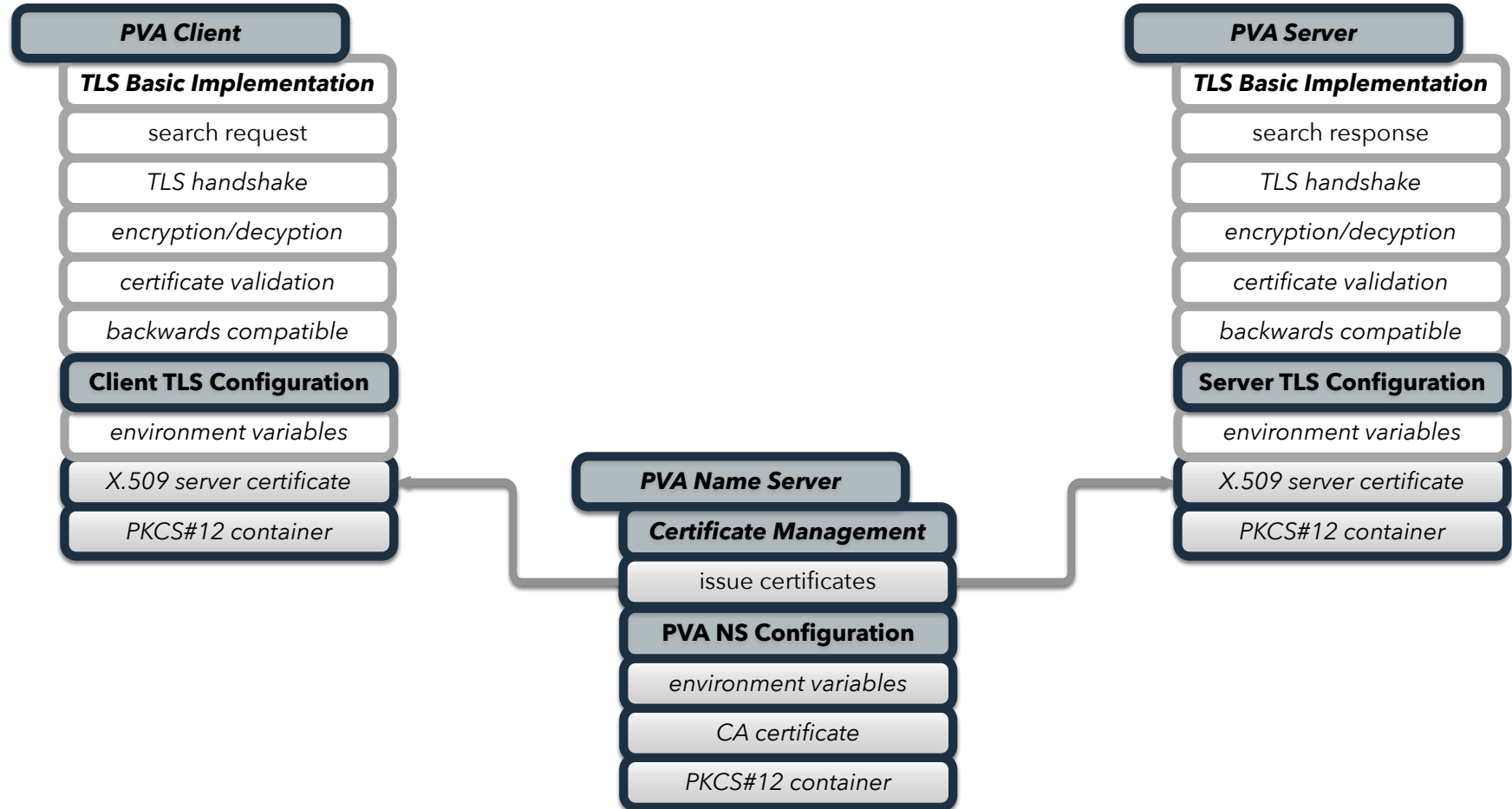


Certificate Management



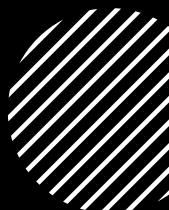
Certificate Management

- Update PVA Name Server to issue client and server certificates
- Manage Site Certificate Authorities





Certificate Management



Manage Site Certificate Authority

- Secure Storage for Site CA
- Management of Private Key



Issue and Distribute Server and Client Certificates

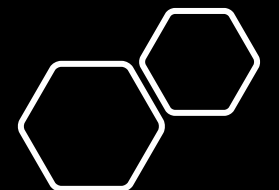
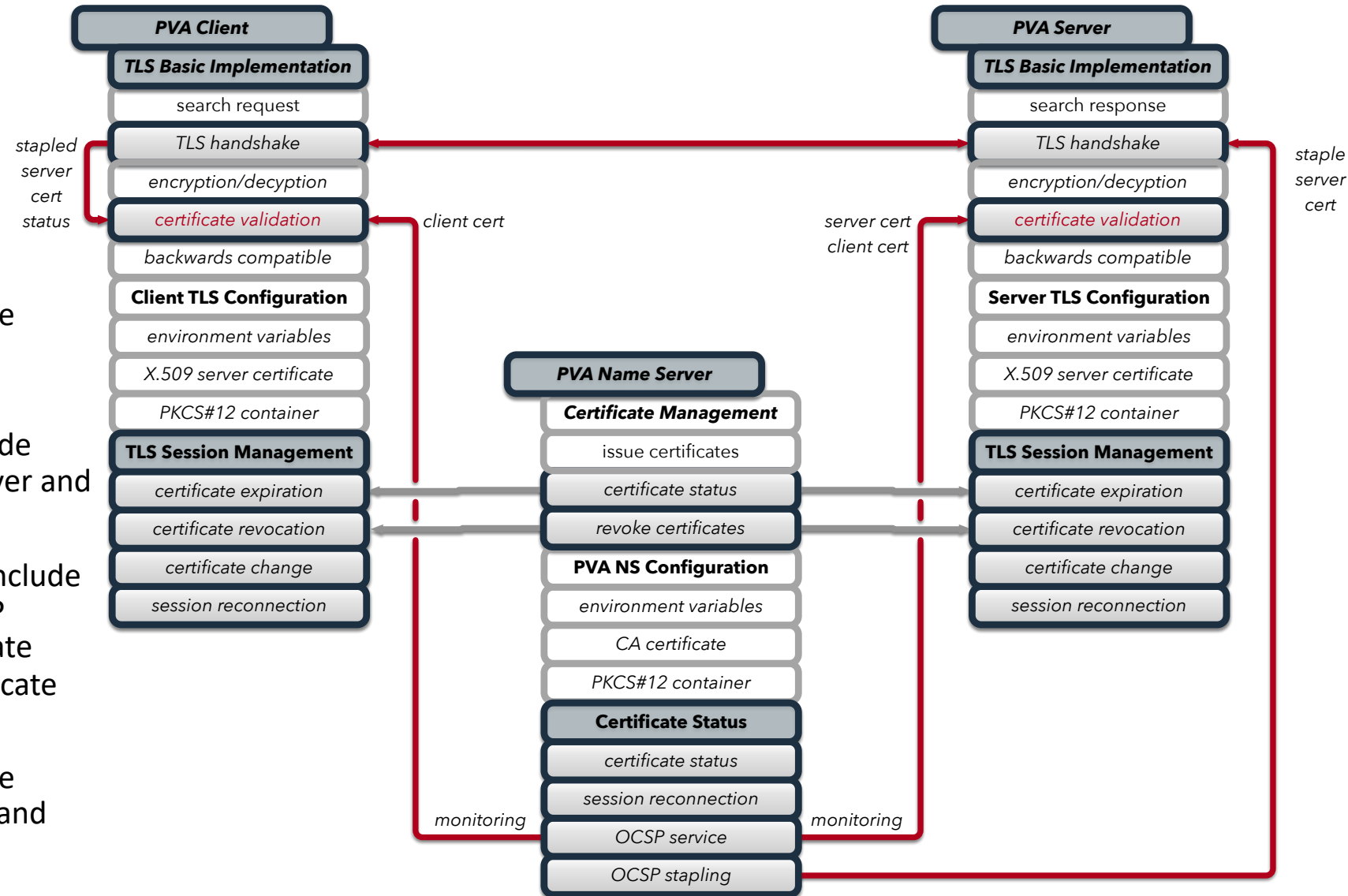
- Create Certificate Key Pairs
- Create Signing Request
- Sign Certificates with CA
- Deliver server and client certificates



Session Status and Management

Session Status and Management

- Update Client and Server to handle reconnections after session disconnections
- Update PVA Name Server to provide certificate status via OCSP for Server and Client's certificate validation
- Update server TLS handshake to include server certificate status with OCSP stapling and update client certificate validation to use this server certificate status
- Update Client and Server to handle certificate expiration, revocation, and change



Session Status & Management



Certificate Status

Implement OCSP server to deliver certificate status

Allow pub/sub event model for server status monitoring

Client and Server use this service to monitor for their own status changes



Revocation

Implements a function that revokes a certificate by setting the appropriate status

Clients and servers respond appropriately to revoked status



Expiration

Recognizes certificate expiration

Alerts listeners to upcoming expirations

Clients and Servers respond appropriately when certificates expire



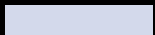
Rotation

Allow orderly rotation of valid certificates with new valid certificates

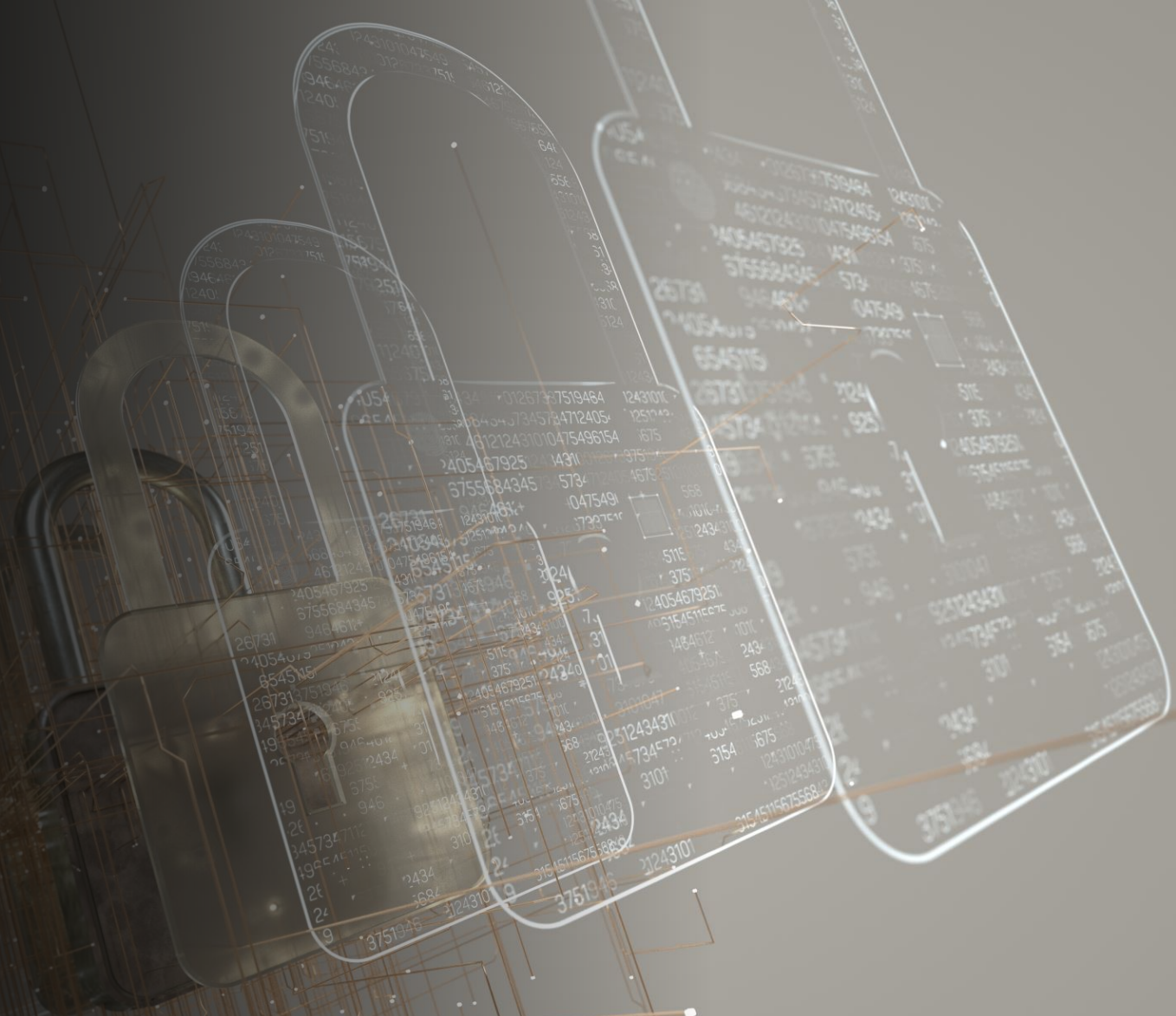


Stapling

Append server certificate status to TLS handshake with the OCSP stapling extension so that clients don't have to verify the status

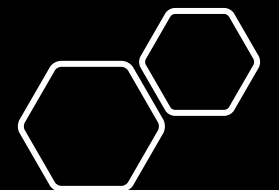
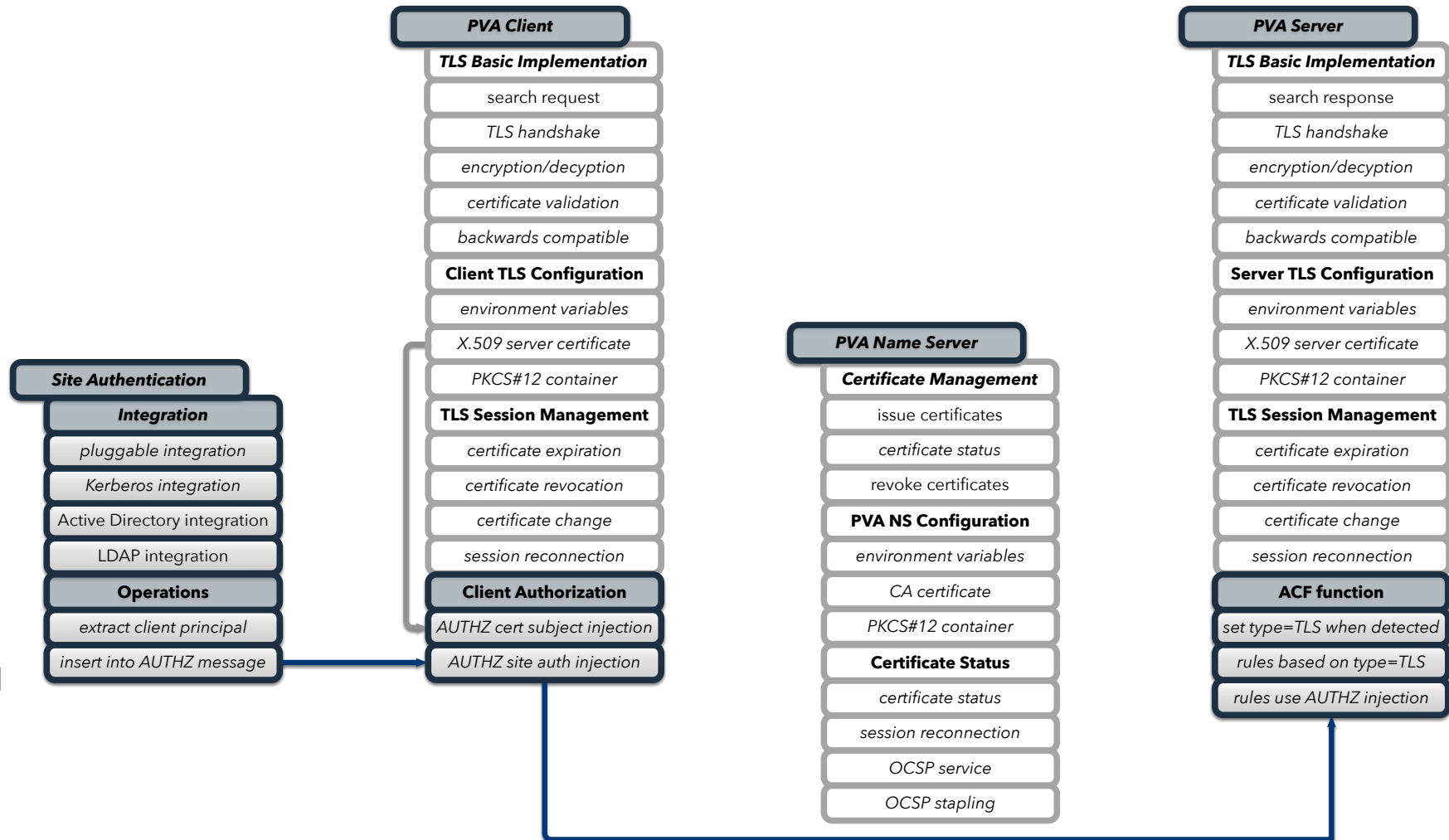


Enhanced Client Authorization

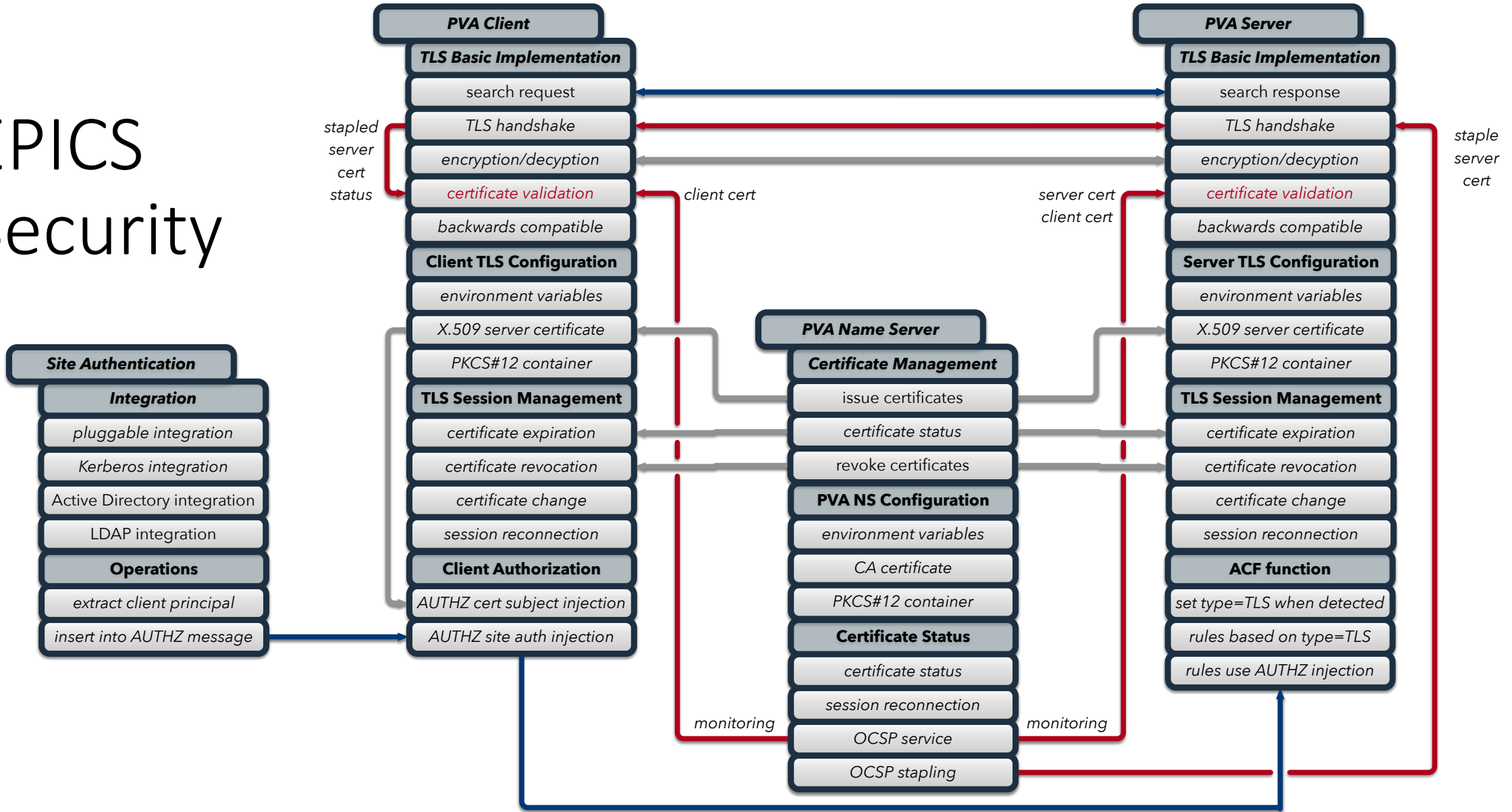


Enhanced Client Authorization

- Integrate with common site authentication methods
 - Kerberos
 - LDAP
 - Active Directory
- In PVA Connection validation message inject into the AUTHZ name field either the certificate subject or principal from site authentication integration
- Update ACF function that controls authorization so that it can use TLS status to control access

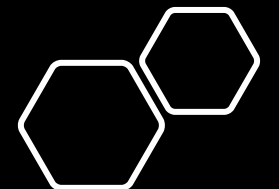


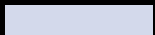
EPICS Security



Recap

1. **Basic TLS** to authenticate and encrypt
2. **Certificate Management** to issue and distribute certificates
3. **Session Management** to handle expiration, revocation, recycling, and reconnection
4. **Authorization** using site authentication or client certificates





TLS Demo

Manually create certificates

CA Certificate

```
openssl genpkey -algorithm RSA -out ca.key
# create ca_config.cnf
openssl req -x509 -new -nodes -key ca.key -sha256 -days 3650 -out ca.crt -config ca_config.cnf
```

```
ca_config.cnf ...
keyUsage = digitalSignature, cRLSign, keyCertSign
authorityKeyIdentifier = keyid:always,issuer
basicConstraints = CA:TRUE
```

Server Certificate

```
openssl genpkey -algorithm RSA -out server.key
# create server_config.cnf
openssl req -new -key server.key -out server.csr -config server_config.cnf
openssl x509 -req -in server.csr -CA ca.crt -CAkey ca.key -CAcreateserial -out server.crt -days 365 -sha256
-extfile server_config.cnf -extensions v3_req
openssl pkcs12 -export -out server.p12 -inkey server.key -in server.crt -certfile ca.crt
```

```
server_config.cnf ...
keyUsage = digitalSignature, keyEncipherment
extendedKeyUsage = serverAuth
```

Client Certificate

```
openssl genpkey -algorithm RSA -out client.key
# create client_config.cnf
openssl req -new -key client.key -out client.csr -config client_config.cnf
openssl x509 -req -in client.csr -CA ca.crt -CAkey ca.key -CAcreateserial -out client.crt -days 365 -sha256
-extfile client_config.cnf -extensions v3_req
openssl pkcs12 -export -out client.p12 -inkey client.key -in client.crt -certfile ca.crt
```

```
client_config.cnf ...
keyUsage = digitalSignature, keyEncipherment
extendedKeyUsage = clientAuth
```

Use
KeyStore
explorer to
view
certificates

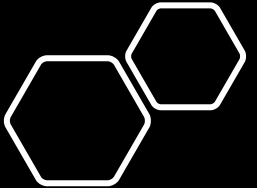
The screenshot shows the KeyStore Explorer 5.5.2 application window. The main window displays a table with one entry:

Entry Name	Algorithm	Key Size	Certificate Expiry	Last Modified
1	RSA	2048	07/09/2024, 14:54:36 CEST	-

A dialog box titled "Certificate Details for Entry '1'" is open, showing the following details:

- Certificate Hierarchy: level-n.com, mcinpro.level-n.com
- Version: 3
- Subject: C=GB,ST=Gloucestershire,L=Lydney,O=Level N Ltd,CN=mcinpro.level-n.com
- Issuer: CN=level-n.com
- Serial Number (hex.): 0x1F0EC7D5434FD942065C58431AB88B3B801F74FA
- Serial Number (dec.): 177308332242835902107509237383924524607269205242
- Valid From: 08/09/2023, 14:54:36 CEST
- Valid Until: 07/09/2024, 14:54:36 CEST
- Public Key: RSA 2048 bits
- Signature Algorithm: SHA-256 with RSA
- Fingerprint: SHA-1, DB:40:E4:62:CB:32:3D:D4:41:B3:8A:A6:86:A3:4A:D0:C6

Buttons at the bottom of the dialog include: Export, Extensions, PEM, Verify, ASN.1, and OK.



Configure Wireshark

Set up LUA scripts

```
cd ~/.config/wireshark/plugins/
```

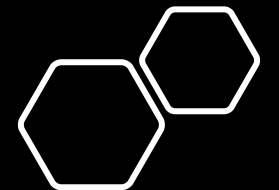
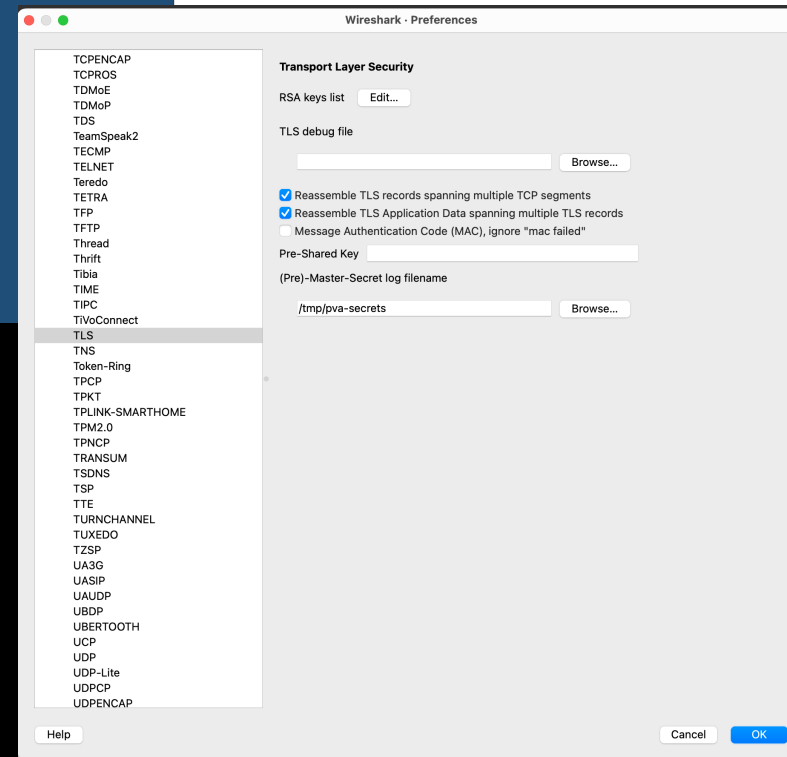
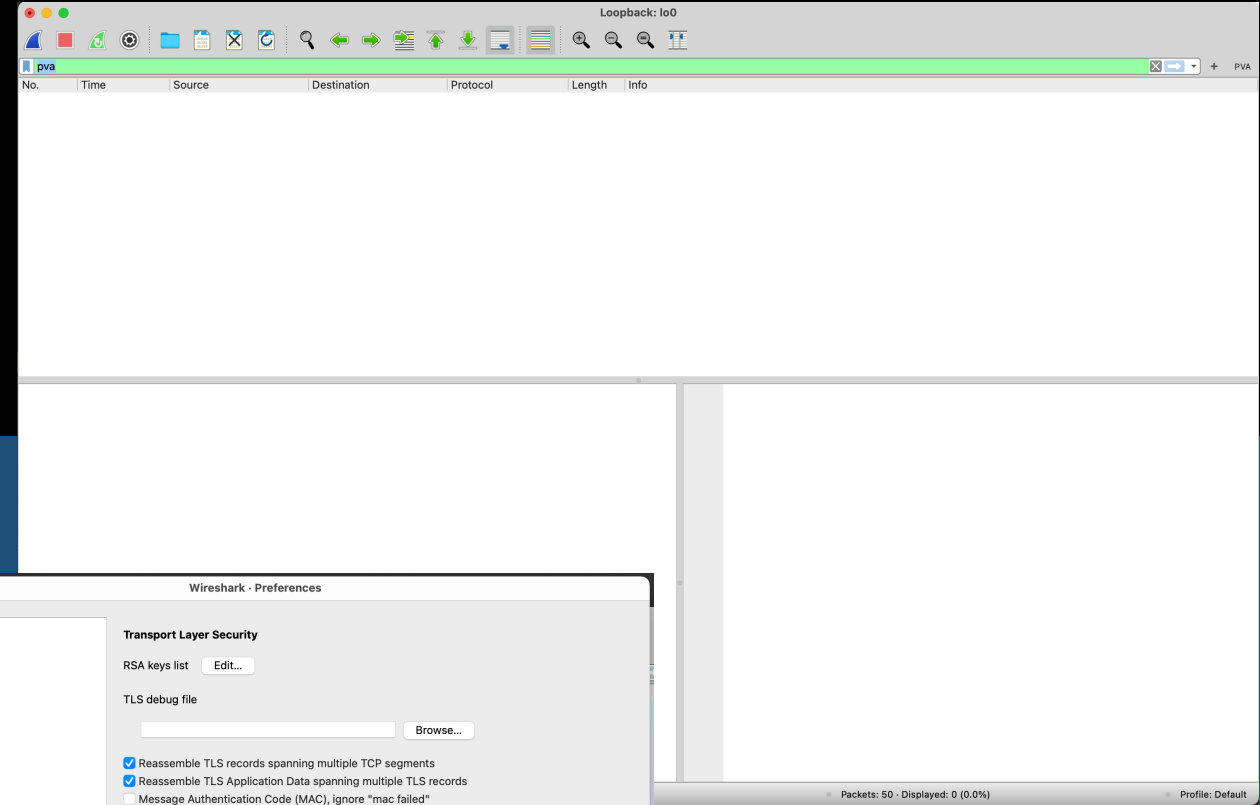
```
ln -s ~/Projects/com/osprey-dcs/cashark/*.lua
```

Set up key log file to capture session keys

```
export SSLKEYLOGFILE=/tmp/pva-secrets
```

Set display filter

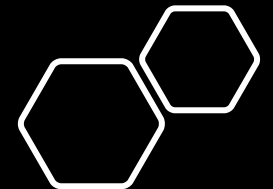
```
pva
```



Non-Secure PVA Communications

Start IOC server

```
softIocPVX -v -m user=test,N=tst,P=tst -d test/testioc.db -d test/testiocg.db -a test/testioc.acf
dbLoadDatabase("/Users/george/Projects/com/osprey-dcs/pvxs/bin/darwin-aarch64/../../dbd/softIocPVX.dbd")
softIocPVX_registerRecordDeviceDriver(pdbbase)
dbLoadRecords("test/testioc.db", "user=test,N=tst,P=tst")
dbLoadRecords("test/testiocg.db", "user=test,N=tst,P=tst")
asSetSubstitutions("user=test,N=tst,P=tst")
asSetFilename("test/testioc.acf")
asSetFilename: Warning - relative paths won't usually work
iocInit()
INFO: PVXS QSRV2 is loaded and ENABLED.
Starting iocInit
#####
## EPICS R7.0.7.1-DEV
## Rev. R7.0.7-56-g718da5c9be96b7eccd7c
## Rev. Date Git: 2023-02-04 22:56:19 -0600
#####
epics>
```



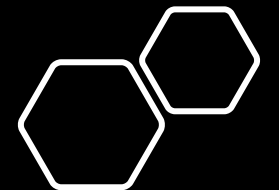
Non-Secure PVA Communications

Get a PV

```
pvxget test:calcExample
```

```
test:calcExample
```

```
value.double = 4
alarm.severity.int32_t = 1
alarm.status.int32_t = 1
alarm.message.string = "LOW"
timeStamp.secondsPastEpoch.int64_t = 1695756172
timeStamp.nanoseconds.int32_t = 258063000
timeStamp.userTag.int32_t = 0
display.limitLow.double = 0
display.limitHigh.double = 10
display.description.string = "Counter"
display.units.string = "Counts"
display.form.index.int32_t = 0
display.form.choices.string[] = {7}["Default", "String", "Binary", "Decimal", "Hex", "Exponential", "Engineering"]
control.limitLow.double = 0
control.limitHigh.double = 10
valueAlarm.lowAlarmLimit.double = 2
valueAlarm.lowWarningLimit.double = 4
valueAlarm.highWarningLimit.double = 6
valueAlarm.highAlarmLimit.double = 8
```



Capturing from Loopback: lo0

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.2.3	192.168.2.255	PVA	94	60009 -> 5076 Client SEARCH(1, 270544961:'test:calcExample'),
2	0.000974	192.168.2.3	192.168.2.3	PVA	85	5076 -> 60009 Server SEARCH_RESPONSE(1, 270544961)
7	0.001956	192.168.2.3	192.168.2.3	PVA	92	5075 -> 63068 Server SET_BYTE_ORDER, CONNECTION_VALIDATION,
9	0.003790	192.168.2.3	192.168.2.3	PVA	114	63068 -> 5075 Client CONNECTION_VALIDATION,
11	0.003880	192.168.2.3	192.168.2.3	PVA	65	5075 -> 63068 Server CONNECTION_VALIDATED,
13	0.003945	192.168.2.3	192.168.2.3	PVA	87	63068 -> 5075 Client CREATE_CHANNEL('test:calcExample'),
15	0.004042	192.168.2.3	192.168.2.3	PVA	73	5075 -> 63068 Server CREATE_CHANNEL(cid=270544961, sid=117768961),
17	0.004118	192.168.2.3	192.168.2.3	PVA	79	63068 -> 5075 Client GET(sid=117768961, ioid=2154848337, sub=08),
19	0.004208	192.168.2.3	192.168.2.3	PVA	515	5075 -> 63068 Server GET(ioid=2154848337, sub=08),
21	0.004375	192.168.2.3	192.168.2.3	PVA	73	63068 -> 5075 Client GET(sid=117768961, ioid=2154848337, sub=08),
23	0.004518	192.168.2.3	192.168.2.3	PVA	254	5075 -> 63068 Server GET(ioid=2154848337, sub=08),
29	0.267694	192.168.2.3	192.168.2.3	PVA	85	5076 -> 60009 Server SEARCH_RESPONSE(1, 270544961)
50	21.045525	192.168.2.3	192.168.2.255	PVA	79	50803 -> 5075 Server BEACON(0x734deb58ba42096376b0ee66,

Frame 1: 94 bytes on wire (752 bits), 94 bytes captured (752 bits) on Null/Loopback
 Internet Protocol Version 4, Src: 192.168.2.3, Dst: 192.168.2.255
 User Datagram Protocol, Src Port: 60009, Dst Port: 5076
 Process Variable Access
 Magic: 0xca
 Version: 2
 Flags: 0x00
 Command: SEARCH (0x03)
 Size: 54
 Search Sequence #: 1
 Mask: 0x00
 Address: 00000000000000000000000000000000ffff00000000
 Port: 60009
 Transport Protocol: tcp
 PV Count: 1
 CID: 270544961
 Name: test:calcExample

Loopback: lo0: <live capture in progress> Packets: 94 · Displayed: 13 (13.8%)

Capturing from Loopback: lo0

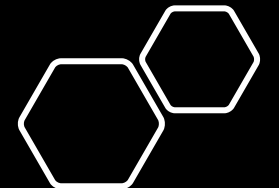
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.2.3	192.168.2.255	PVA	94	60009 -> 5076 Client SEARCH(1, 270544961:'test:calcExample'),
2	0.000974	192.168.2.3	192.168.2.3	PVA	85	5076 -> 60009 Server SEARCH_RESPONSE(1, 270544961)
7	0.001956	192.168.2.3	192.168.2.3	PVA	92	5075 -> 63068 Server SET_BYTE_ORDER, CONNECTION_VALIDATION,
9	0.003790	192.168.2.3	192.168.2.3	PVA	114	63068 -> 5075 Client CONNECTION_VALIDATION,
11	0.003880	192.168.2.3	192.168.2.3	PVA	65	5075 -> 63068 Server CONNECTION_VALIDATED,
13	0.003945	192.168.2.3	192.168.2.3	PVA	87	63068 -> 5075 Client CREATE_CHANNEL('test:calcExample'),
15	0.004042	192.168.2.3	192.168.2.3	PVA	73	5075 -> 63068 Server CREATE_CHANNEL(cid=270544961, sid=117768961),
17	0.004118	192.168.2.3	192.168.2.3	PVA	79	63068 -> 5075 Client GET(sid=117768961, ioid=2154848337, sub=08),
19	0.004208	192.168.2.3	192.168.2.3	PVA	515	5075 -> 63068 Server GET(ioid=2154848337, sub=08),
21	0.004375	192.168.2.3	192.168.2.3	PVA	73	63068 -> 5075 Client GET(sid=117768961, ioid=2154848337, sub=08),
23	0.004518	192.168.2.3	192.168.2.3	PVA	254	5075 -> 63068 Server GET(ioid=2154848337, sub=08),
29	0.267694	192.168.2.3	192.168.2.3	PVA	85	5076 -> 60009 Server SEARCH_RESPONSE(1, 270544961)
50	21.045525	192.168.2.3	192.168.2.255	PVA	79	50803 -> 5075 Server BEACON(0x734deb58ba42096376b0ee66, 16, 2)

Frame 2: 85 bytes on wire (680 bits), 85 bytes captured (680 bits) on Null/Loopback
 Internet Protocol Version 4, Src: 192.168.2.3, Dst: 192.168.2.3
 User Datagram Protocol, Src Port: 5076, Dst Port: 60009
 Process Variable Access
 Magic: 0xca
 Version: 2
 Flags: 0xc0
 Command: SEARCH_RESPONSE (0x04)
 Size: 45
 GUID: 734deb58ba42096376b0ee66
 Search Sequence #: 1
 Address: 00000000000000000000000000000000ffff00000000
 Port: 5075
 Transport Protocol: tcp
 Found: True
 CID: 270544961

Loopback: lo0: <live capture in progress> Packets: 95 · Displayed: 13 (13.7%) Profile: Default

View Non-Secure Results in Wireshark

- Only **tcp** is included in search message
- No **TLS Handshake** takes place



Capturing from Loopback: lo0

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.2.3	192.168.2.255	PVA	94	60009 -> 5076 Client SEARCH(1, 270544961: 'test:calcExample'),
2	0.000974	192.168.2.3	192.168.2.3	PVA	85	5076 -> 60009 Server SEARCH_RESPONSE(1, 270544961)

Frame 9: 114 bytes on wire (912 bits), 114 bytes captured (912 bits) on interface Loopback
 Internet Protocol Version 4, Src: 192.168.2.3, Dst: 192.168.2.3
 Transmission Control Protocol, Src Port: 63068, Dst Port: 5075, Seq: 114
 Process Variable Access
 Magic: 0xca
 Version: 2
 Flags: 0x00
 Command: CONNECTION_VALIDATION (0x01)
 Size: 50
 Client Queue Size: 146988
 Client Introspection registry size: 32767
 Client QoS: 0x0000
 AuthZ name: ca
 Data: fd010080000204757365726004686f7374600667656f7267650d4d634

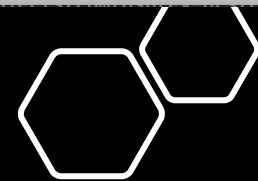
Capturing from Loopback: lo0

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.2.3	192.168.2.255	PVA	94	60009 -> 5076 Client SEARCH(1, 270544961: 'test:calcExample'),
2	0.000974	192.168.2.3	192.168.2.3	PVA	85	5076 -> 60009 Server SEARCH_RESPONSE(1, 270544961)

Frame 19: 515 bytes on wire (4120 bits), 515 bytes captured (4120 bits) on interface Loopback
 Internet Protocol Version 4, Src: 192.168.2.3, Dst: 192.168.2.3
 Transmission Control Protocol, Src Port: 5075, Dst Port: 63068, Seq: 515
 Process Variable Access
 Magic: 0xca
 Version: 2
 Flags: 0x40
 Command: GET (0x0a)
 Size: 451
 Operation ID: 2154848337
 Sub-command: 0x08
 Status: OK (0xff)
 Body: 801565706963733a6e742f4e545363616c61723a312e30060576616c7

View Non-Secure Results in Wireshark

- Only **tcp** is included in search message
- No **TLS Handshake** takes place

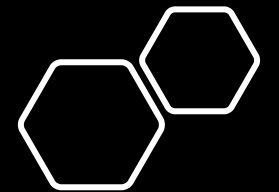


Secure PVA Communications

Configure TLS Client and Server environments

```
export EPICS_PVAS_TLS_KEYCHAIN="~/Projects/com/osprey-dcs/certificates/server.p12"
```

```
export EPICS_PVA_TLS_KEYCHAIN="~/Projects/com/osprey-dcs/certificates/client.p12"
```



Secure PVA Communications

Start IOC server

```
softIocPVX -v -m user=test,N=tst,P=tst -d test/testioc.db -d test/testiocg.db -a test/testioc.acf
dbLoadDatabase("/Users/george/Projects/com/osprey-dcs/pvxs/bin/darwin-aarch64/../../dbd/softIocPVX.dbd")
softIocPVX_registerRecordDeviceDriver(pdbbase)
NOTICE: debug logging TLS SECRETS to SSLKEYLOGFILE=/tmp/pva-secrets
dbLoadRecords("test/testioc.db", "user=test,N=tst,P=tst")
dbLoadRecords("test/testiocg.db", "user=test,N=tst,P=tst")
asSetSubstitutions("user=test,N=tst,P=tst")
asSetFilename("test/testioc.acf")
asSetFilename: Warning - relative paths won't usually work
iocInit()
INFO: PVXS QSRV2 is loaded and ENABLED.
Starting iocInit
#####
## EPICS R7.0.7.1-DEV
## Rev. R7.0.7-56-g718da5c9be96b7eccd7c
## Rev. Date Git: 2023-02-04 22:56:19 -0600
#####
epics>
```



Secure PVA Communications

Get a PV

```
pvxget test:calcExample
```

```
NOTICE: debug logging TLS SECRETS to SSLKEYLOGFILE=/tmp/pva-secrets
```

```
test:calcExample
```

```
value.double = 4
```

```
alarm.severity.int32_t = 1
```

```
alarm.status.int32_t = 1
```

```
alarm.message.string = "LOW"
```

```
timestamp.secondsPastEpoch.int64_t = 1695756172
```

```
timestamp.nanoseconds.int32_t = 258063000
```

```
timestamp.userTag.int32_t = 0
```

```
display.limitLow.double = 0
```

```
display.limitHigh.double = 10
```

```
display.description.string = "Counter"
```

```
display.units.string = "Counts"
```

```
display.form.index.int32_t = 0
```

```
display.form.choices.string[] = {7}["Default", "String", "Binary", "Decimal", "Hex", "Exponential", "Engineering"]
```

```
control.limitLow.double = 0
```

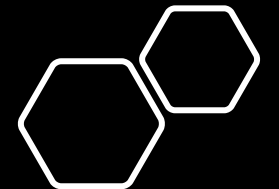
```
control.limitHigh.double = 10
```

```
valueAlarm.lowAlarmLimit.double = 2
```

```
valueAlarm.lowWarningLimit.double = 4
```

```
valueAlarm.highWarningLimit.double = 6
```

```
valueAlarm.highAlarmLimit.double = 8
```



Capturing from Loopback: lo0

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.2.3	192.168.2.255	PVA	98	51760 -> 5076 Client SEARCH(1718185572, 305419896:'test:calcExamp
2	0.000167	192.168.2.3	192.168.2.3	PVA	85	5076 -> 51760 Server SEARCH_RESPONSE(1718185572, 305419896)

Frame 1: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on Null/Loopback
 Internet Protocol Version 4, Src: 192.168.2.3, Dst: 192.168.2.255
 User Datagram Protocol, Src Port: 51760, Dst Port: 5076
 Process Variable Access
 Magic: 0xca
 Version: 2
 Flags: 0x00
 Command: SEARCH (0x03)
 Size: 58
 Search Sequence #: 1718185572
 Mask: 0x00
 Address: 00000000000000000000000000000000
 Port: 51760
Transport Protocol: tls
 Transport Protocol: tcp
 PV Count: 1
 CID: 305419896
 Name: test:calcExample

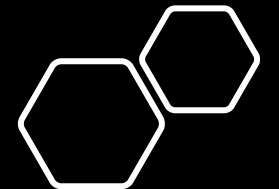
Capturing from Loopback: lo0

No.	Time	Source	Destination	Protocol	Length	Info
2	0.000167	192.168.2.3	192.168.2.3	PVA	85	5076 -> 51760 Server SEARCH_RESPONSE(1718185572, 305419896)
3	0.000553	192.168.2.3	192.168.2.3	PVA	2485	5076 -> 63108 Server SET_BYTE_ORDER, CONNECTION_VALIDATION,

Frame 2: 85 bytes on wire (680 bits), 85 bytes captured (680 bits) on Null/Loopback
 Internet Protocol Version 4, Src: 192.168.2.3, Dst: 192.168.2.3
 User Datagram Protocol, Src Port: 5076, Dst Port: 51760
 Process Variable Access
 Magic: 0xca
 Version: 2
 Flags: 0xc0
 Command: SEARCH_RESPONSE (0x04)
 Size: 45
 GUID: 501536078ea537da3c521a3a
 Search Sequence #: 1718185572
 Address: 000000000000000000000000ffff00000000
 Port: 5076
Transport Protocol: tls
 Found: True
 CID: 305419896

View Secure Results in Wireshark

- Both **tcp** and **tls** are included in search message
- **TLS Handshake** takes place
- PVA session is encrypted, and we see the decrypted message thanks to the key log.



Wireshark interface showing a capture from Loopback: lo0. The packet list pane shows a search request (No. 17) and its response (No. 19). The packet details pane for packet 17 shows the following information:

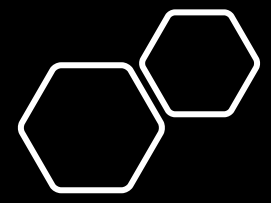
- Frame 17: 92 bytes on wire (736 bits), 92 bytes captured (736 bits) on Null/Loopback
- Internet Protocol Version 4, Src: 192.168.2.3, Dst: 192.168.2.3
- Transmission Control Protocol, Src Port: 63108, Dst Port: 5076, Seq: 117768961, Len: 92
- Transport Layer Security
 - [2 Reassembled TLS segments (22 bytes): #15(8), #17(14)]
 - Process Variable Access
 - Magic: 0xca
 - Version: 2
 - Flags: 0x00
 - Command: CONNECTION_VALIDATION (0x01)
 - Size: 14
 - Client Queue Size: 65536
 - Client Introspection registry size: 32767
 - Client QoS: 0x0000
 - AuthZ name: x509
 - Data: ff

Wireshark interface showing a capture from Loopback: lo0. The packet list pane shows a search request (No. 37) and its response (No. 41). The packet details pane for packet 37 shows the following information:

- Frame 37: 298 bytes on wire (2384 bits), 298 bytes captured (2384 bits) on Null/Loopback
- Internet Protocol Version 4, Src: 192.168.2.3, Dst: 192.168.2.3
- Transmission Control Protocol, Src Port: 5076, Dst Port: 63108, Seq: 117768961, Len: 298
- Transport Layer Security
 - [2 Reassembled TLS segments (198 bytes): #37(8), #37(190)]
 - Process Variable Access
 - Magic: 0xca
 - Version: 2
 - Flags: 0x40
 - Command: GET (0x0a)
 - Size: 190
 - Operation ID: 268443648
 - Sub-command: 0x00
 - Status: OK (0xff)
 - Body: 04ba7b361e000000000000004002000000100000044c4f4c4ff6e61

View Secure Results in Wireshark

- Both **tcp** and **tls** are included in search message
- **TLS Handshake** takes place
- PVA session is encrypted, and we see the decrypted message thanks to the key log.





EPICS Security Technical Plan Osprey DCS

- Thank You

George McIntyre
george@level-n.com

