

# Forward-Looking Statements



This presentation may contain forward-looking statements regarding future events, plans or the expected financial performance of our company, including our expectations regarding our products, technology, strategy, customers, markets, acquisitions and investments. These statements reflect management's current expectations, estimates and assumptions based on the information currently available to us. These forward-looking statements are not guarantees of future performance and involve significant risks, uncertainties and other factors that may cause our actual results, performance or achievements to be materially different from results, performance or achievements expressed or implied by the forward-looking statements contained in this presentation.

For additional information about factors that could cause actual results to differ materially from those described in the forward-looking statements made in this presentation, please refer to our periodic reports and other filings with the SEC, including the risk factors identified in our most recent quarterly reports on Form 10-Q and annual reports on Form 10-K, copies of which may be obtained by visiting the Splunk Investor Relations website at [www.investors.splunk.com](http://www.investors.splunk.com) or the SEC's website at [www.sec.gov](http://www.sec.gov). The forward-looking statements made in this presentation are made as of the time and date of this presentation. If reviewed after the initial presentation, even if made available by us, on our website or otherwise, it may not contain current or accurate information. We disclaim any obligation to update or revise any forward-looking statement based on new information, future events or otherwise, except as required by applicable law.

In addition, any information about our roadmap outlines our general product direction and is subject to change at any time without notice. It is for informational purposes only and shall not be incorporated into any contract or other commitment. We undertake no obligation either to develop the features or functionalities described, in beta or in preview (used interchangeably), or to include any such feature or functionality in a future release.

Splunk, Splunk> and Turn Data Into Doing are trademarks and registered trademarks of Splunk Inc. in the United States and other countries. All other brand names, product names or trademarks belong to their respective owners. © 2023 Splunk Inc. All rights reserved.

# TLS of the SSLippery Slope

The .key to Your Splunk Connections  
SEC1936B

**Nick Bertram**

Senior On-Demand Consultant | Splunk

**Caroline “Cat” McGee**

Senior Assigned Expert/On-Demand Consultant | Splunk





## Caroline "Cat" McGee

Senior Assigned Expert/On-Demand Consultant  
Splunk



## Nick Bertram

Senior On-Demand Consultant  
Splunk

# About Us

Why we can speak to this

## Caroline McGee

6 years of Splunk experience in the Public Sector

- DoD/Fed/Civ/IC
- SIEM SME (SIEMstress), SAML SME, Splunk SME

Professional Services Consultant

- FRTIB/TSP
- Department of Veteran Affairs

Senior Assigned Expert/On-Demand Consultant

- DoD, State Department, FDIC, Supreme Court VA
- I also did some private sector work here too

## Nick Bertram

6 years of Splunk experience

- Insurance/Bank/Medical/Government
  - Most industry verticals, really
- Splunk SME

Splunk Team Lead - Hurricane Labs (MSSP)

- Operations manager (direction, SOP, runbooks, etc)
- Coaching/Mentoring/Customer Retention

Senior On-Demand Consultant

- Escalation and customer kick-off cases
- Automation and Procedures

# Brief Agenda

Our goal is targeted an Intermediate audience, covering some basics, but also hopefully not too far off in the weeds.

1. **TLS Basics for a Splunk Admin**
  - Even your 5 year old will understand
2. **Get Organized**
  - Be confident before you break prod
3. **Parameters, Configuration, Connections, oh my!**
  - Untangle and understand TLS configuration within the Splunk Platform
4. **Troubleshooting and Not Commonly Known**
  - Let our suffering have purpose





# TLS Basics

What we're talking about

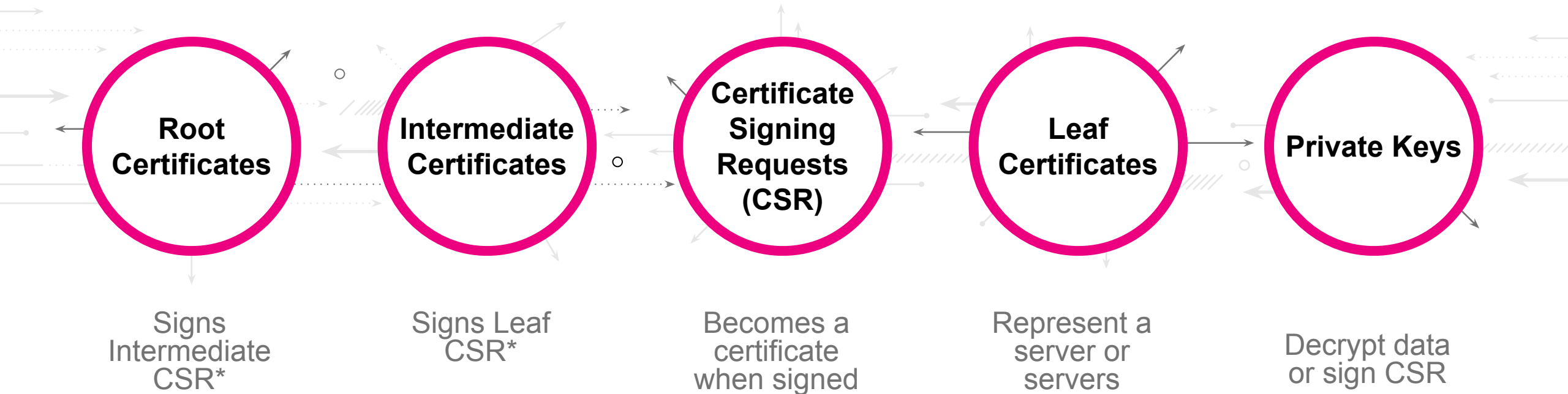
I mean if you signed up for this you probably have realized you're talking about TLS

# Transport Layer Security

## Oversimplified

- The client initiating the connection retrieves the public key from the remote server's certificate
- The client and server establish a shared secret
- Using the shared secret and public key, the client encrypts the traffic it sends to the remote server
- The remote server uses the private key to decrypt the traffic
- TLS is “one-way” by default in Splunk Platform
  - For “two-way” TLS we have `requireClientCert`
  - *Until 9.0 we only have certificate authentication*
  - Starting in 9.0 we have hostname validation
    - [SVD-2022-0603](#)

# Parts of TLS



\* Generally speaking. Though roots can sign leafs directly, and intermediates can sign other intermediates



# Parts of TLS

Definitions you need to know

## Issuer certificates & certificate authorities

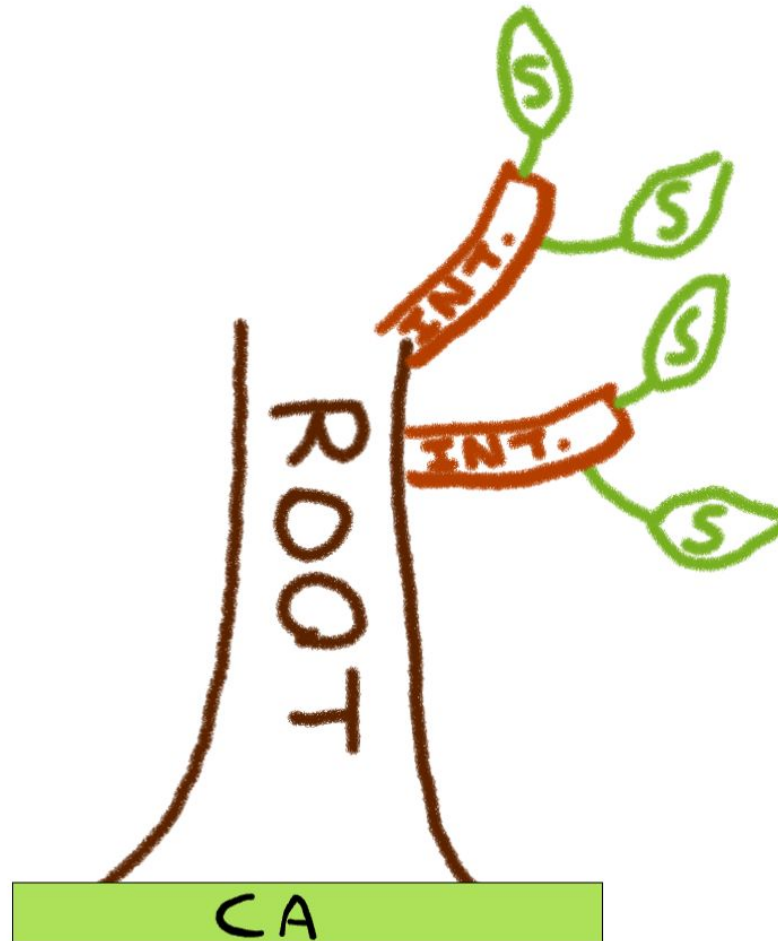
- *The terms “issuer certificate” and “certificate authority” are often used interchangeably.*
  - *But really, the CA is the team or organization in charge of the issuer certificates*
- These are your root and intermediate certificates. They don't represent a system or set of systems; instead they represent a trusted authority on verification.
- A root certificate is most often used to issue intermediate certificates, and then the intermediate is used to issue leaf (client/server) certificates
  - Root > Intermediate > Leaf

## Leaf certificates

- *They are often referred to as client or server certificates.*
- Leaf means that the certificate is unable to sign any additional certificates.
- Server certificates are used when “someone will initiate a connection to me”.
- Client certificates are used when “I'm going to initiate a connection to someone that will require I present a certificate for authentication purposes”.

# The “Trust” of Certs

Royalty free, and Legal approved



# Parts of TLS

Continued

## Public & Private Keys

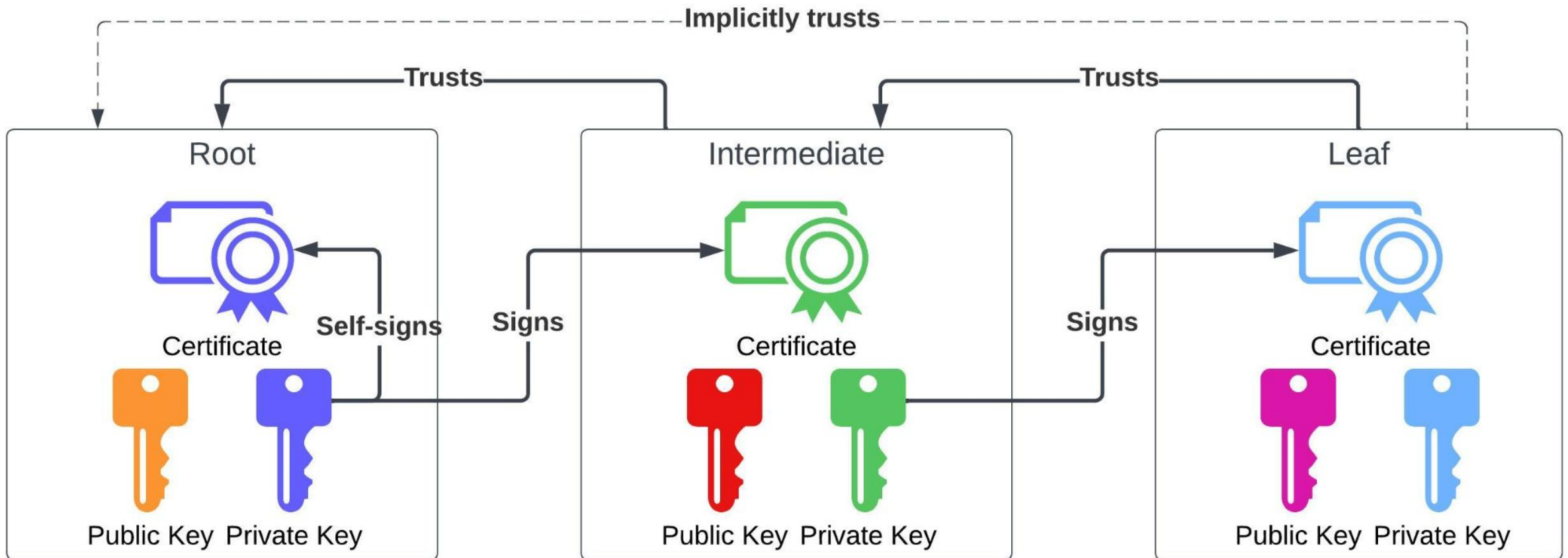
- Public keys are used to encrypt the traffic
  - This is automatically generated as part of the TLS handshake since it's embedded in the certificate.
  - You do not manage/configure these
- Private keys are used to decrypt the traffic
  - They will be generated and used to create the CSR
  - You do manage/configure these.

## Certificate Signing Request (CSR)

- *The CSR must be signed by an issuer certificate in order to become a leaf (client/server) certificate.*
- A CSR contains information about a specific server or subset of servers.
- Typically you as the application owner will generate the CSR, then provide it to someone who manages the CA to sign it.
  - Once they sign it, they will send you back the certificate which you can then use on the server.

# A Matter of Trust

'Cause it's always been a matter of trust



# What it Looks Like

## Certificates

```
-----BEGIN CERTIFICATE-----
MIIEOTCCAiGgAwIBAgIJAL3LsMxKn0ZWMA0GCSqGSIB3DQEBBQUAME
0xCzAJBgNVBAYTAlVTMQswCQYDVQQIDAJUWDERMA8GA1UECgwITmlj
ayBMdGQxHjAcBgNVBAMMFU5pY2tzIEludGVyYWVkaWF0ZSBDQTAeFw
0yMzAyMDgyMTA4NDJaFw0yNDYyMDgyMTA4NDJAMFAxCzAJBgNVBAYT
AlVTMQswCQYDVQQIDAJUWDEPMA0GA1UEBwwGQXVzdGluMQ4wDAYDVQ
QKDAVTaGFtdTETMBEGA1UEAwwKZm9yY2FyZDZyZGVycyCCASIWdQYJKoZI
hvcNAQEBAQADggEPAADCCAQoCggEBAK8+xL0CEZxmmXp6Cu8Vyy60XT
geQUZh
-----END CERTIFICATE-----
```

## Private Keys

### **UNENCRYPTED (no sslPassword)**

```
-----BEGIN PRIVATE KEY-----
MIIEvgIBADANBgkqhkiG9w0BAQEFAASCBAKgwggSkAgEAAoIBAQCvPsS9AhGcZpl6egrVfcsutF04HkFGYXvTvUN4QAXoFtCHvh4L3a7MIzLXrrjy7Q3w8kvhUnSS9Xi1FAKN+QLei6f/ShrPXEkv7Ii+35PNvTbwB6lZCoSs31QTM69EEphn7vddyc1e8H0SXV3kscu3PT2JUoE/WnechyeQ6P42KmdKQh23PZpREfh2ZVi+zEpu++Dk207/iHXJ+ke6EuDzOs5b6s4Z82d79++1
-----END PRIVATE KEY-----
```

### **ENCRYPTED (requires sslPassword)**

```
-----BEGIN RSA PRIVATE KEY-----
MIIEvgIBADANBgkqhkiG9w0BAQEFAASCBAKgwggSkAgEAAoIBAQCvPsS9AhGcZpl6egrVfcsutF04HkFGYXvTvUN4QAXoFtCHvh4L3a7MIzLXrrjy7Q3w8kvhUnSS9Xi1FAKN+QLei6f/ShrPXEkv7Ii+35PNvTbwB6lZCoSs31QTM69EEphn7vddyc1e8H0SXV3kscu3PT2JUoE/WnechyeQ6P42KmdKQh23PZpREfh2ZVi+zEpu++Dk207/iHXJ+ke6EuDzOs5b6s4Z82d79++1
-----END RSA PRIVATE KEY-----
```

# What Makes a Certificate Valid?

Server



**Issuer**

That we trust



**Not  
Before/After**

That covers  
today's date



**Subject CN**

That does not  
match the  
issuer CN

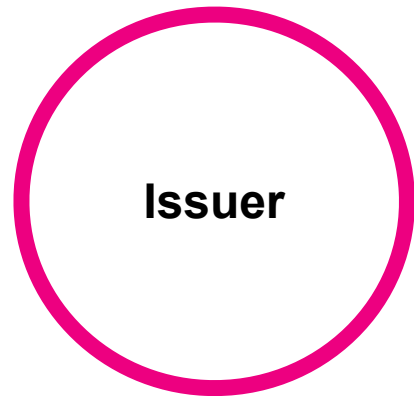


**x509v3 SAN**

That covers  
how the server  
will be  
connected to  
(IP, FQDN, etc)

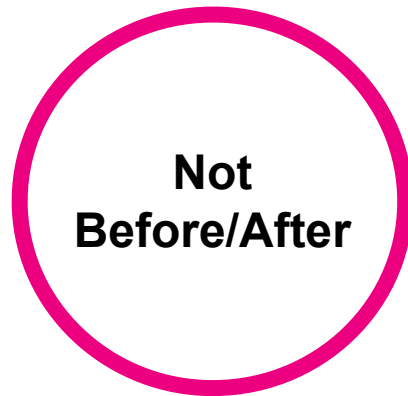
# What Makes a Certificate Valid?

Client



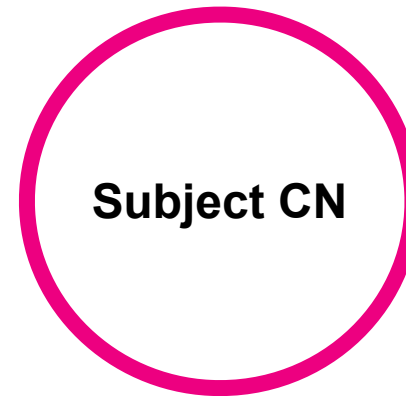
Issuer

That we trust



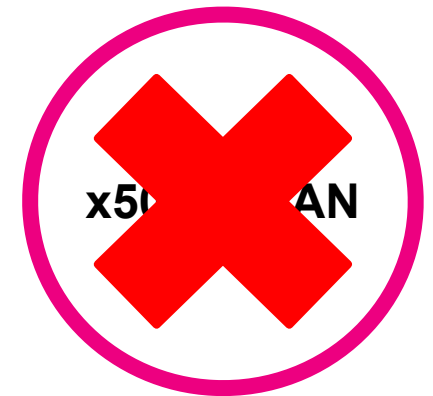
Not  
Before/After

That covers  
today's date



Subject CN

That does not  
match the  
issuer CN



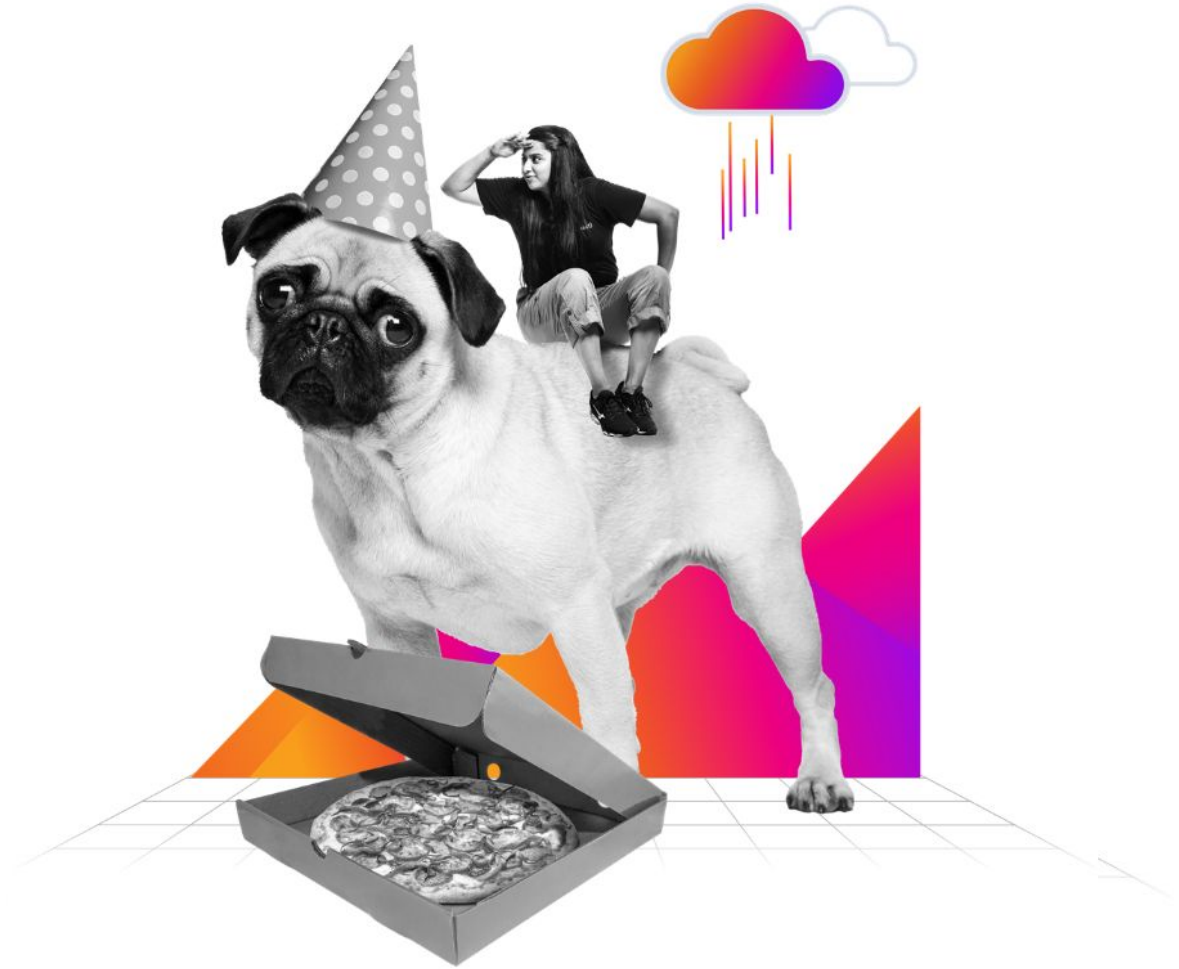
x509 SAN

SAN doesn't  
matter on a  
client as of  
version 9.0

# Let's Get Organized

Your room won't clean itself

1. Determine where you want to use TLS
2. Map out what certificates you'll need
3. Verify your chain and certificate validity





# A Foreword About Splunk<sup>®</sup> Cloud

- **This presentation is focused on-prem Splunk Platform (Enterprise/Universal Forwarder), and you will almost always have on-prem components even if you're running Splunk Cloud**
  - Deployment Server, Heavy Forwarder, Intermediate Heavy/Universal Forwarder
  - On-prem also includes customer controlled Cloud (Azure, AWS, GCP, etc)
- The primary concepts and configurations do not change
- If dual forwarding or using Federated Search; You must combine the Splunk Cloud CA's with your On-prem CA's into a single sslRootCAPath
  - This means you must place it in an app that will take precedence over the sslRootCAPath set by your 100\_stack\_splunkcloud UF credential package.
  - Splunk Cloud CA file is "stack\_cacert.pem" within 100\_stack\_splunkcloud/default
- You can not change any TLS settings or use your own certificates in Splunk Cloud.

# A Foreword About Splunk® Cloud

Function	UF (OP)	HF (OP)	Intermediate (OP)	DS (OP)	SH (OP)	IDX (OP)	Splunk Cloud
Management	Your App	Your App	Your App	Your App	Your App	Your App	Splunk Controlled
S2S Data->OP	Your App	Your App	Your App	Your App	Your App		
S2S Data->Cloud	Splunk UF Package**	Splunk UF Package**		Splunk UF Package**	Splunk UF Package**		Splunk Controlled
S2S Data->Int->Cloud	Your App	Your App	Your App + Splunk UF Package**	Your App	Your App		Splunk Controlled
Web	Your App	Your App	Your App	Your App	Your App	Your App	Splunk Controlled
Federated Search					Your App**		Your Config (GUI)

\* OP = On-prem or customer controlled Cloud

\*\* You must include Splunk Cloud CA in sslRootCAPath

# Step 1 - Determine Where You'll Use TLS

Management



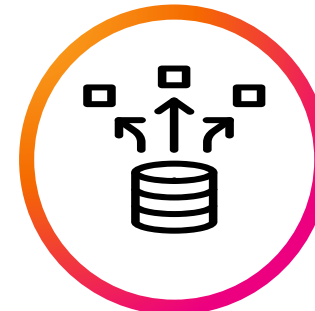
Data



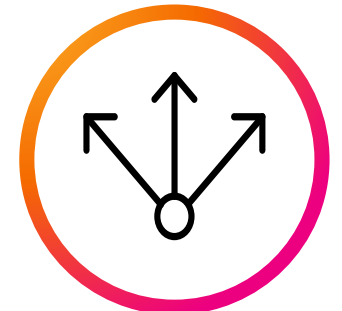
Web



KVStore



Replication



# Where TLS is Used

## Never all-or-nothing

- Management

- Encrypted by default
- How Splunk Platform instances talk to each other
- How third parties talk to Splunk
- How users/scripts can interact with REST API

- Data

- Splunk-to-Splunk (s2s) - **Not** encrypted by default\*
  - How Splunk Platform forwards data (by default) to another Splunk instance
- httpout - Encrypted by default\*\*
  - Can be used to forward data from UF to HF/IDX
- HEC - Encrypted by default
  - Used for HTTP data into Splunk Platform
  - Re-uses Management serverCert by default
- API (modular/scripted input) - Depends on code
  - Generally going to be HTTPS out to a third party

- Web

- **Not** encrypted by default\*
- Used to access the Splunk Platform GUI

- KVStore

- Encrypted by default
- Re-uses Management TLS settings by default
  - Unless you're FIPS...

- Replication (IDX/SH Cluster)

- **Not** encrypted by default
- Used for various functions of the cluster among peers
- Indexer Bucket Replication, KO Replication, etc.

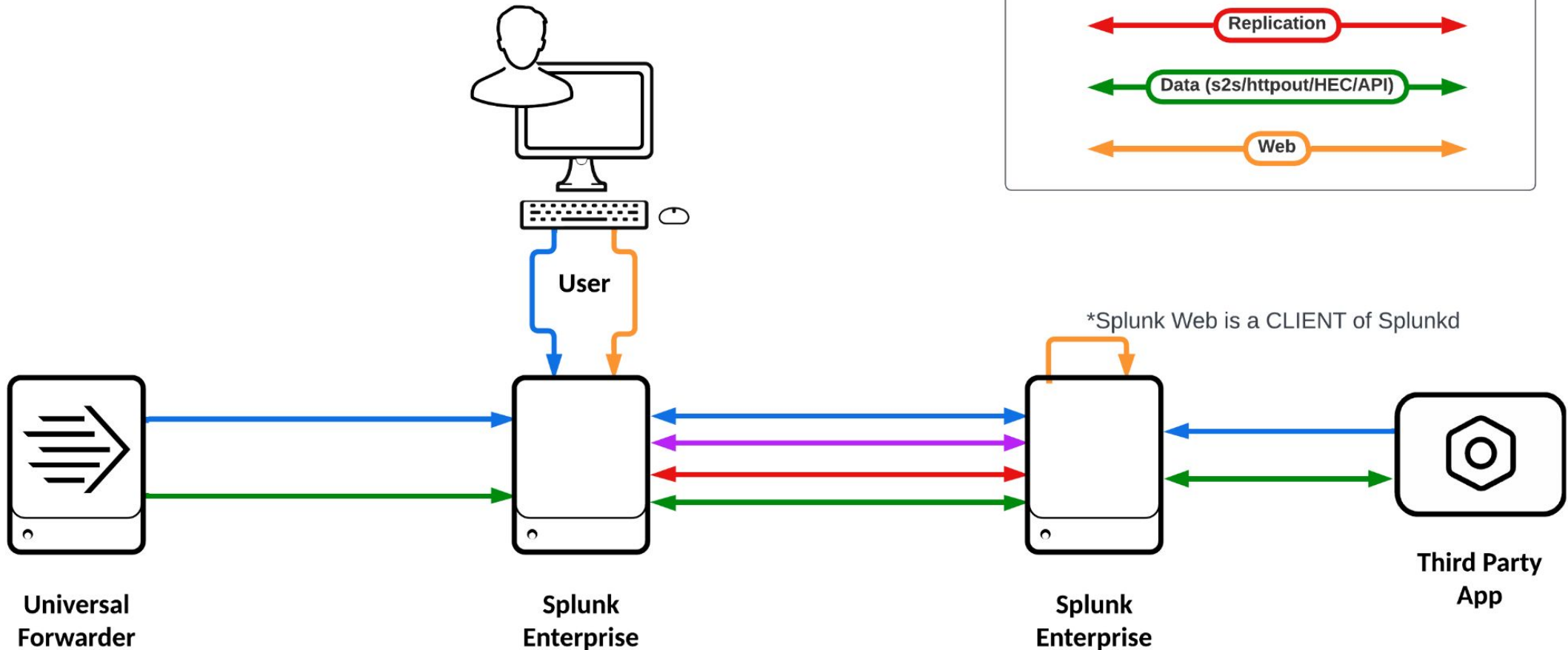
\* Splunk Cloud does encrypt these by default, not manageable by customer

\*\* Only because it sends to Enterprise HEC which is encrypted by default, we don't "enable" it on the forwarder.

Ref: <https://docs.splunk.com/Documentation/Splunk/latest/Security/AboutsecuringyourSplunkconfigurationwithSSL>

# Let's Map It Out

Direction of traffic is critical to understand



# Step 2 - Gather Your Certs

What we need

## Server Certificate



One for each Splunk Enterprise System

## Single Forwarder Certificate



One for **all** Universal Forwarders

## Private Key



For each Server and Forwarder certificate

## All Issuer Certificates



For those certificates and any other Splunk Platform you connect to

# Gather Your Certs

## Take inventory

- **Each Splunk® Enterprise server in your environment needs a certificate**
  - Plus it's private key
  - These need a subject alternative name (SAN) list that match the server (IP, hostname, FQDN)
  - Intermediate UF's, too
- **A single “forwarder” certificate used for data forwarding and management on UF's**
  - Plus it's private key
  - Can be used for data forwarding on Enterprise as well
  - The client certificate does not need to match (SAN/CN) the server it resides on. We only need to trust who signed it.
- We only need one certificate for a server and it will be used for all TLS functions on that system.
  - If you have a genuine reason to use a different certificate for each function (I.E., Web vs data vs management) that is totally supported, just more admin overhead
- The issuer certs for the following:
  - All Enterprise certificates in your environment
  - The single forwarder certificate
  - Other environments you connect to by a **Splunk Initiated Connection:**
    - Data forwarding to any other Splunk Platform environments or third parties using TLS
    - Management connection to any other Splunk Platform environment
      - I.E. Federated Search

# Deployment Considerations

The type of certificate affects how you can configure it in Splunk Platform

- For Enterprise, choose a strategy for certs
  - Wildcard (on the SAN), can only wildcard the leftmost subdomain
    - Valid: \*.splunk.com, \*.idx.splunk.com
    - Invalid: host\*.splunk.com, \*.\*.splunk.com
  - Multi-host (on the SAN)
  - Individual
  - A mix of the above
- Wildcards and multi-host can be bundled in an app and deployed
- Individual certs must be placed manually on each server
  - You can place the certs into same location (**not in an app!**) on each box manually, then use an app from the DS that points to that location
    - I.E., manually place the cert in \$SPLUNK\_HOME/etc/auth/mycert.pem, then in an app deployed via DS, serverCert would be set to that path
- Avoid setting a passphrase on your private keys
  - **You can not use requireClientCert if you have a passphrase on your keys!**
  - If using a passphrase:
    - You **must** set the sslPassword directly in etc/system/local/server.conf for [sslConfig].
    - It can not be set in an app.
      - This does not apply to other conf files like inputs, outputs, or web.
  - If you think a passphrase on the keys is more secure... Anyone with access to view the private key would also have the ability to decrypt the sslPassword, get the passphrase, and view the private key
  - *If you have a genuine requirement to do so, go for it, just understand it increases administrative overhead and adds another failure point*



# Policy Considerations

Find who's responsible

- What security policies do you need to consider?
  - Key generation, TLS version, ciphers, use of SAN
  - Are wildcards allowed? Multi-host SAN?
    - Watch out if you're on a shared domain with other teams!
  - Are you using FIPS?
- Who manages certificates in your organization?
  - Do they require that you provide the CSR?
  - Do they have any special requirements?
  - Can it be signed with an internal CA or does it need a publicly-trusted CA (\$\$)?
  - If no one manages certs, can you make your own CA?
- **Remember, in the end all we need**
  - A server certificate with its corresponding private key
  - The issuer certificate(s) for the above certificate
- Some **current** best practices to consider
  - For private key generation, use RSA w/ AES256 with at minimum 2048 key length
  - Server/client certs should have an expiry of less than 365 days
  - Private keys should ideally not leave the server(s) they were generated for
    - You're going to break this rule, so aim for "only put private keys on the systems they're intended for or the mechanism of deployment"
  - Monitor **all** your certificates for expiry, act well before this happens!

**Never put the private key for the issuers in Splunk Platform.**

# Placing the.....

Glue your files together

## server/clientCert

---

- In a single file, place (in order)
  - The certificate
  - The private key specific to that certificate
  
- *Put nothing else in this file! No issuers!*
- *Name these files in a way that makes sense*

## sslRootCAPath

---

- Put all the issuer certificates into a single file, one after the other.
  
- *Put nothing else in this file! No server/clients certs or private keys!*
- *Intermediates at the top, followed by roots at the bottom.*
- *You can add comments above each certificate line*
  - *I'd recommend adding the Subject+Issuer above each certificate block*

# Step 3 - Verify You Have Everything

Do this right and you'll be home before 5

## Certificate



Ensure the certificate is valid

## Private Key



Ensure the private key matches the certificate

## Issuer Certificates



Ensure your CA bundle verifies the certificate

# Verify You Have Everything

Do this right and you'll be home before 5

- View the certificate
  - Verify the Subject, Issuer, Validity, and SAN
  - `openssl x509 -noout -text -in certificate_here`
- Verify you have the correct private key
  - The modulus must match
  - `openssl x509 -noout -modulus -in certificate_here | openssl md5`
  - `openssl rsa -noout -modulus -in certificate_here | openssl md5`
  - Remember, your “certificate\_here” at this point should be the certificate followed by the private key.
  - You **must** run **both commands** against the **same file**
- Verify you have the full chain
  - A non-OK status is a show stopper
  - `openssl verify -CAfile ca_here certificate_here`
  - **Get the order correct, do not flip ca\_here and certificate\_here**
- (Optional) View all the issuer certificates
  - `openssl crl2pkcs7 -nocrl -certfile combined_ca.pem | openssl pkcs7 -print_certs -noout`
  - This is more of a handy command so you can easily see all of the certs inside of the CA

# Verify (Optional)

Some extras after you've configured Splunk Platform

- Test TLS Connectivity
  - `openssl s_client -connect host:port`
  - `openssl s_client -CAfile /path/here/cabundle.pem -connect host:port`
- View the certificates of a remote host (verify your change)
  - `openssl s_client -showcerts -connect host:port`

# Verify Server Cert

It should look like this

```

-----BEGIN CERTIFICATE-----
MIIEKzCCAHOgAwIBAgIJAL3LsMxKn0ZQMA0GCSqGSIb3DQEBBQUAME0xChAJBgNV
BAYTAVtMQ4wDAYDVQQKDAVtMQ4wDAYDVQQKDAVtMQ4wDAYDVQQKDAVtMQ4wDAY
FU5pY2tzIEIudGVyYbWVkaWF0ZSBDQTAeFw0yMzAyMDgyMTA4MzZaFw0yNDYy
MTA4MzZaMEkxChAJBgNVBAYTAVtMQ4wDAYDVQQKDAVtMQ4wDAYDVQQKDAVtMQ4
dGluMQ4wDAYDVQQKDAVtMQ4wDAYDVQQKDAVtMQ4wDAYDVQQKDAVtMQ4wDAYDV
AQEFAAOCAQ8AMIIBChKCAQEA01Yf5QyScdvecMFKiwwHmZnKVeJz99FbAIAAe7D1
mJ9Ug+zZwFmBmxI//Lkdz2jhQpLw5bIVGn4Rh1n6ye/SW9BbQxYhyCyEUUaoL+xN
IAC/qXhCFQuQ7o4dTrfPflmYVBH3ps4/LFIZ/QMbcgQCKYg2EkfHjJpHwzLFee3K
RT10jywTLRL8pT0Ya1v5gR0eULYJ9ogaXU4Jpr9956QDz/PjdQhVEb9UfHbrJMC7
zuiUeRAUY/gqW0jIMKzQZrp5S6c7r7X45EPnzIQU1sh1+bZdm8fvzAmXBNxYqX/C
0efGaKag50ugkEPjrnHUj5dq2xMhxHtaRjP/41oamb+HwIDAQABoxIwEDA0BgNV
HREEBzAFggNoZjEwDQYJKoZIhvcNAQEFBQADggIBAKzlgCricBCP9k+g7Km2EDW/t
ua8YJujabKfSYXswonHvTvSiode4RB4wD4s00yNmdl8zFBkx0b4MwD90y5FJVTZ
dJLtvCTC5ckWa0ccD7MY5caSUPVPUdJCjQ1krW71GKPkCg5vF50Mc0caHcdBR6jU
ADSZr0zjxSdwBG3QDApBHZjrxjce0UULr0hDV6Ij/RCKdkEzo4oYRSBe+baIKimY
ayROYbgGue2/ruMLEmarNF/ZMScMhca+mtlyWGxbsUjrrj5R7u/mCpzJ3w7sTwhZ
VMAgp0mpbcpPttDppj0z3sZDK909evZYidue0khrZ/1NDOHIAN50KfminIEXk9n0x
QuBUceB07s8B00uw7b2/0MK7aExIY00i102ajri9YqBsq6RPk90wvMPz1ITxCb65
HZuNlybLMP/r7CyzJa00LH6ZTQTQon606GZLLoJBme0hIkCUcfbyhVwGcGZkYMLV
bs0VaJwa5FtMEH8sQ9hPQ3zmqHDZi7F7KJFUVJyS/Mg8ZDp8aLoqm1n3tvYEVovr
YdmfZNL40YLS4D7bL0iFs4i054e5ZikDa1KD56XqsAbB7Alv+jeVc4KSHSgS0xE
+ww0cmafjjXydHJKTYxVGNg9Hb1F0HCTTLEV4t+NRbpa+edrf/ZlchSUUkVp8kKe
C0gIn2fkQZe3CNFJPBST
-----END CERTIFICATE-----
-----BEGIN PRIVATE KEY-----
MIIEwAIBADANBgkqhkiG9w0BAQEFAASCBAwggSmAgEAAoIBAQDTVh/LDJJx295w
wUqLc8eZk2Rv6Pb318FogAB7sPWYn1SD7NnAWYGbEj/8uQPPa0FckvD1shUafhGH
WfrJ79Jb0FtDFiHILIRZrqiX7E0gAL+peEIVC5Dujh10t89+WZHUefemzj8sUhn9
AxtYBAKRiDYSR8e00kFDMSV57cpFPU6PLBmtEvyLPRhrW/mBE55Qtgn2iBpdTgmm
v33npAPP8+N1AdURv1QUduskLw06JR5EBRj+CpbSMgwrNBmun1LpzuvtfjkQ+fM
hBTWYHX5tL2bx+/MCZcE3Fipf8LR58ZopqDk66CQQ+0s0dSP12rbEyhE1pEmn/j
WihqZv4fAgMBAACggEBAMw7gNsziRqrZo4E3er92UjbbHa3AU1s0kF5SxSTD4LQ7
1csgS1cNqKiZvGiYy8vXCUnzAwaXULmnd30cQbaBIwNDmc59RuxRyR0VobvfyndR
CGfYJdR8tvUzNbNrAwSRxRoJuxDiZD6KoT7Xj+Bha7IEtmx4HauF0tay7BBm8oBx
eeINidvArSloyAB18liTV11NrfKlc/UTQXjnclCEP2CZFY6GFQcUG3fthZLPRBk
yAe3wFwr72KKZHGEbeTJBzBA+0mjPAarKzqweZpW01RV4GwbyMsQan0BZBRrh2sB
cdwpf0/nNoLMBZyVN0u2MSwenf0HutVcgIsRXt/PboECgYEA+z0g19n78KF6MI4E
QKkKwj0t8B+J6+Vq5sds4bk2sYJwz0DuFYBdsLR+zennyNeOC4RbHy2pJahuz0dj
44K0zMJgMH9/p6Ccg6j35GBNs0BmDiZvkIBzgcjG2CD4ee7o3xMf2RKPu/ILVd
GsLXFq0sZzDmdBkuW0PdVG4U1D8CgYEA11+OUfqkKhu3TMOXP4X13Cmk7aVLjmFT
MIWh1d4/11pgWaKuscqt2FVQQC1yWx85nwYGIVf8Zn+bbDKW1Z7J9hSXMW60bXxs
M8GFttoFhJZFsV+U95rWkgmCc+zC/VgtfbWtJpZ+z8J9osPeKNC7EhplXsDdnKA
GV8ja0gX3iECgYEAy1iawQeRtt0GCFtOpfWj9kkEwgdmSggCqj8iRmhc2UXn9QF
n12McFBd9S6gFW1fkIpxR/oDF3svYsYrWy0a20/MYnHSHb2woHU730PhW1sfjZo
T+xCeU0ujsZceJAZc0nCOtkNici+hBEjY5uHgJZePkasDQ2hBxpWfSyUk5cGcYEA
oBJYT9u7egzlhftU/ZSCG9kjyLDy8Bar4sNFXfo6Ts/QDq7X0jKGD0QqLY7ZkJor
lGLAyaWAZZnWJUGQU6MaDfiNyukVnxA7x1fNDvMJ6NpbYBJfQIXnb9D7kTdmUS9p
WCZgU+7Ev5uZDxEum/1PHIxe8DgU5v3Nsg3A3qapnqECgYEAzLk7aY4vAm47WLN
au90Qi9/cQWRzPs2ox009LGi2/JjRjvHvY2SdgYydPCucyNbwTfvtaWma0hA0b1X
8gwGskGJDAK+nvDNGZiD9bTHhBSEdHwF+39vGYSEIjckh5asSrMbSsG/npfmlhA9
h7s5LcZWAMNSnpw4+igjc2iev+c=
-----END PRIVATE KEY-----

```

# Verify Your CA File

Your CA file should look something like this

```
subject=/C=US/ST=TX/O=Nick Ltd/CN=Nicks Intermediate CA
issuer=/C=US/ST=TX/L=Austin/O=Nick Ltd/CN=Nicks Root CA
-----BEGIN CERTIFICATE-----
MIIFhDCCA2ygAwIBAgIBATANBgkqhkiG9w0BAQUFADBWMQswCQYDVQQGEwJVUzEL
MAkGA1UECAwCVFgxZzANBgNVBACMBkF1c3RpbjERMA8GA1UECgwITmIjayBMdGQx
FjAUBgNVBAMMDU5pY2tzIFJvb3QgQ0EwHhcNMjMwMjA4MjEwODM1WhcNMzAwNTEy
MjEwODM1WjBNMQswCQYDVQQGEwJVUzELMAkGA1UECAwCVFgxETAPBgNVBAoMCE5p
zpUWUShbRMsEOHXHKBwLSUk4hB307E5T4xmgAPdPloqyy943vpqtIHVUdaqF3DY
/aRj70HrwcWm9UUAEEjKdhjUndNvni13xTt+Z+3gRLb15lKvl+yFnL+GeyXm3N0B
6hsdd0zi83RHbBXD01wi5Fcr6izEDWJ0cePDrRHp07FnOzB2mLnk7oNZB/Pi8BMF
t4C0j56Tt+P9rCg/BQUkpVEo2F0PResb
-----END CERTIFICATE-----
subject=/C=US/ST=TX/L=Austin/O=Nick Ltd/CN=Nicks Root CA
issuer=/C=US/ST=TX/L=Austin/O=Nick Ltd/CN=Nicks Root CA
-----BEGIN CERTIFICATE-----
MIIFfzCCA2egAwIBAgIJAN4UuVoGk1DkMA0GCSqGSIb3DQEBCwUAMFYxCzAJBgNV
BAYTA1VTMQswCQYDVQQIDAjUWDEPMA0GA1UEBwwGQXVzdGluMREwDwYDVQQKDAh0
aWNrIEEx0ZDEWMBQGA1UEAwwNTmIjay3MgUm9vdCBDQTAEFw0yMzAyMDgyMTA4MjNa
Fw0zMzAyMDUyMTA4MjNaMFYxCzAJBgNVBAYTA1VTMQswCQYDVQQIDAjUWDEPMA0G
I+FWgnZ03CV9racKrq0nkCnWsqhCZMF6Xu/XVNVfowDKu5WyywVJ5IdeJ1dG2ocU3
ZSFebWgbhrxAijQcxIR/rzN2RARqjJRRj+4SP77592qIwU5qZc5PiC/FCv9PaQfT
QUZWeKcms6AQduzMUNqIywiXBw==
-----END CERTIFICATE-----
```

# Verify the Certs Are Valid

CN, Validity, Subject, SAN by your powers combined... I AM VALID CERT!

```
openssl x509 -noout -text -in certificate_here
```

```
nbertram@ ██████████ CA_Gen % openssl x509 -in tls/server_certs/hf1.pem -noout -text
Certificate:
  Data:
    Version: 3 (0x2)
    Serial Number:
      bd:cb:b0:cc:4a:9f:46:50
    Signature Algorithm: sha1WithRSAEncryption
    Issuer: C=US, ST=TX, O=Nick Ltd, CN=Nicks Intermediate CA
    Validity
      Not Before: Feb  8 21:08:36 2023 GMT
      Not After : Feb  8 21:08:36 2024 GMT
    Subject: C=US, ST=TX, L=Austin, O=Shamu, CN=hf1
    Subject Public Key Info:
      Public Key Algorithm: rsaEncryption
      RSA Public-Key: (2048 bit)
      Modulus:
        00:d3:56:1f:e5:0c:92:71:db:de:70:c1:4a:8b:0b:
        c7:99:93:64:55:e8:f6:f7:d7:c1:68:80:00:7b:b0:
        f5:98:9f:54:83:ec:d9:c0:59:81:9b:12:3f:fc:b9:
        03:cf:68:e1:42:92:f0:e5:b2:15:1a:7e:11:87:59:
        fa:c9:ef:d2:5b:d0:5b:43:16:21:c8:2c:84:59:46:
        a8:97:ec:4d:20:00:bf:a9:78:42:15:0b:90:ee:8e:
        1d:4e:b7:cf:7e:59:98:54:11:f7:a6:ce:3f:2c:52:
        19:fd:03:1b:72:04:02:91:88:36:12:47:c7:8e:3a:
        47:c3:32:c5:79:ed:ca:45:3d:4e:8f:2c:13:2d:12:
        fc:a5:3d:18:6b:5b:f9:81:13:9e:50:b6:09:f6:88:
        1a:5d:4e:09:a6:bf:7d:e7:a4:03:cf:f3:e3:75:01:
        d5:11:bf:54:14:76:eb:24:c0:bb:ce:e8:94:79:10:
        14:63:f8:2a:5b:48:c8:30:ac:d0:66:ba:79:4b:a7:
        3b:af:b5:f8:e4:43:e7:cc:84:14:d6:c8:75:f9:b6:
        5d:9b:c7:ef:cc:09:97:04:dc:58:a9:7f:c2:d1:e7:
        c6:68:a6:a0:e4:eb:a0:90:43:e3:ac:d1:d4:8f:97:
        6a:db:13:21:c4:7b:5a:44:9a:7f:e3:5a:28:6a:66:
        fe:1f
      Exponent: 65537 (0x10001)
    X509v3 extensions:
      X509v3 Subject Alternative Name:
        DNS:hf1
    Signature Algorithm: sha1WithRSAEncryption
      ac:e5:18:2a:e2:04:23:fd:93:e8:3b:2a:6d:84:0d:6f:ed:b9:
      af:18:26:e8:da:6c:a7:d2:61:25:f0:a2:71:ef:4e:f4:a2:a1:
```



# Verify the Correct Private Key

Wrong key = won't work

```
openssl x509 -noout -modulus -in certificate_here | openssl md5
```

```
openssl rsa -noout -modulus -in certificate_here | openssl md5
```

```
nbertram@ [REDACTED] CA_Gen % openssl x509 -noout -modulus -in tls/server_certs/hf1.pem | openssl md5
2e40005fe6843aa42cdb21a16db97091 ←
nbertram@ [REDACTED] CA_Gen % openssl rsa -noout -modulus -in tls/server_certs/hf1.pem | openssl md5
2e40005fe6843aa42cdb21a16db97091 ←
```

# Verify All the Issuers Are There

Missing issuers will make verification fail later

```
openssl verify -CAfile ca_here certificate_here
```

```
nbertram@ [REDACTED] CA_Gen % openssl verify -CAfile combined_ca.pem tls/server_certs/hf1.pem  
tls/server_certs/hf1.pem: OK ←
```

# Verify the Issuer Certificate

You should see all the issuer certificates you expect

```
openssl crl2pkcs7 -nocrl -certfile combined_ca.pem | openssl pkcs7 -print_certs -noout
```

```
nbertram@[REDACTED] CA_Gen % openssl crl2pkcs7 -nocrl -certfile combined_ca.pem | openssl pkcs7 -print_certs -noout
subject=/C=US/ST=TX/O=Nick Ltd/CN=Nicks Intermediate CA
issuer=/C=US/ST=TX/L=Austin/O=Nick Ltd/CN=Nicks Root CA

subject=/C=US/ST=TX/L=Austin/O=Nick Ltd/CN=Nicks Root CA
issuer=/C=US/ST=TX/L=Austin/O=Nick Ltd/CN=Nicks Root CA
```

# Configuration Files

Not so hard after all

1. Parameter Definitions & Locations
2. TLS Verification
3. Config Templates



# Cert Parameters

The core pieces

## serverCert



When I receive a connection, this is the certificate I will present to the client.\*

## clientCert



When I establish a connection, this is the certificate I will present to the remote server.

## sslRootCAPath



This is the collection of issuer certificates that I trust. I'll refer to this bundle when validating a TLS connection.

\* Be aware that serverCert is also... the client certificate for requireClientCert as clientCert only exists for outputs

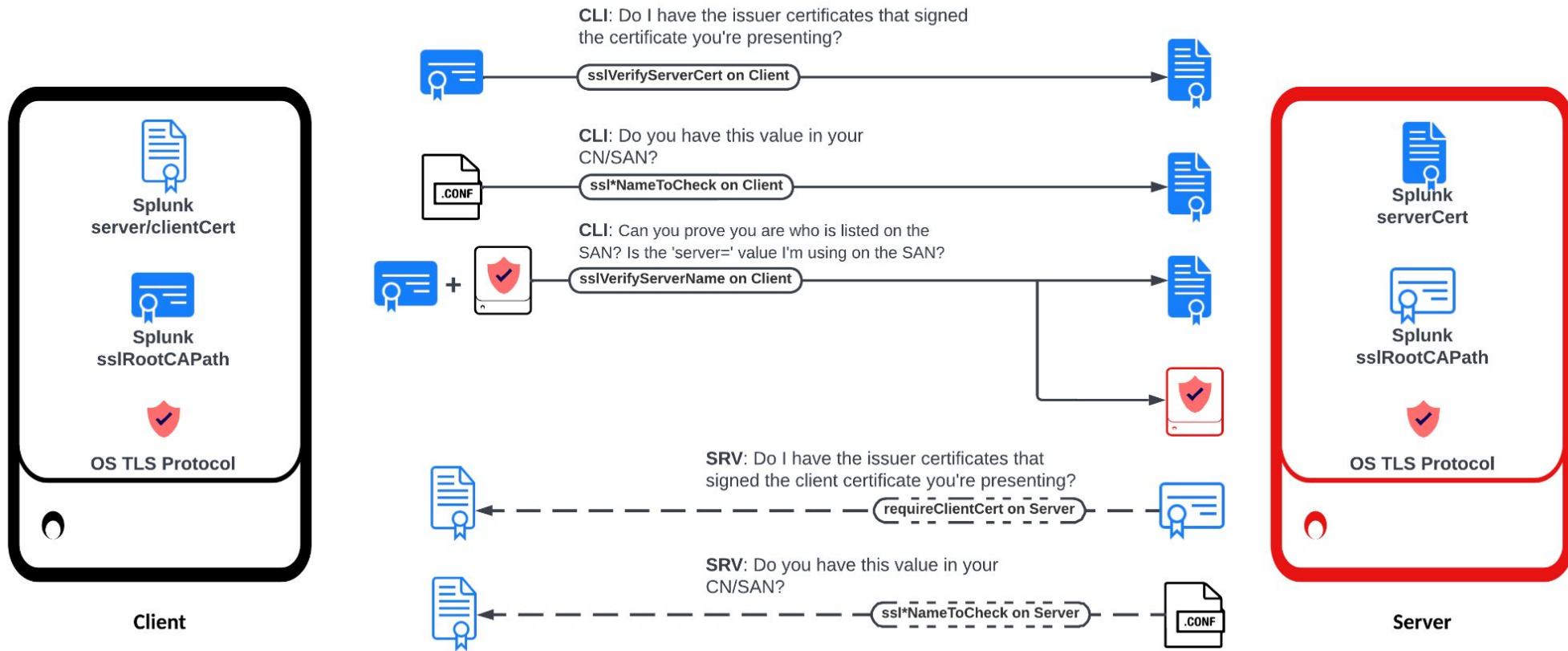
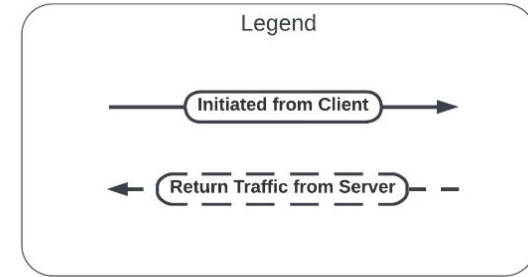
# What Goes Where

Function	Config File	Stanza	Client Parameter (requireClientCert)	Server Parameter
Management	server.conf	[sslConfig]	serverCert	serverCert
Web	web.conf	[settings]	serverCert for Splunkd Browser for GUI	serverCert + privKeyPath
Data Forwarding (s2s)	outputs.conf	[tcpout:<name>]	clientCert	
Data Receiving (s2s + tcp/udp)	inputs.conf	[SSL]		serverCert
Replication (SH/IDX Cluster)	server.conf	[replication_port-ssl://<port>]	serverCert	serverCert
HEC [ <b>defaults</b> to Management]	inputs.conf	[http]		serverCert
KVStore [ <b>defaults</b> to Management]	server.conf	[kvstore]	serverCert	serverCert
Certificate Authority [ <b>everything</b> ]	server.conf	[sslConfig]	sslRootCAPath	sslRootCAPath

# TLS Verification Parameters

- sslVerifyServerCert - When I initiate a connection, I'll check if whoever issued the certificate being presented to me is in my sslRootCAPath. I will not verify if they are who they say they are.
  - Client—>Server
- sslVerifyServerName - When I initiate a connection, I'll check if how I'm connecting to the server is on the SAN list of the certificate being presented to me. I will also challenge them to prove they are who they say they are.
  - Client—>Server
  - Requires sslVerifyServerCert=true + version 9.x+
- requireClientCert - When I receive a connection, the client must present a certificate to me. I'll check if whoever issued the certificate is in my sslRootCAPath. I will not verify if they are who they say they are.
  - Client<--[return traffic]--Server
- ssl\*NameToCheck - When I establish a connection, I'll look at the [CN/SAN] of the certificate being presented to me and check if it matches the [CN/SAN] I require. I will not verify if they are who they say they are.
  - Client—>Server
    - Requires sslVerifyServerCert=true
  - Client<--[return traffic]--Server
    - Only with requireClientCert=true

# TLS Verification Direction



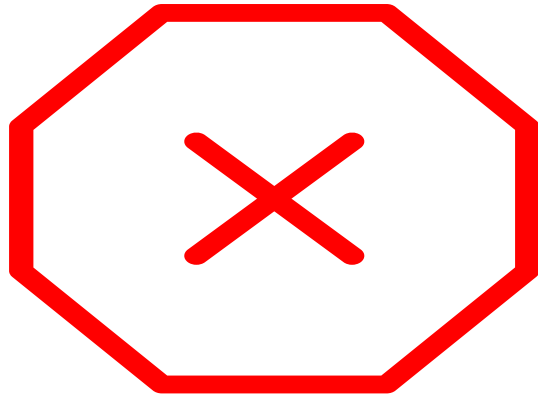


# Before You Deploy

- Break the deployment up into two stages:
  - First, enable TLS
  - Second, enable verification [if desired]
- Use a single CA file for your entire environment, don't overcomplicate things
  - Remember if you're dual forwarding (I.E., Splunk® Cloud) you may need another set of issuers appended to your sslRootCAPath
- Triple check all file paths
- Optional: *Consider opening a port for non-TLS and TLS data forwarding*

**Test as much beforehand as possible**

# Before You Deploy



**These next slides are not copy-paste templates. These are to show you the stanzas and the settings you are generally going to be configuring. Refer to the spec sheets for full configuration options if you need something specific like version or cipher or timeouts.**

We can't stress this enough, if you turn on verification before certs are deployed, you will break everything. Seriously. Do not just copy paste the following slides into your environment.

# Management

```
server.conf

[sslConfig]
sslRootCAPath = $SPLUNK_HOME/etc/apps/000_enterprise_tls/certs/ca_certs.pem
serverCert= $SPLUNK_HOME/etc/apps/000_enterprise_tls/certs/server_cert.pem
sslVerifyServerCert= true
sslVerifyServerName = true
requireClientCert = true
## Password must be set in $SPLUNK_HOME/etc/system/local (except for IDX Cluster)
sslPassword =
```

# Web

```
web.conf

[settings]
enableSplunkWebSSL = true
## Do not use $SPLUNK_HOME in web.conf!
serverCert= /opt/splunk/etc/apps/000_enterprise_tls/certs/server_cert.pem
privKeyPath = /opt/splunk/etc/apps/000_enterprise_tls/certs/server_cert.key
sslPassword =
```

# Data Receiving/Forwarding

```
inputs.conf

[splunktcp-ssl:9996]
disabled = 0

[SSL]
serverCert=
$SPLUNK_HOME/etc/apps/000_enterprise_tls/certs/server_cert.pem

sslPassword =
requireClientCert = true
```

```
outputs.conf

[tcpout]
defaultGroup = primary_indexers

[tcpout:primary_indexers]
server = idx1:9996, idx2:9996
clientCert=
$SPLUNK_HOME/etc/apps/000_splunk_forwarding_tls/certs/forwarder_cert.pem

sslPassword =
sslVerifyServerCert= true
sslVerifyServerName = true
```

# Replication (SH/IDX Cluster)

```
inputs.conf

[replication_port-ssl://<port>]
serverCert = $SPLUNK_HOME/etc/apps/000_enterprise_tls/certs/server_cert.pem
sslPassword =
requireClientCert = true
```

# HEC

```
inputs.conf

[http]
enableSSL = 1
# Not needed if you already set serverCert in server.conf under [sslConfig]
serverCert= $SPLUNK_HOME/etc/apps/000_enterprise_tls/certs/server_cert.pem
sslPassword =
# Be careful setting this to true for HEC
requireClientCert = false
```

# KVStore

```
server.conf

[kvstore]
# Not needed if you already set serverCert in server.conf under [sslConfig]
serverCert= $SPLUNK_HOME/etc/apps/000_enterprise_tls/certs/server_cert.pem
sslPassword =
requireClientCert = true
sslVerifyServerCert= true
sslVerifyServerName = true
```





# Aftercare & Troubleshooting

It's a happy little accident

# After You Deploy

Check your work

- Connect to each port and verify you see the new cert showing up
  - `openssl s_client -quiet -CAfile /path/here/cabundle.pem -connect host:port`
- Check internal logs for issues (see next slide)
- Set up monitoring or calendar reminders for certificate expiry
  - Remember, it's industry practice to rotate your private keys yearly

# When You Have Trouble Trusting

## Where to Look

---

- index=\_internal component=
  - TcpOutputFd
  - TcpInputProc
  - TcpInputConfig
  - SSLCommon
  - X509Verify

## What to Look For

---

- Misconfiguration
- Missing Issuers from sslRootCAPath
- Wrong Private Key or sslPassword
- Failing Hostname Validation
- Failing requireClientCert challenge
- Version and Cipher Mismatch

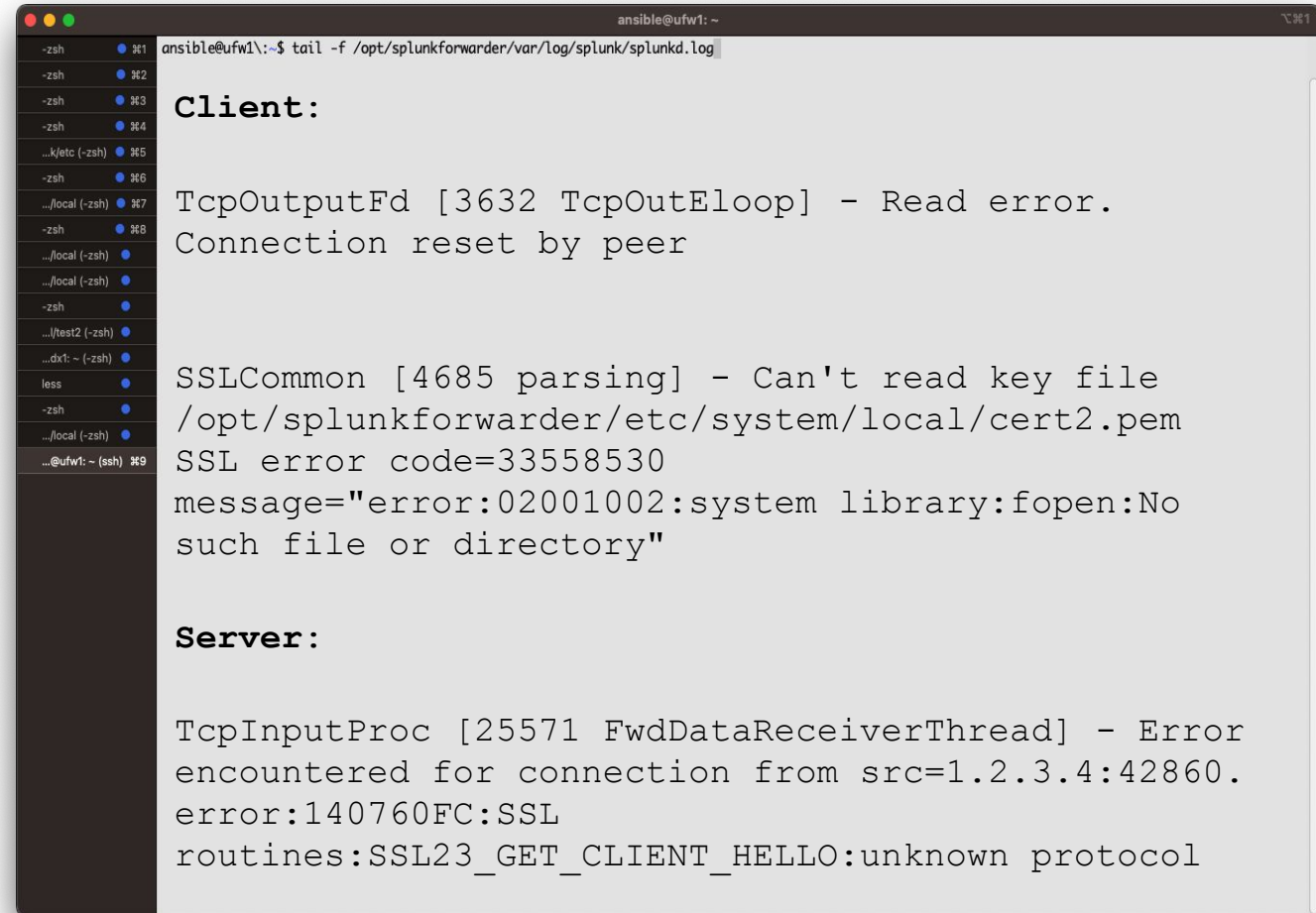
# Missing clientCert

## Problem

- Missing clientCert in outputs.conf
- Typo to location of cert file

## Fix

- Add clientCert to outputs.conf
- Correct file path



```
ansible@ufw1:~$ tail -f /opt/splunkforwarder/var/log/splunk/splunkd.log

Client:
TcpOutputFd [3632 TcpOutEloop] - Read error.
Connection reset by peer

SSLCommon [4685 parsing] - Can't read key file
/opt/splunkforwarder/etc/system/local/cert2.pem
SSL error code=33558530
message="error:02001002:system library:fopen:No
such file or directory"

Server:
TcpInputProc [25571 FwdDataReceiverThread] - Error
encountered for connection from src=1.2.3.4:42860.
error:140760FC:SSL
routines:SSL23_GET_CLIENT_HELLO:unknown protocol
```

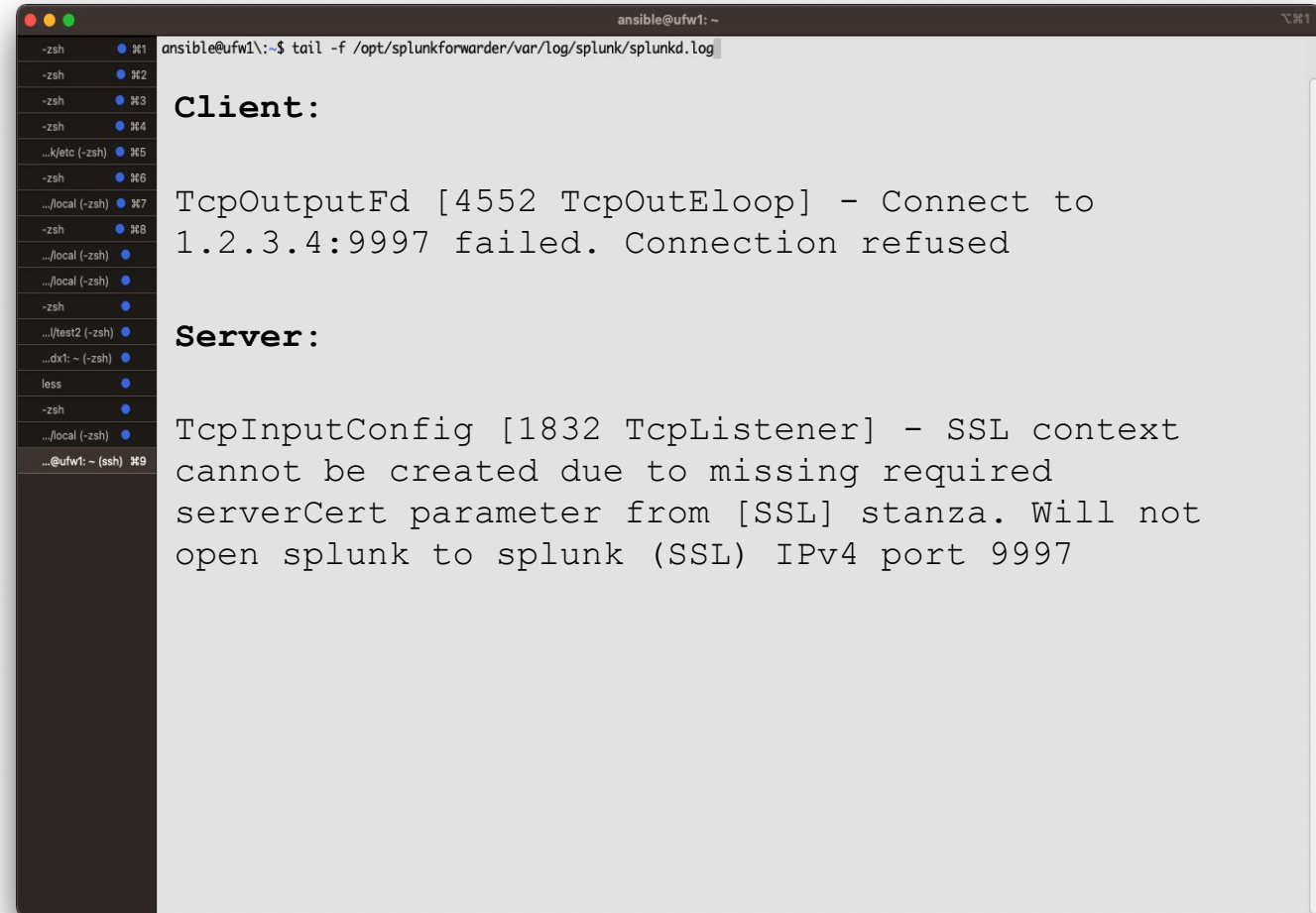
# Missing serverCert

## Problem

- Missing serverCert in inputs.conf
- Typo to location of cert file

## Fix

- Add serverCert in inputs.conf
- Correct file path



```
ansible@ufw1:~$ tail -f /opt/splunkforwarder/var/log/splunk/splunkd.log
Client:
TcpOutputFd [4552 TcpOutEloop] - Connect to
1.2.3.4:9997 failed. Connection refused
Server:
TcpInputConfig [1832 TcpListener] - SSL context
cannot be created due to missing required
serverCert parameter from [SSL] stanza. Will not
open splunk to splunk (SSL) IPv4 port 9997
```

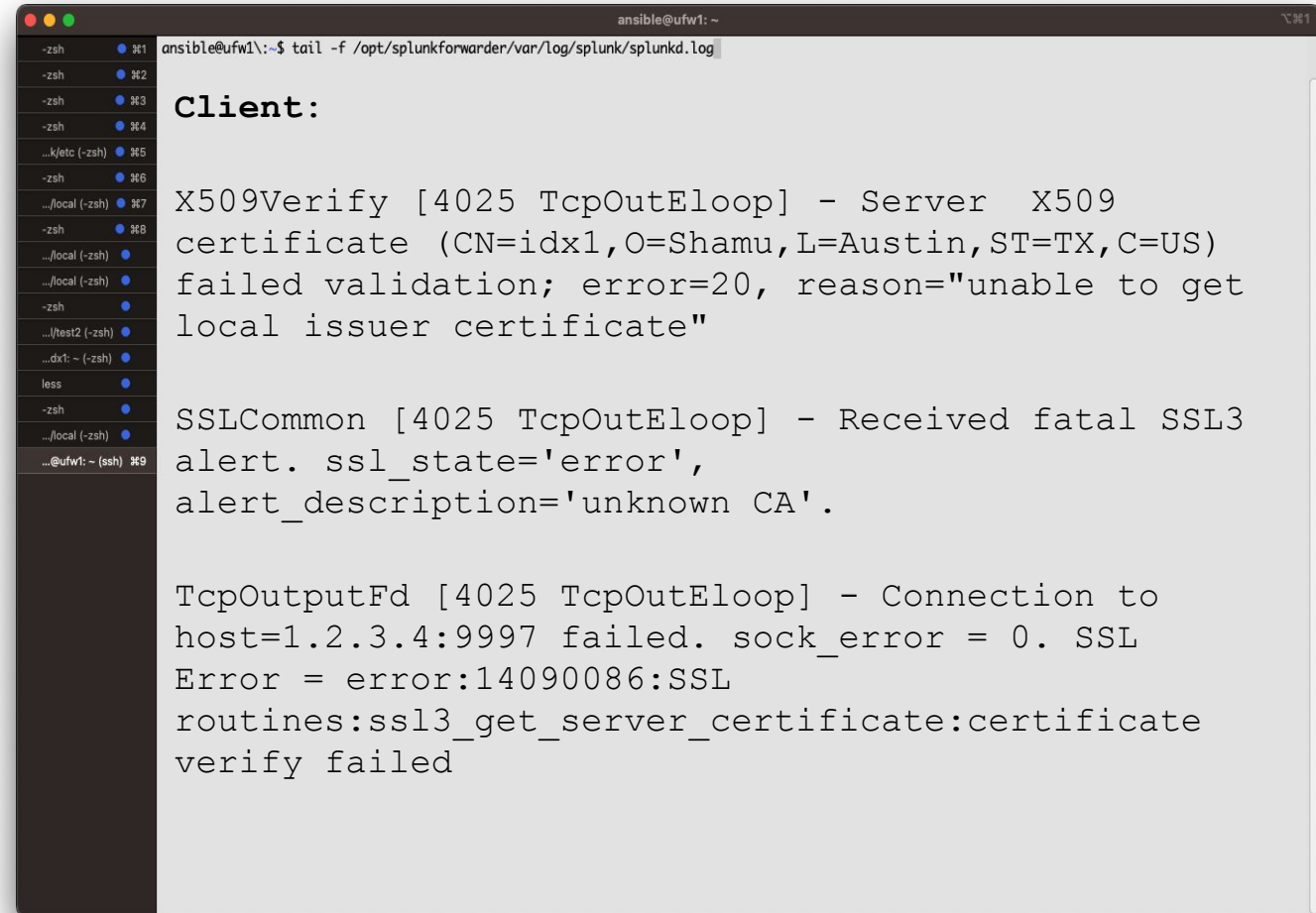
# Missing Issuers

## Problem

- FWD doesn't trust the serverCert of IDX

## Fix

- Add issuer of IDX serverCert to sslRootCAPath on FWD
- Verify sslRootCAPath file path



```
ansible@ufw1:~$ tail -f /opt/splunkforwarder/var/log/splunk/splunkd.log

Client:
X509Verify [4025 TcpOutEloop] - Server X509
certificate (CN=idx1,O=Shamu,L=Austin,ST=TX,C=US)
failed validation; error=20, reason="unable to get
local issuer certificate"

SSLCommon [4025 TcpOutEloop] - Received fatal SSL3
alert. ssl_state='error',
alert_description='unknown CA'.

TcpOutputFd [4025 TcpOutEloop] - Connection to
host=1.2.3.4:9997 failed. sock_error = 0. SSL
Error = error:14090086:SSL
routines:ssl3_get_server_certificate:certificate
verify failed
```

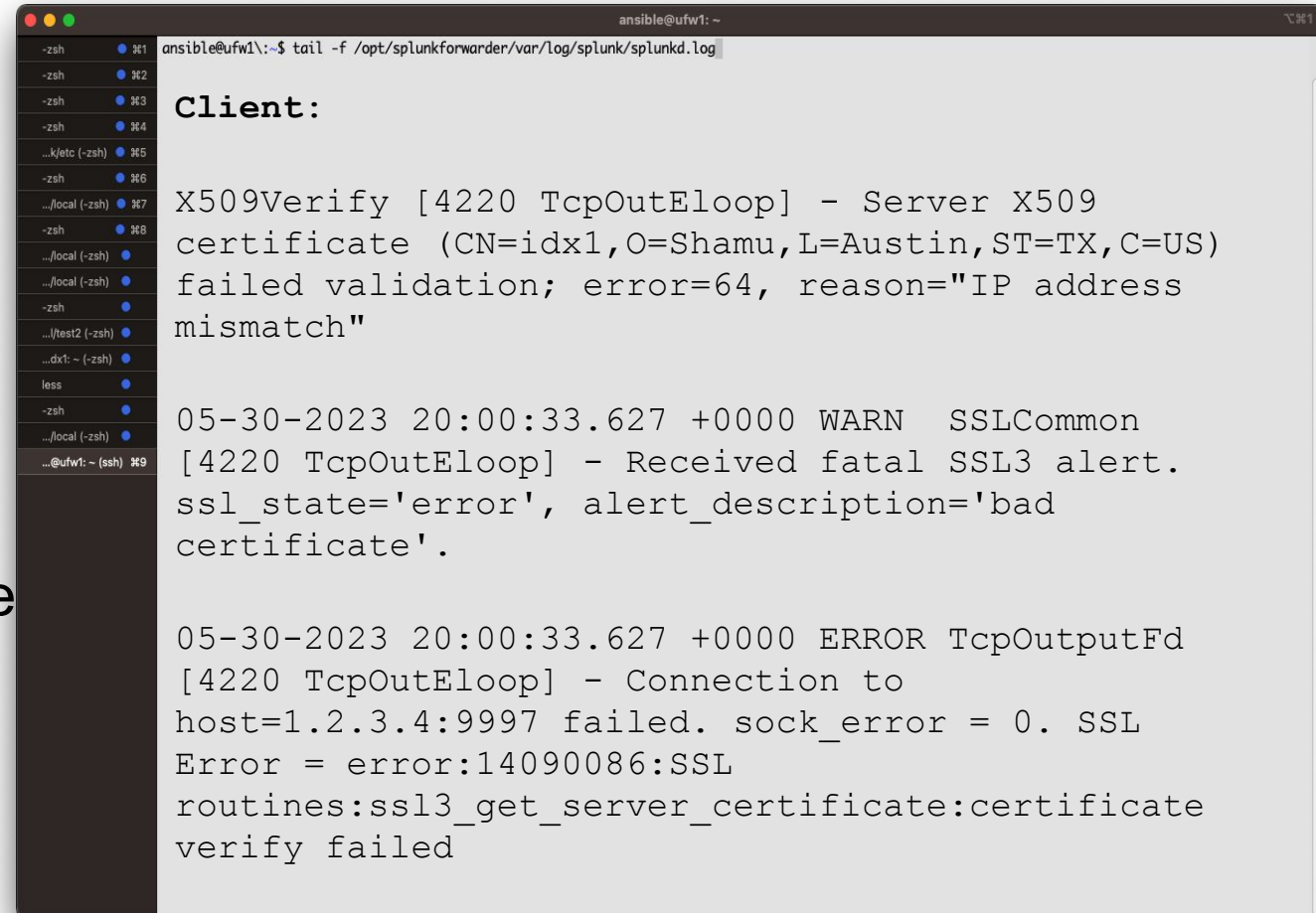
# Hostname Validation Fail

## Problem

- FWD doesn't trust the serverCert of IDX

## Fix

- Update outputs.conf URI to use a value on the SAN of IDX serverCert
- Re-generate certificate if SAN is wrong



```
ansible@ufw1:~$ tail -f /opt/splunkforwarder/var/log/splunk/splunkd.log

Client:

X509Verify [4220 TcpOutEloop] - Server X509
certificate (CN=idx1,O=Shamu,L=Austin,ST=TX,C=US)
failed validation; error=64, reason="IP address
mismatch"

05-30-2023 20:00:33.627 +0000 WARN SSLCommon
[4220 TcpOutEloop] - Received fatal SSL3 alert.
ssl_state='error', alert_description='bad
certificate'.

05-30-2023 20:00:33.627 +0000 ERROR TcpOutputFd
[4220 TcpOutEloop] - Connection to
host=1.2.3.4:9997 failed. sock_error = 0. SSL
Error = error:14090086:SSL
routines:ssl3_get_server_certificate:certificate
verify failed
```

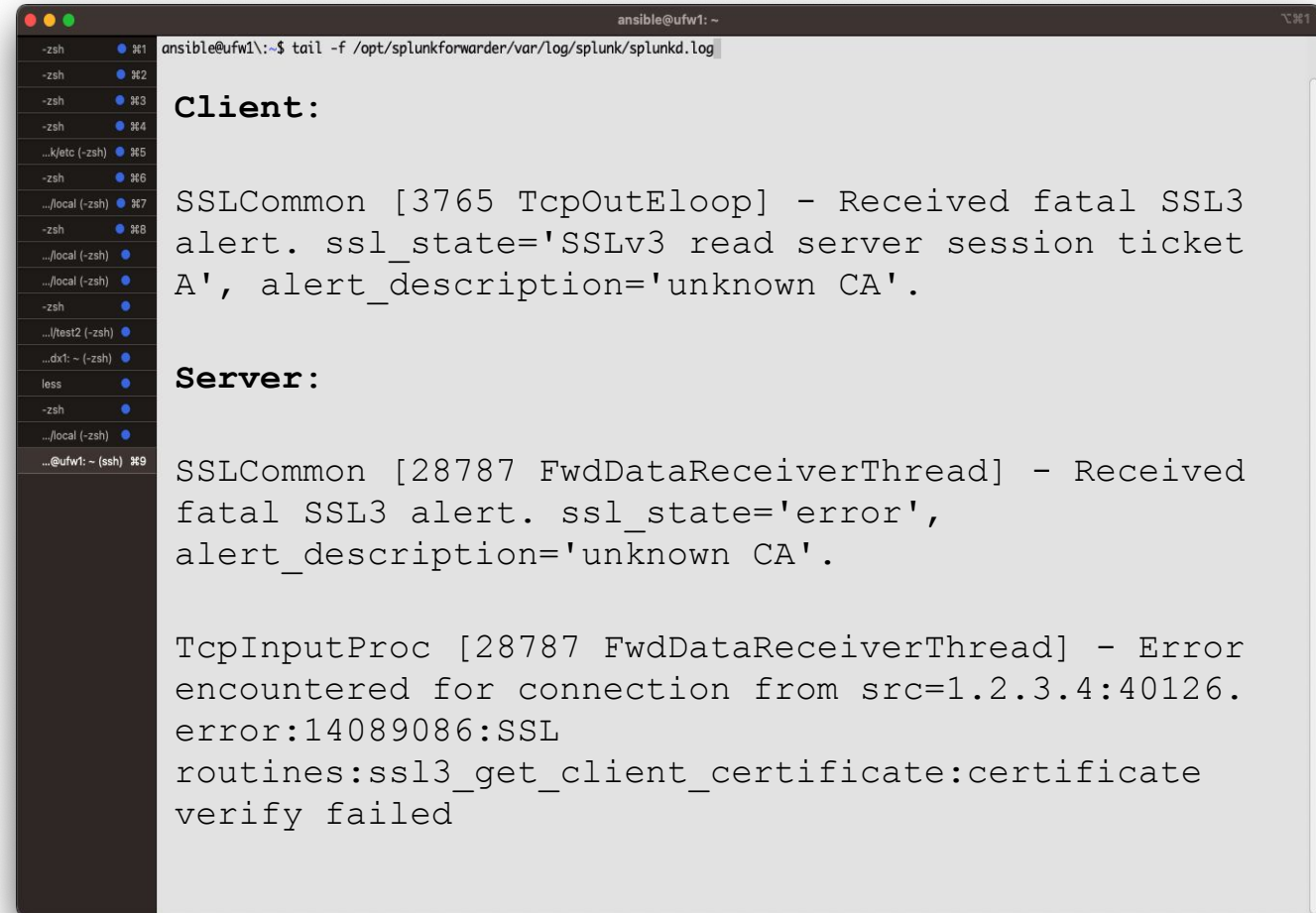
# requireClientCert Fail

## Problem

- IDX doesn't trust the clientCert of FWD

## Fix

- Add issuer of FWD clientCert to sslRootCAPath on IDX
- Verify sslRootCAPath file path



```
ansible@ufw1:~$ tail -f /opt/splunkforwarder/var/log/splunk/splunkd.log

Client:
SSLCommon [3765 TcpOutEloop] - Received fatal SSL3
alert. ssl_state='SSLv3 read server session ticket
A', alert_description='unknown CA'.

Server:
SSLCommon [28787 FwdDataReceiverThread] - Received
fatal SSL3 alert. ssl_state='error',
alert_description='unknown CA'.

TcpInputProc [28787 FwdDataReceiverThread] - Error
encountered for connection from src=1.2.3.4:40126.
error:14089086:SSL
routines:ssl3_get_client_certificate:certificate
verify failed
```



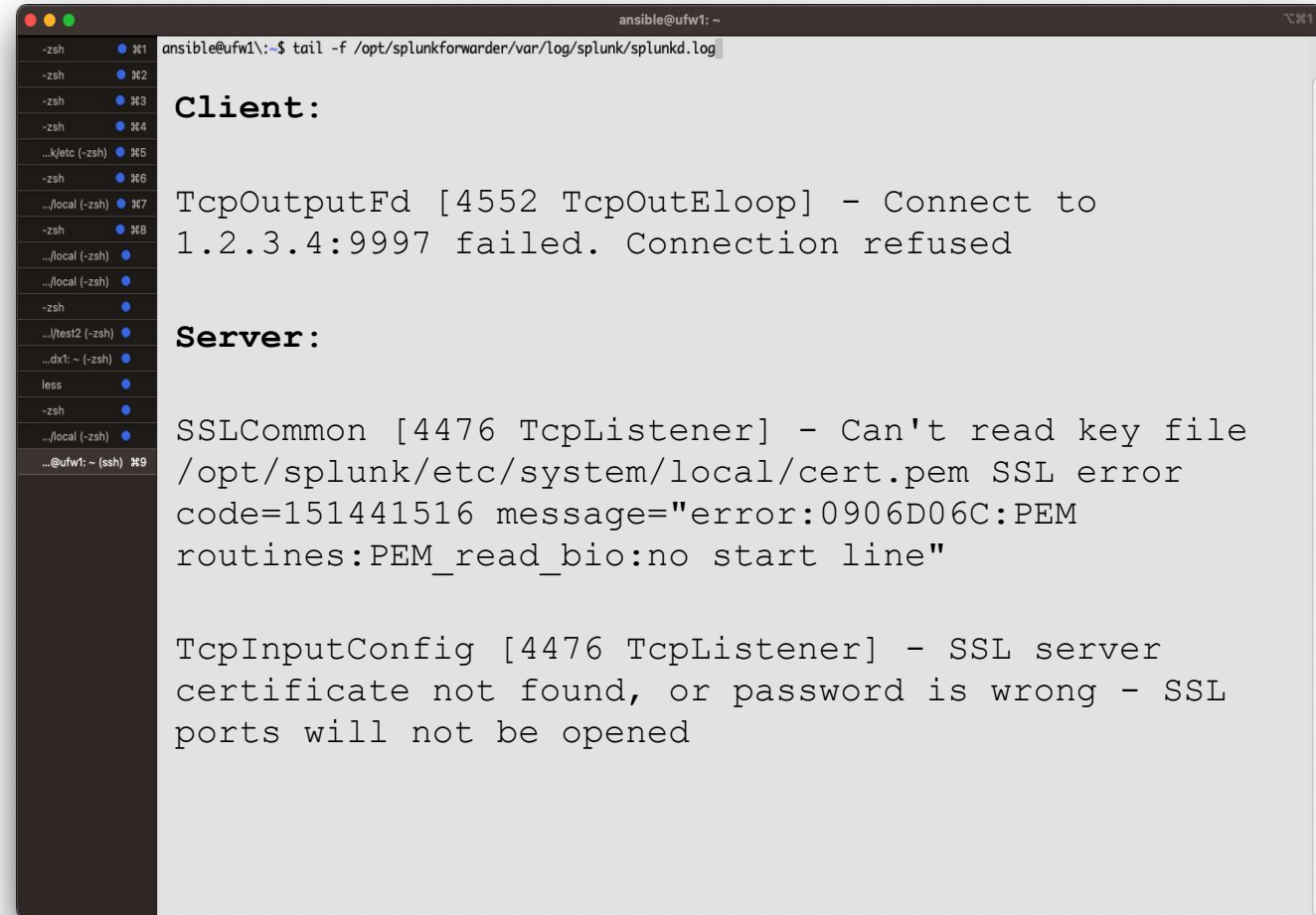
# Missing Private Key

## Problem

- IDX doesn't have private key in serverCert

## Fix

- Add private key to bottom of serverCert file



```
ansible@ufw1:~$ tail -f /opt/splunkforwarder/var/log/splunk/splunkd.log

Client:
TcpOutputFd [4552 TcpOutEloop] - Connect to
1.2.3.4:9997 failed. Connection refused

Server:
SSLCommon [4476 TcpListener] - Can't read key file
/opt/splunk/etc/system/local/cert.pem SSL error
code=151441516 message="error:0906D06C:PEM
routines:PEM_read_bio:no start line"

TcpInputConfig [4476 TcpListener] - SSL server
certificate not found, or password is wrong - SSL
ports will not be opened
```

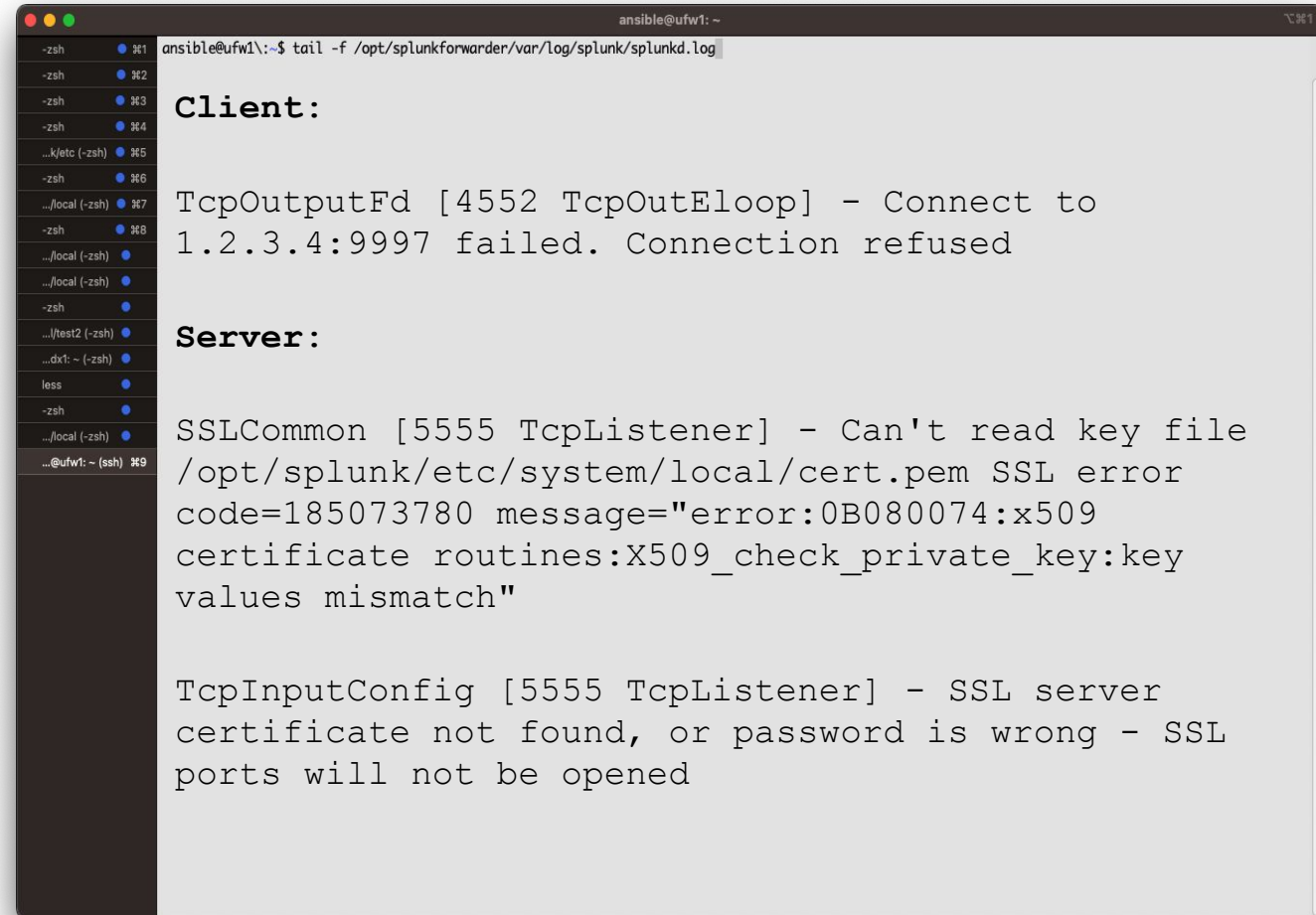
# Wrong Private Key

## Problem

- IDX has incorrect private key in serverCert

## Fix

- Replace private key at bottom of serverCert file



```
ansible@ufw1:~$ tail -f /opt/splunkforwarder/var/log/splunk/splunkd.log

Client:
TcpOutputFd [4552 TcpOutEloop] - Connect to
1.2.3.4:9997 failed. Connection refused

Server:
SSLCommon [5555 TcpListener] - Can't read key file
/opt/splunk/etc/system/local/cert.pem SSL error
code=185073780 message="error:0B080074:x509
certificate routines:X509_check_private_key:key
values mismatch"

TcpInputConfig [5555 TcpListener] - SSL server
certificate not found, or password is wrong - SSL
ports will not be opened
```

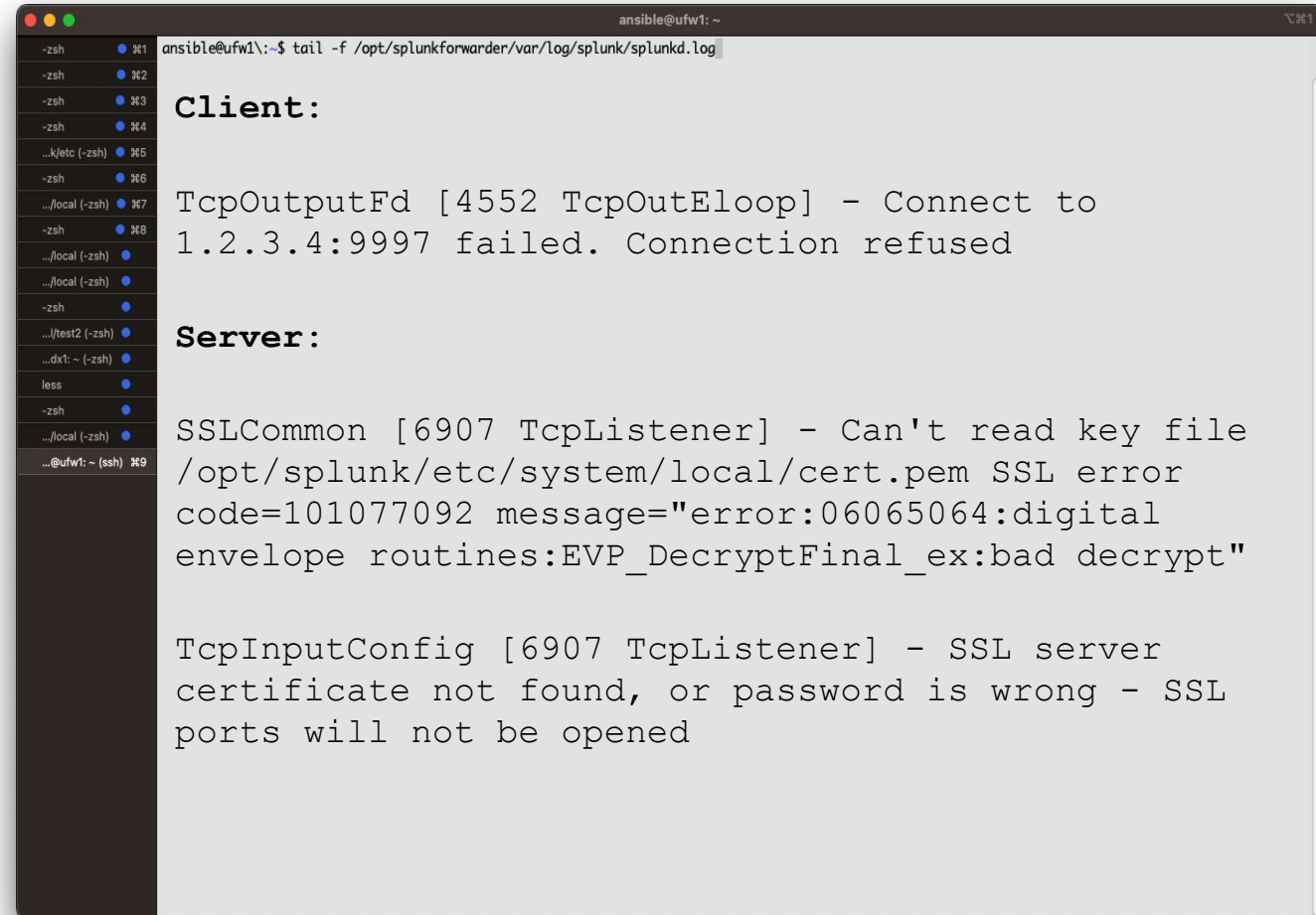
# Wrong sslPassword

## Problem

- Wrong sslPassword set for serverCert

## Fix

- Correct/Add sslPassword
- Remove passphrase from key



```
ansible@ufw1:~$ tail -f /opt/splunkforwarder/var/log/splunk/splunkd.log

Client:
TcpOutputFd [4552 TcpOutEloop] - Connect to
1.2.3.4:9997 failed. Connection refused

Server:
SSLCommon [6907 TcpListener] - Can't read key file
/opt/splunk/etc/system/local/cert.pem SSL error
code=101077092 message="error:06065064:digital
envelope routines:EVP_DecryptFinal_ex:bad decrypt"

TcpInputConfig [6907 TcpListener] - SSL server
certificate not found, or password is wrong - SSL
ports will not be opened
```

# Mismatched sslVersion

## Problem

- FWD requires TLS1.2, IDX require TLS 1.0

## Fix

- Correct sslVersions list on one or both sides of the connection



```
ansible@ufw1:~$ tail -f /opt/splunkforwarder/var/log/splunk/splunkd.log

Client:
SSLCommon [4946 TcpOutEloop] - Received fatal SSL3
alert. ssl_state='error', alert_description='protocol
version'.

TcpOutputFd [4946 TcpOutEloop] - Connection to
host=1.2.3.4:9997 failed. sock_error = 0. SSL Error =
error:14077102:SSL
routines:SSL23_GET_SERVER_HELLO:unsupported protocol

Server:
SSLCommon [13047 FwdDataReceiverThread] - Received
fatal SSL3 alert. ssl_state='error',
alert_description='handshake failure'.

TcpInputProc [13047 FwdDataReceiverThread] - Error
encountered for connection from src=1.2.3.4:52192.
error:1408A0C1:SSL routines:ssl3_get_client_hello:no
shared cipher
```

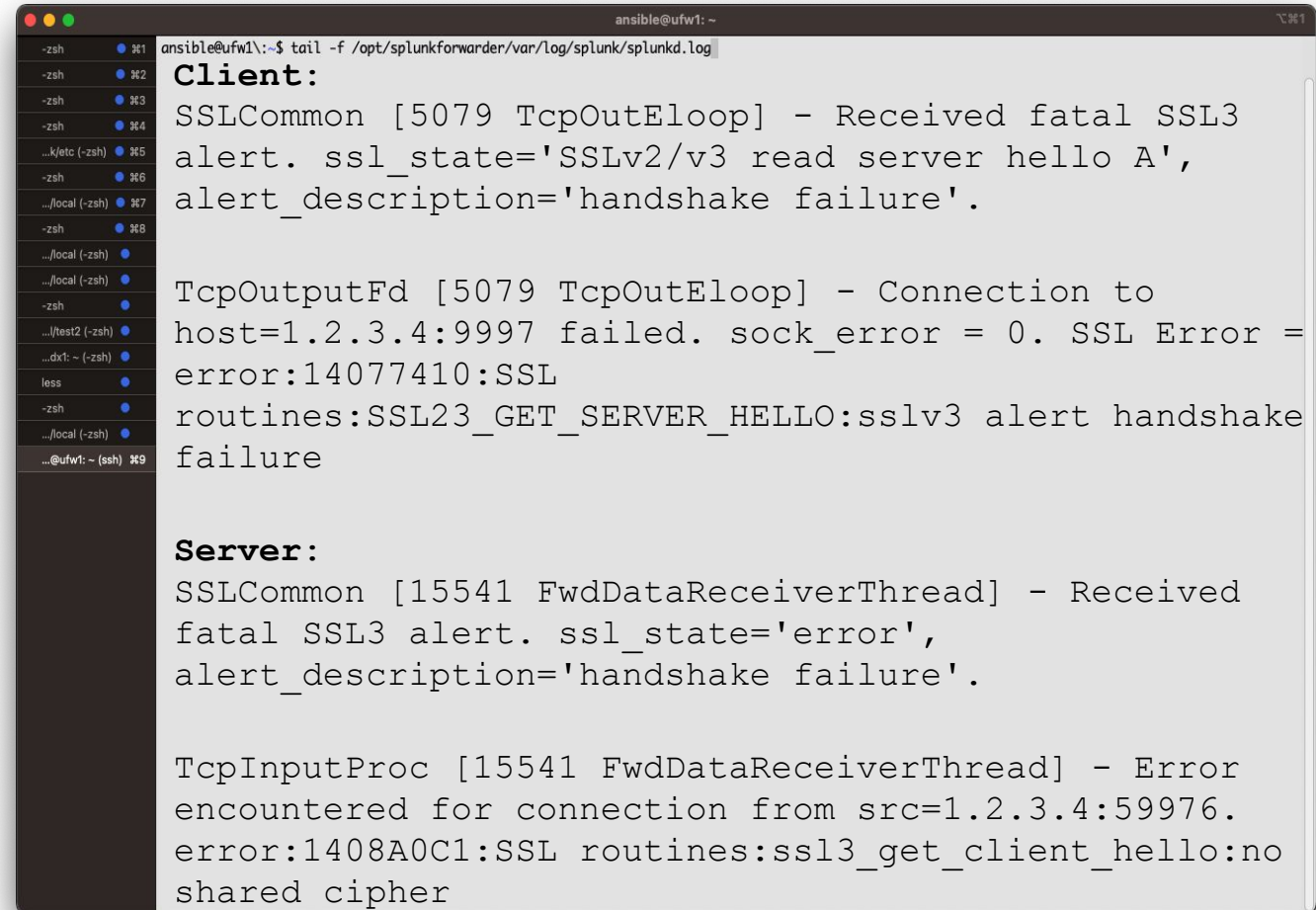
# Mismatched cipherSuite

## Problem

- FWD and IDX can't agree on a cipher suite

## Fix

- Add list of appropriate ciphers to cipherSuite



```
ansible@ufw1:~$ tail -f /opt/splunkforwarder/var/log/splunk/splunkd.log
Client:
SSLCommon [5079 TcpOutEloop] - Received fatal SSL3
alert. ssl_state='SSLv2/v3 read server hello A',
alert_description='handshake failure'.

TcpOutputFd [5079 TcpOutEloop] - Connection to
host=1.2.3.4:9997 failed. sock_error = 0. SSL Error =
error:14077410:SSL
routines:SSL23_GET_SERVER_HELLO:sslv3 alert handshake
failure

Server:
SSLCommon [15541 FwdDataReceiverThread] - Received
fatal SSL3 alert. ssl_state='error',
alert_description='handshake failure'.

TcpInputProc [15541 FwdDataReceiverThread] - Error
encountered for connection from src=1.2.3.4:59976.
error:1408A0C1:SSL routines:ssl3_get_client_hello:no
shared cipher
```



# The Lesser Known Stuff

Hope it's 'never known' to you

# Unexpected Behavior

## requireClientCert

---

- You can not use this setting if you set a passphrase on the private key of any connecting client
- Remember that Splunk Web is a client of Splunkd

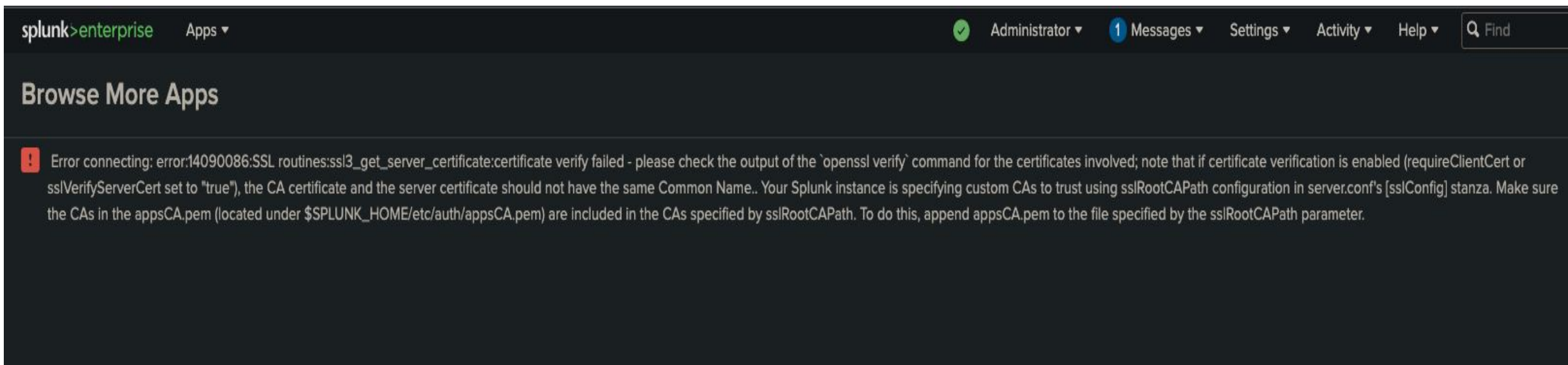
## FIPS KVstore

---

- You must configure the [kvstore] stanza explicitly with these two settings at minimum:
  - serverCert
  - sslPassword (even if set to blank value)

# App Browser

- You must append `$SPLUNK_HOME/etc/auth/appsCA.pem` to your `sslRootCAPath` in `server.conf`



The screenshot shows the Splunk App Browser interface. At the top, there is a navigation bar with the text "splunk>enterprise" and "Apps" with a dropdown arrow. On the right side of the navigation bar, there are several menu items: a green checkmark icon, "Administrator" with a dropdown arrow, "1 Messages" with a dropdown arrow, "Settings" with a dropdown arrow, "Activity" with a dropdown arrow, and "Help" with a dropdown arrow. A search bar with a magnifying glass icon and the text "Find" is also present.

Below the navigation bar, the main content area has the heading "Browse More Apps". Below this heading, there is an error message icon (a red exclamation mark in a square) followed by the text: "Error connecting: error:14090086:SSL routines:ssl3\_get\_server\_certificate:certificate verify failed - please check the output of the `openssl verify` command for the certificates involved; note that if certificate verification is enabled (requireClientCert or sslVerifyServerCert set to "true"), the CA certificate and the server certificate should not have the same Common Name.. Your Splunk instance is specifying custom CAs to trust using sslRootCAPath configuration in server.conf's [sslConfig] stanza. Make sure the CAs in the appsCA.pem (located under \$SPLUNK\_HOME/etc/auth/appsCA.pem) are included in the CAs specified by sslRootCAPath. To do this, append appsCA.pem to the file specified by the sslRootCAPath parameter."



# Indexer Discovery

- Forwarder - server.conf - [sslConfig] - sslVerifyServerCert = true
  - Applies to the Cluster Manager connection
- Forwarder - outputs.conf - [tcpout:<name>] - sslVerifyServerCert = true
  - Applies to the connection to the indexer (not CM).
  - Regardless of what URI's you have configured, IP is what will be used to connect to the indexers. You **must** have IP address in the SAN list for the cert on the indexer.

# Now Go!

1

## Determine where you'll use TLS

Everything or individual functions?

2

## Gather your certs, private keys, and issuers

Consider how they'll be deployed

3

## Enable TLS

First in Dev/Test, then Prod

4

## Enable Verification

First in Dev/Test, then Prod

# Need more?

Go check out the Lantern Documentation by Nick!

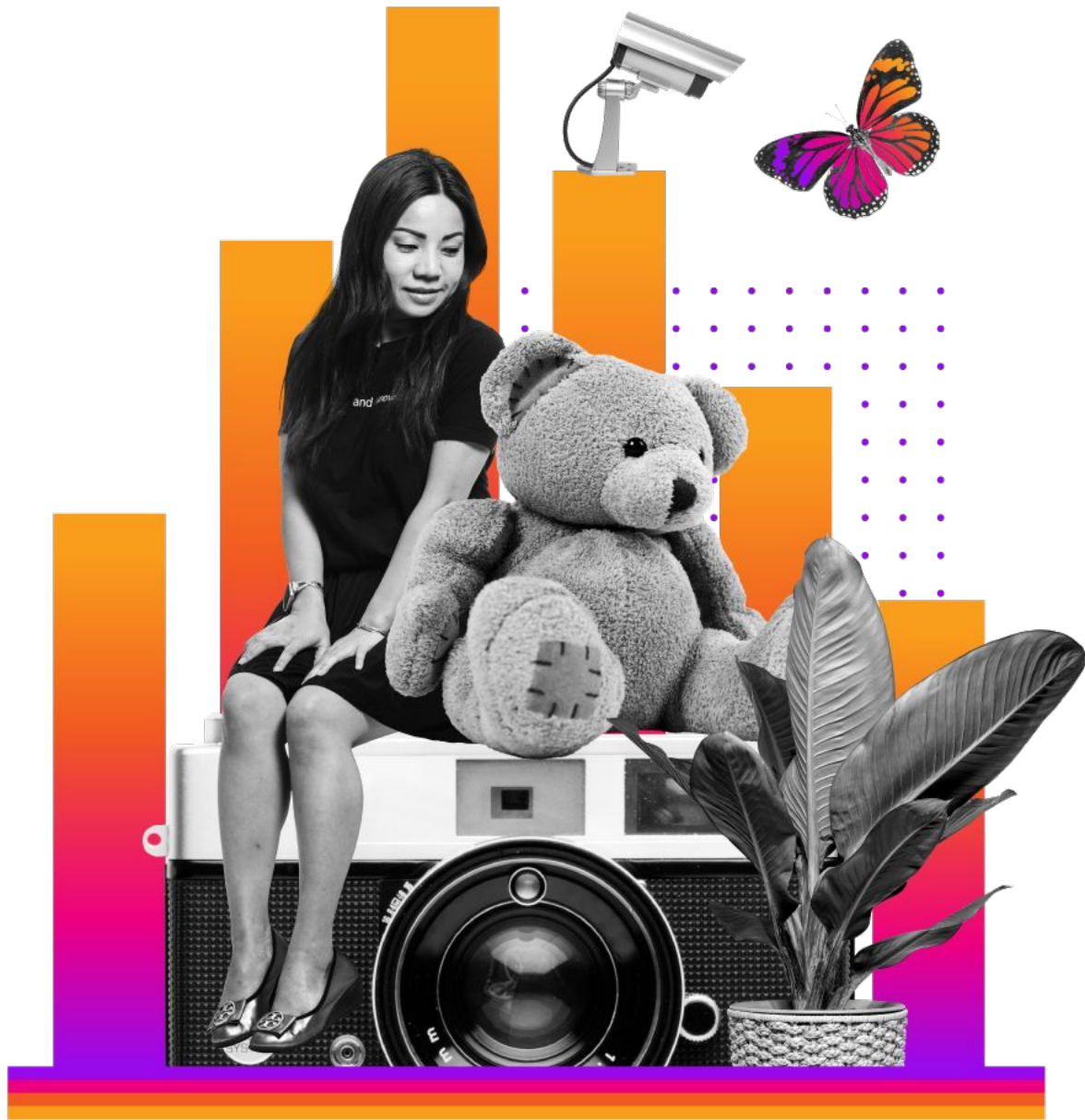
## Splunk Lantern Customer Success Center

Let Splunk experts light your path toward gaining valuable data insights, achieving your key use cases, and managing Splunk more efficiently.

[Click here to learn more.](#)

Securing the Splunk platform with TLS





**“If Nobody Asked Questions, Then We Would Never Learn Anything.”**

splunk> **.conf23**

# Thank You

