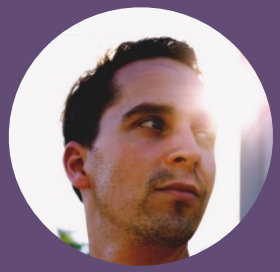
zentral
Development & Professional Services

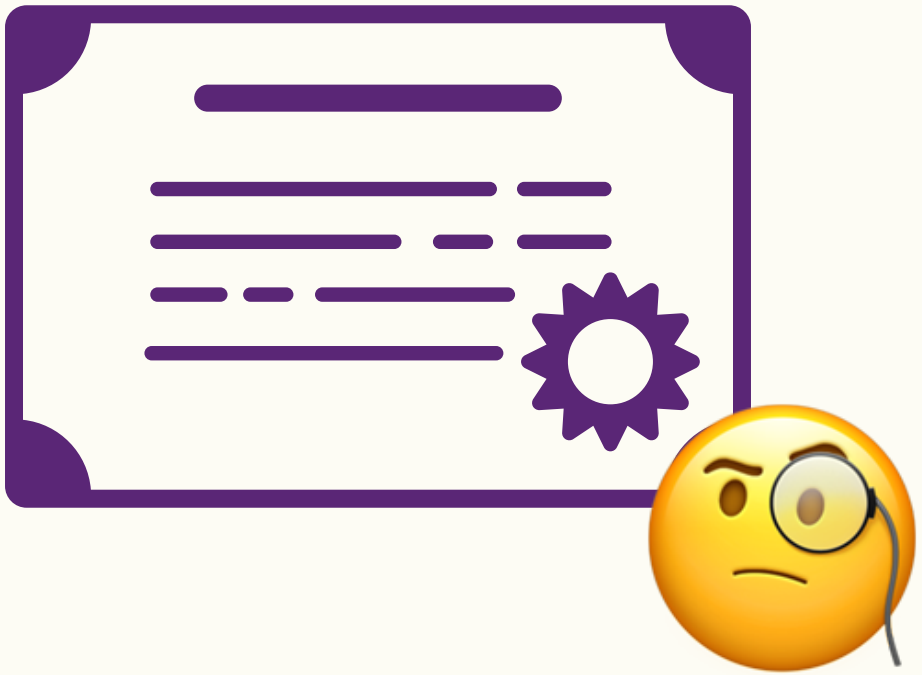
What about mutual TLS ?



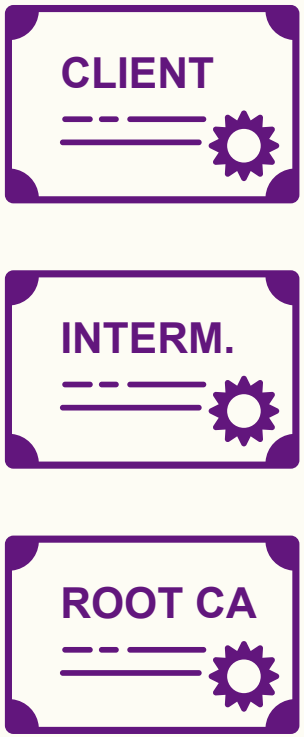
Henry Stamerjohann, October 5-8 2021

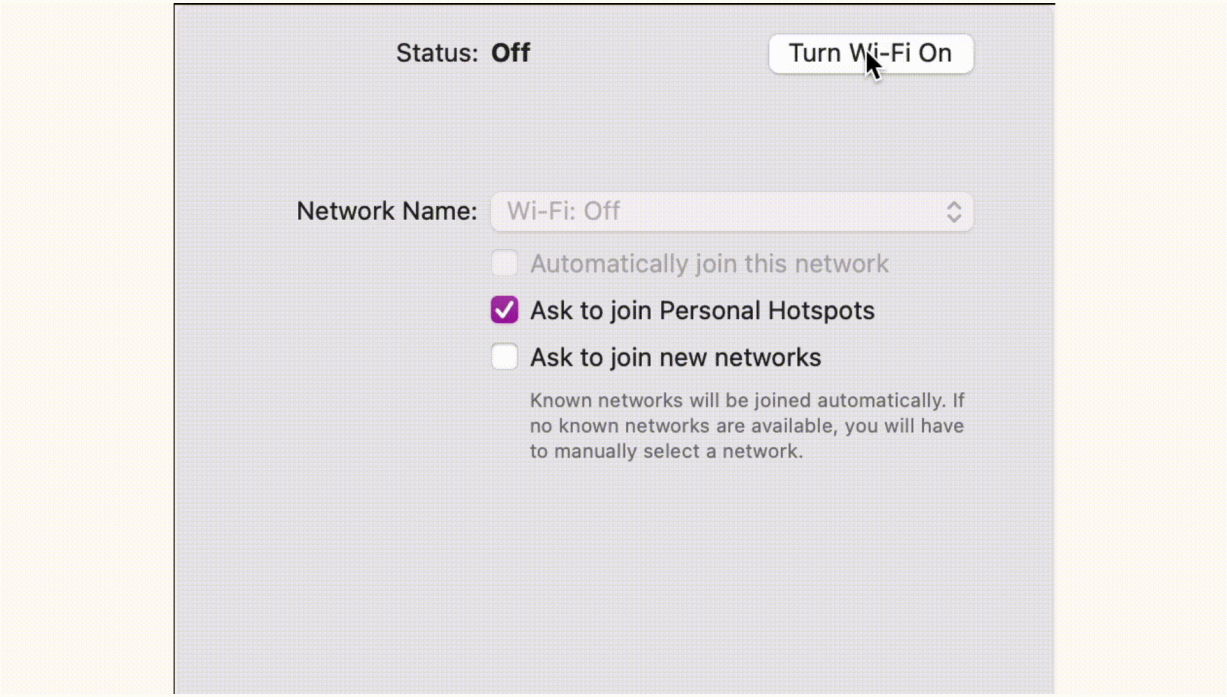
https://zentral.pro

What about mutual TLS ?

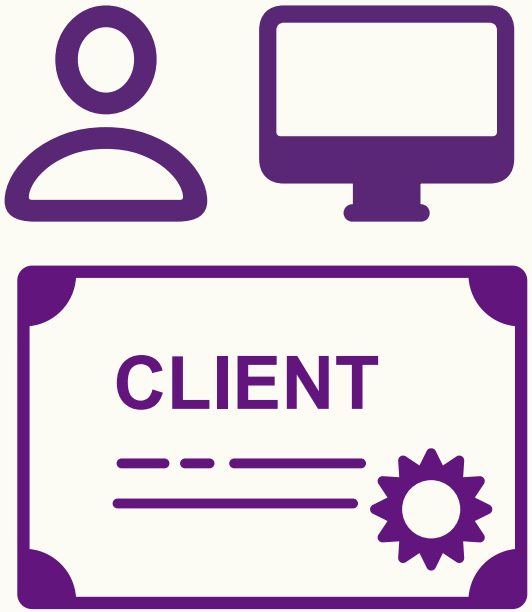


What about mutual TLS ?

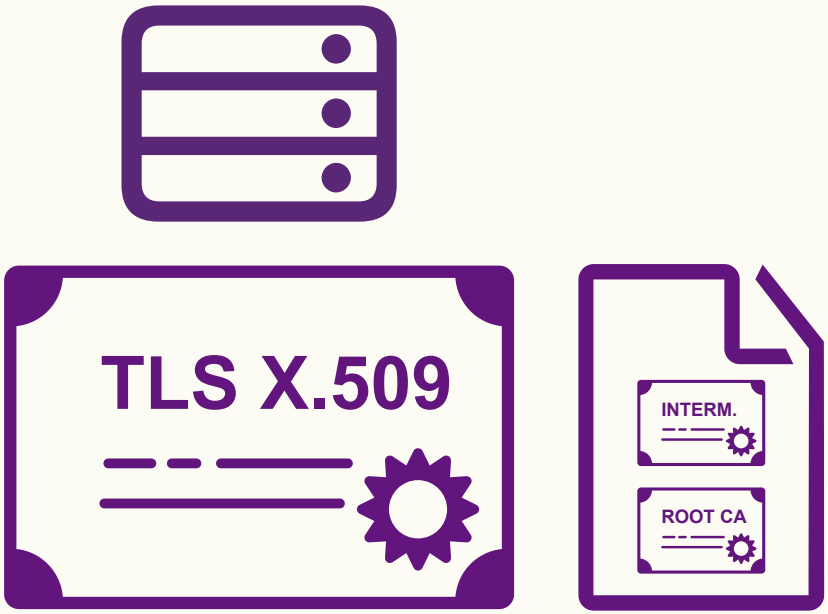




What about mutual TLS ?



Extension	Extended Key Usage (2.5.29.37)
Critical	NO
Purpose #1	Client Authentication (1.3.6.1.5.5.7.3.2)



Extension	Subject Alternative Name (2.5.29.17)
Critical	NO
DNS Name	*.jamfcloud.com

Where is it used?

What we can achieve with client certificates?

MDM client side authentication

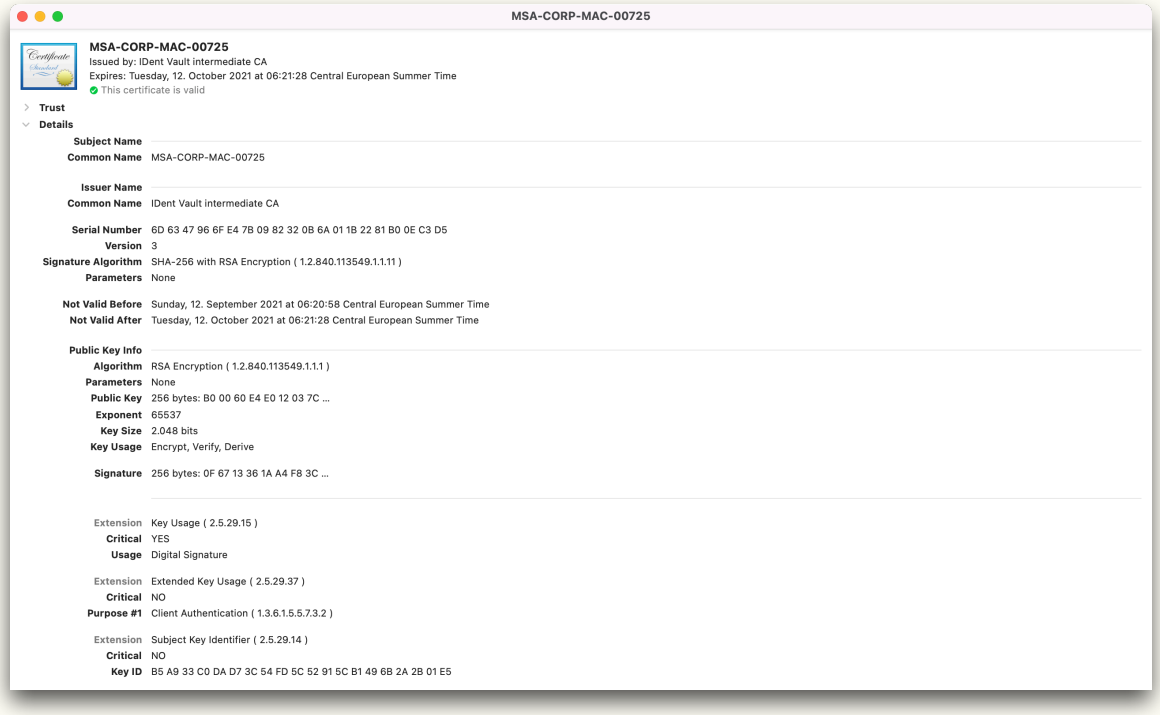
VPN tunnel solutions

Network access control

Interface services / API’s

Identity attestation

Zero Trust setups



Where is it used?

What we can achieve with client certificates?

MDM client side authentication

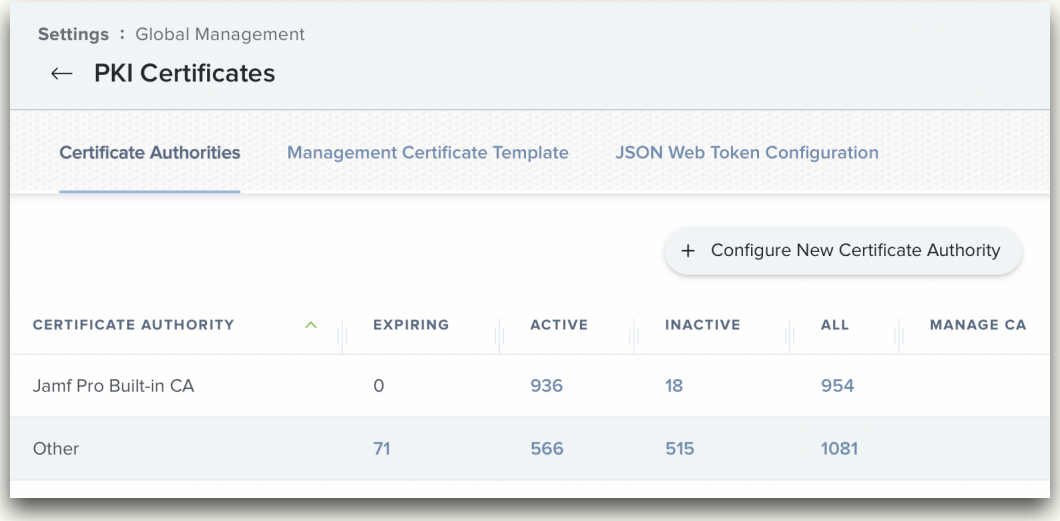
VPN tunnel solutions

Network access control

Interface services / API’s

Identity attestation

Zero Trust setups



Certificates in depth

Explained in excellent detail

Certificates - How Do They Work?
by Marko Jung (MacSysAdmin 2016)

Raiders of the Lost Certificate
by Paul Suh (MacSysAdmin 2012)

http://docs.macsysadmin.se/2020/

Overview

Client certificate authentication in brief

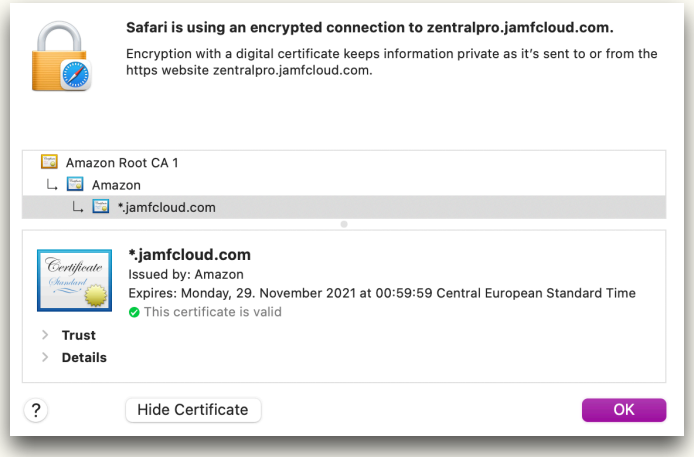
TLS - Transport Layer Security

PKI - Public Key Infrastructure

CA - Certificate Authority

X.509 - identity, public/private keys

Identity - DNS Name, Email (RFC822)



Overview

Client certificate authentication in brief

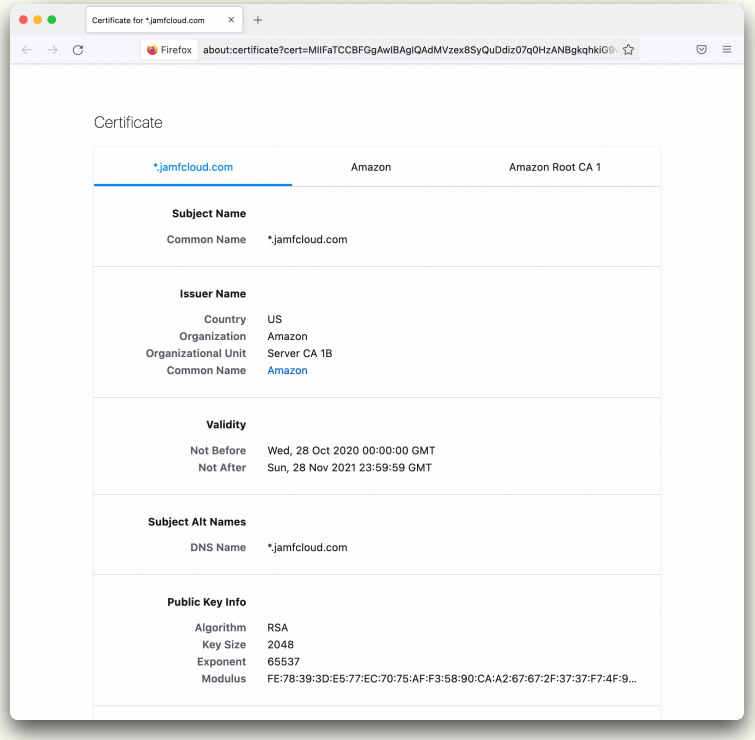
TLS - Transport Layer Security

PKI - Public Key Infrastructure

CA - Certificate Authority

X.509 - identity, public/private keys

Identity - DNS Name, Email (RFC822)



Overview

Client certificate authentication in brief

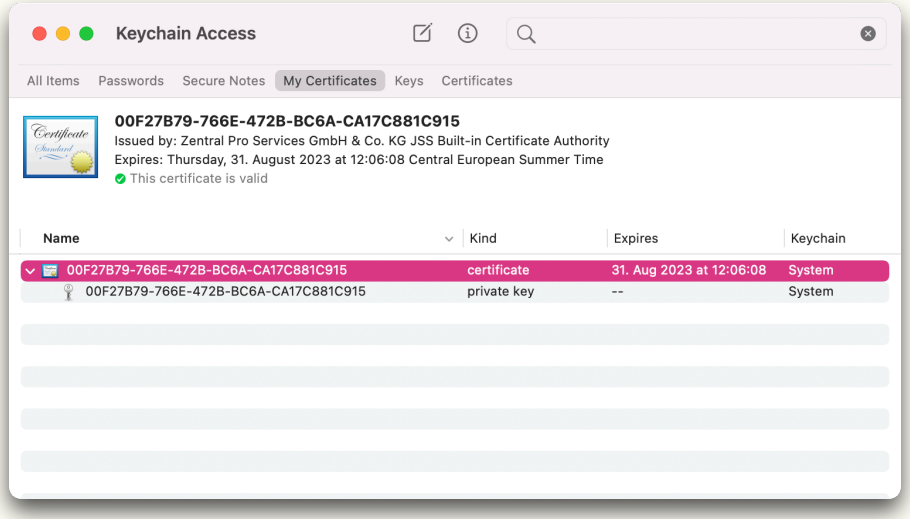
TLS - Transport Layer Security

PKI - Public Key Infrastructure

CA - Certificate Authority

X.509 - identity, public/private keys

Identity - DNS Name, Email (RFC822)



Overview (cont.)

Client certificate authentication in brief

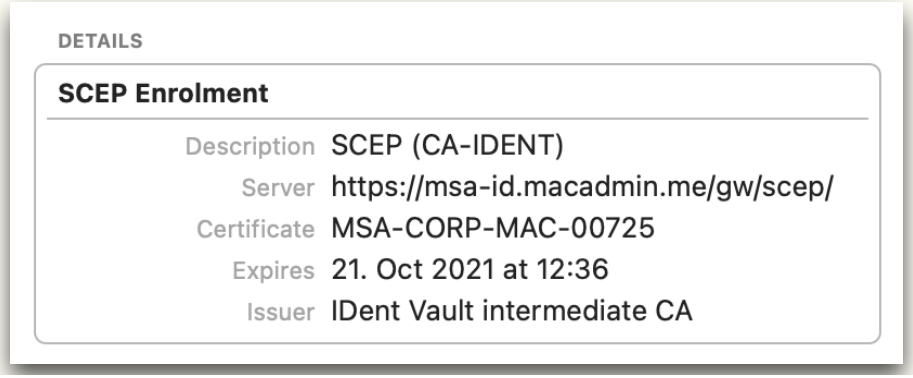
Client certificate - a digital certificate (conforms to the X.509 system)

Authentication - proves the identity in a communication

Certificate extensions - add information to the certificate

TLS Handshake - starts communication session w/ TLS encryption

CA certificates list - acceptable client certificate CA names



Overview (cont.)

Client certificate authentication in brief

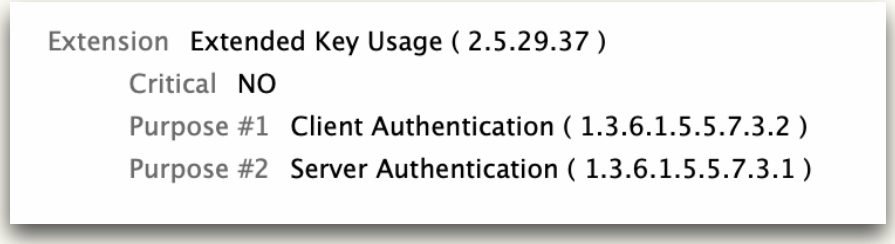
Client certificate - a digital certificate (conforms to the X.509 system)

Authentication - proves the identity in a communication

Certificate extensions - add information to the certificate

TLS Handshake - starts communication session w/ TLS encryption

CA certificates list - acceptable client certificate CA names



Extended Key Usage

Client authentication

Identifies one ore more purposes, in addition to the basic purposes of a certificate

- Client authentication OID

Extension	Extended Key Usage (2.5.29.37)
Critical	NO
Purpose #1	Client Authentication (1.3.6.1.5.5.7.3.2)

Usage Definitions for the Extended Key Usage Extension

Usage	OID
Server authentication	1.3.6.1.5.5.7.3.1
Client authentication	1.3.6.1.5.5.7.3.2
Code signing	1.3.6.1.5.5.7.3.3
Email	1.3.6.1.5.5.7.3.4
IPsec end system	1.3.6.1.5.5.7.3.5
IPsec tunnel	1.3.6.1.5.5.7.3.6
IPsec user	1.3.6.1.5.5.7.3.7
Timestamping	1.3.6.1.5.5.7.3.8

Assumption

Client certificate variations

User client certificate

Device client certificate

Attributes: **?** **?**

Assumption

Client certificate variations

User client certificate

Attributes:

Username
Email

Device client certificate

DNS Name

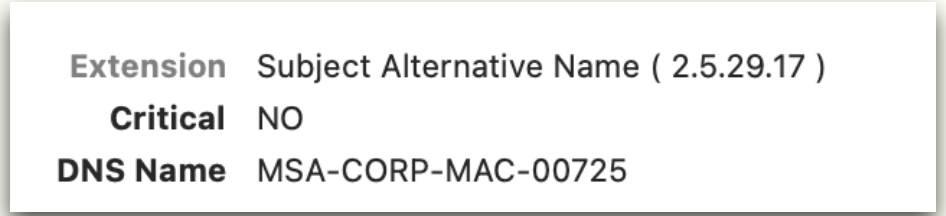
Certificate extensions

Subject Alternative Name (SAN)

The Subject Alternative Name (SAN) is an certificate extension that binds additional information to the Subject Distinguished Name of a certificate.

Key information includes:

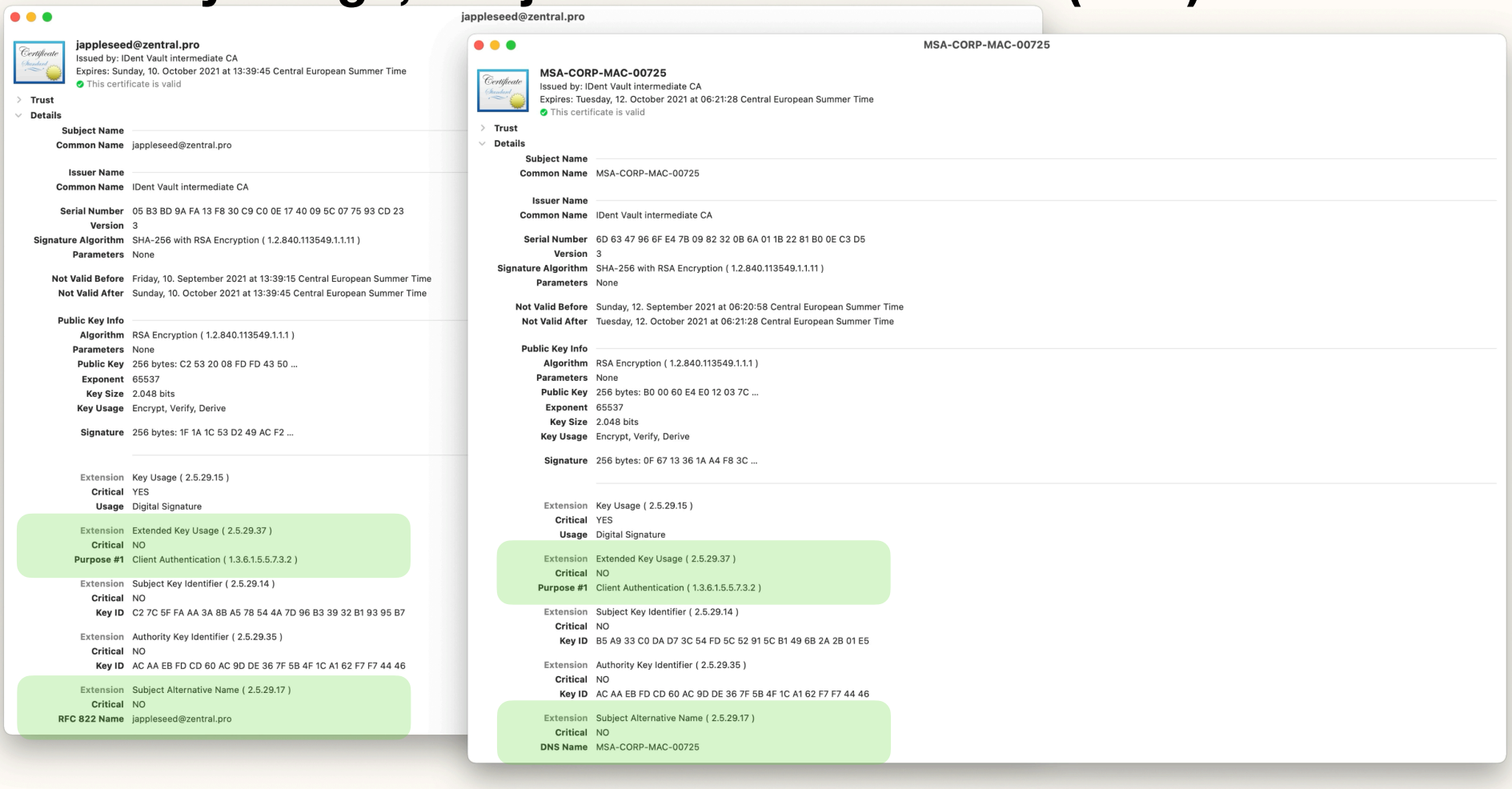
- DNS Name
- RFC822 Name (email address)



Subject Distinguished Name (Subject DN) = the unique identifier

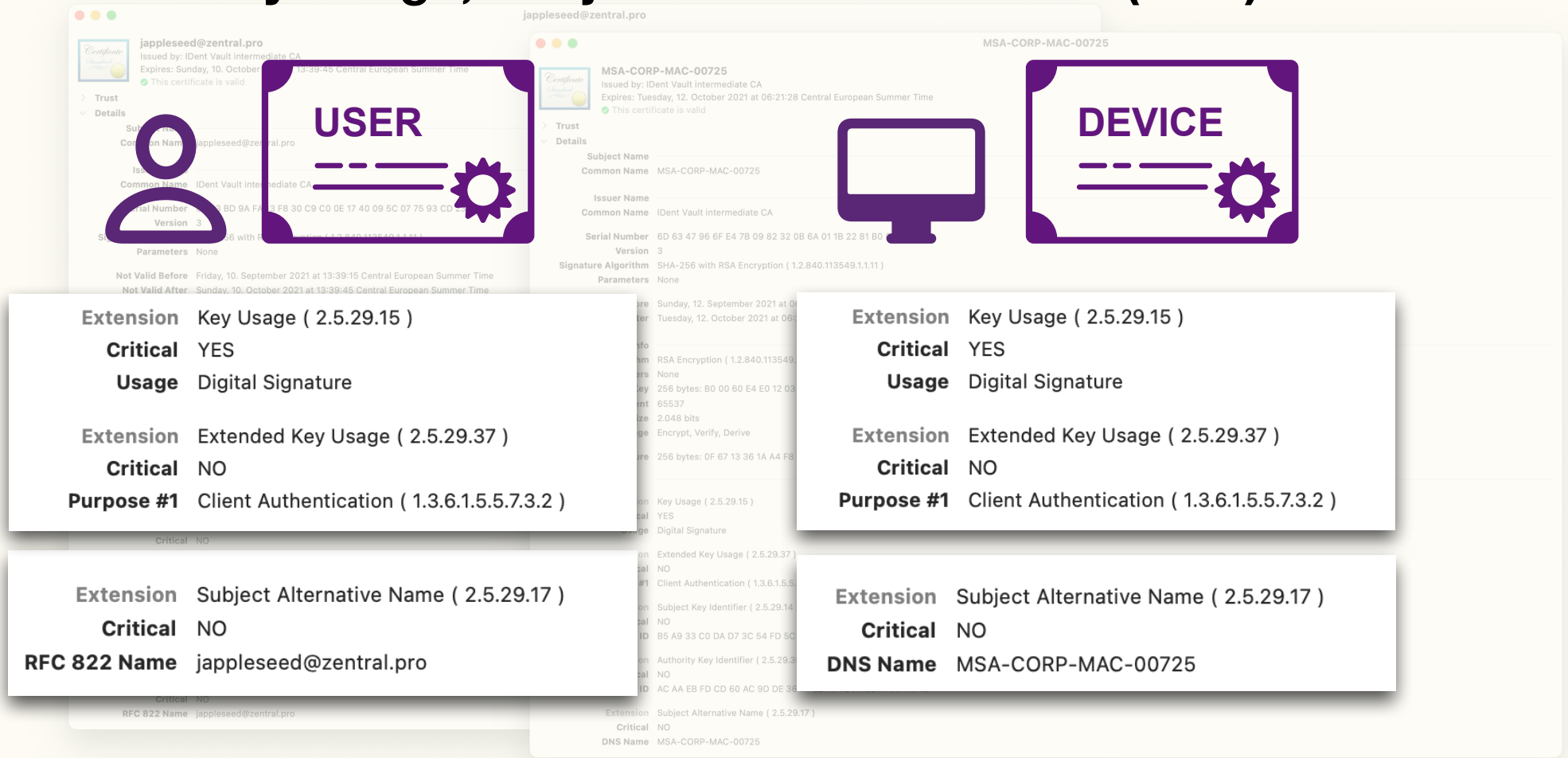
X.509 v3 certificate extension

Extended Key Usage, Subject Alternative Name (SAN)



X.509 v3 certificate extension

Extended Key Usage, Subject Alternative Name (SAN)



TLS Handshake

Starting communication session

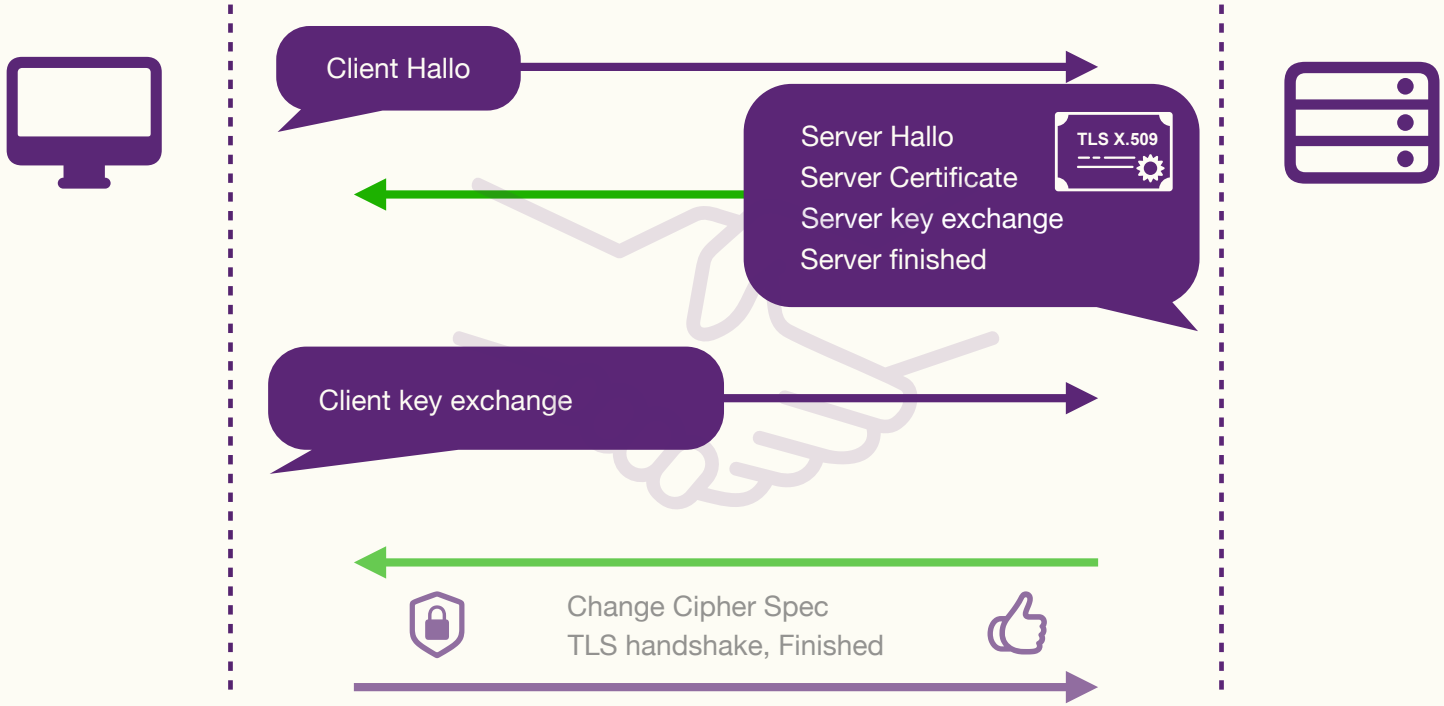
7. The TLS Handshaking Protocols

TLS has three subprotocols that are used to allow peers to agree upon security parameters for the record layer, to authenticate themselves, to instantiate negotiated security parameters, and to report error conditions to each other.

The Handshake Protocol is responsible for negotiating a session, which consists of the following items:

TLS handshake

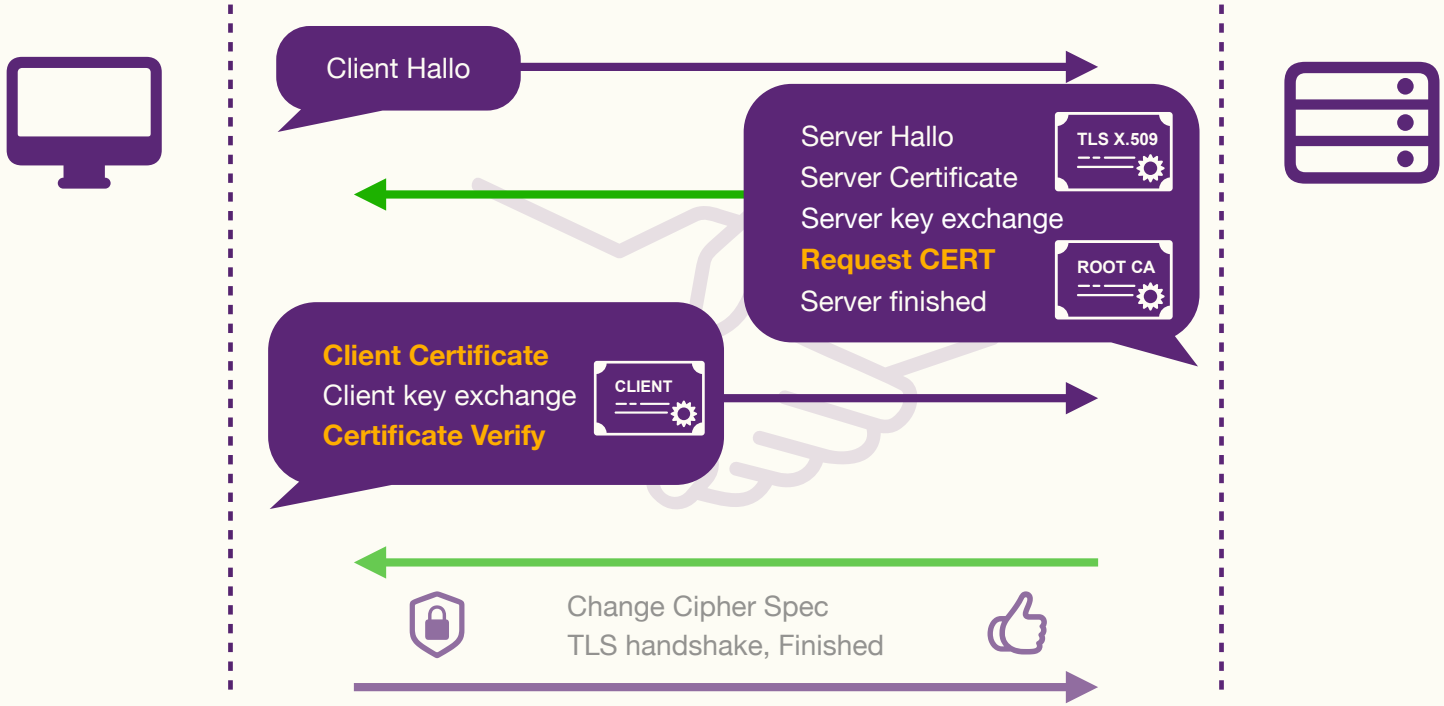
Server side encryption



https://megamorf.gitlab.io/2020/03/03/traffic-analysis-of-a-tls-session/

Two-way authentication

Server and client authentication



https://megamorf.gitlab.io/2020/03/03/traffic-analysis-of-a-tls-session/

TLS / mTLS handshake

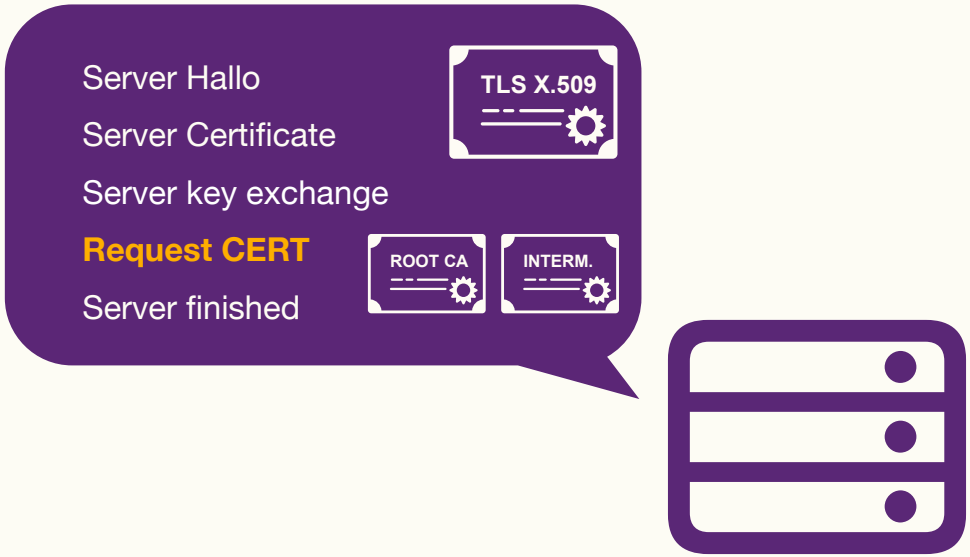
Side by side comparison


```
# TLS handshake
* TLSv1.2 (OUT), TLS handshake, Client hello (1):
* TLSv1.2 (IN),  TLS handshake, Server hello (2):
* TLSv1.2 (IN),  TLS handshake, Certificate (11):
* TLSv1.2 (IN),  TLS handshake, Server key exchange (12):
* TLSv1.2 (IN),  TLS handshake, Server finished (14):
* TLSv1.2 (OUT), TLS handshake, Client key exchange (16):
* TLSv1.2 (OUT), TLS change cipher, Change cipher spec (1):
* TLSv1.2 (OUT), TLS handshake, Finished (20):
* TLSv1.2 (IN),  TLS change cipher, Change cipher spec (1):
* TLSv1.2 (IN),  TLS handshake, Finished (20):
```


```
# Mutual TLS (client certificate auth)
* TLSv1.2 (OUT), TLS handshake, Client hello (1):
* TLSv1.2 (IN),  TLS handshake, Server hello (2):
* TLSv1.2 (IN),  TLS handshake, Certificate (11):
* TLSv1.2 (IN),  TLS handshake, Server key exchange (12):
* TLSv1.2 (IN),  TLS handshake, Request CERT (13):
* TLSv1.2 (IN),  TLS handshake, Server finished (14):
* TLSv1.2 (OUT), TLS handshake, Certificate (11):
* TLSv1.2 (OUT), TLS handshake, Client key exchange (16):
* TLSv1.2 (OUT), TLS handshake, CERT verify (15):
* TLSv1.2 (OUT), TLS change cipher, Change cipher spec (1):
* TLSv1.2 (OUT), TLS handshake, Finished (20):
* TLSv1.2 (IN),  TLS change cipher, Change cipher spec (1):
* TLSv1.2 (IN),  TLS handshake, Finished (20):
```

Two-way authentication

TL;DL



CertificateRequest

Used when the server requires client identity authentication

- Asks client for certificate
- Tells which certificate types acceptable
- Indicates which certificate authorities are trustworthy

Two-way authentication

~~TL;DL~~ → rfc5246

datatracker.ietf.org

7.4.4. Certificate Request

When this message will be sent:

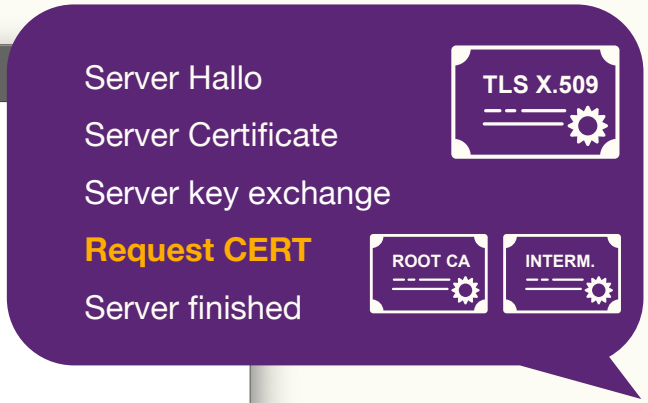
A non-anonymous server can optionally request a certificate from the client, if appropriate for the selected cipher suite. This message, if sent, will immediately follow the ServerKeyExchange message (if it is sent; otherwise, this message follows the server's Certificate message).

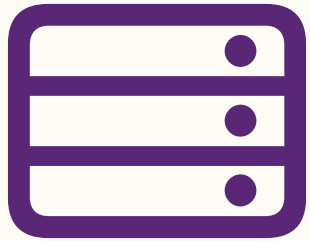
Structure of this message:

```
enum {
    rsa_sign(1), dss_sign(2), rsa_fixed_dh(3), dss_fixed_dh(4),
    rsa_ephemeral_dh_RESERVED(5), dss_ephemeral_dh_RESERVED(6),
    fortezza_dms_RESERVED(20), (255)
} ClientCertificateType;

opaque DistinguishedName<1..2^16-1>;

struct {
    ClientCertificateType certificate_types<1..2^8-1>;
    SignatureAndHashAlgorithm
      supported_signature_algorithms<2^16-1>;
    DistinguishedName certificate_authorities<0..2^16-1>;
} CertificateRequest;
```

certificate_types
A list of the types of certificate types that the client may offer.

rsa_sign	a certificate containing an RSA key
dss_sign	a certificate containing a DSA key
rsa_fixed_dh	a certificate containing a static DH key.
dss_fixed_dh	a certificate containing a static DH key





Look at things

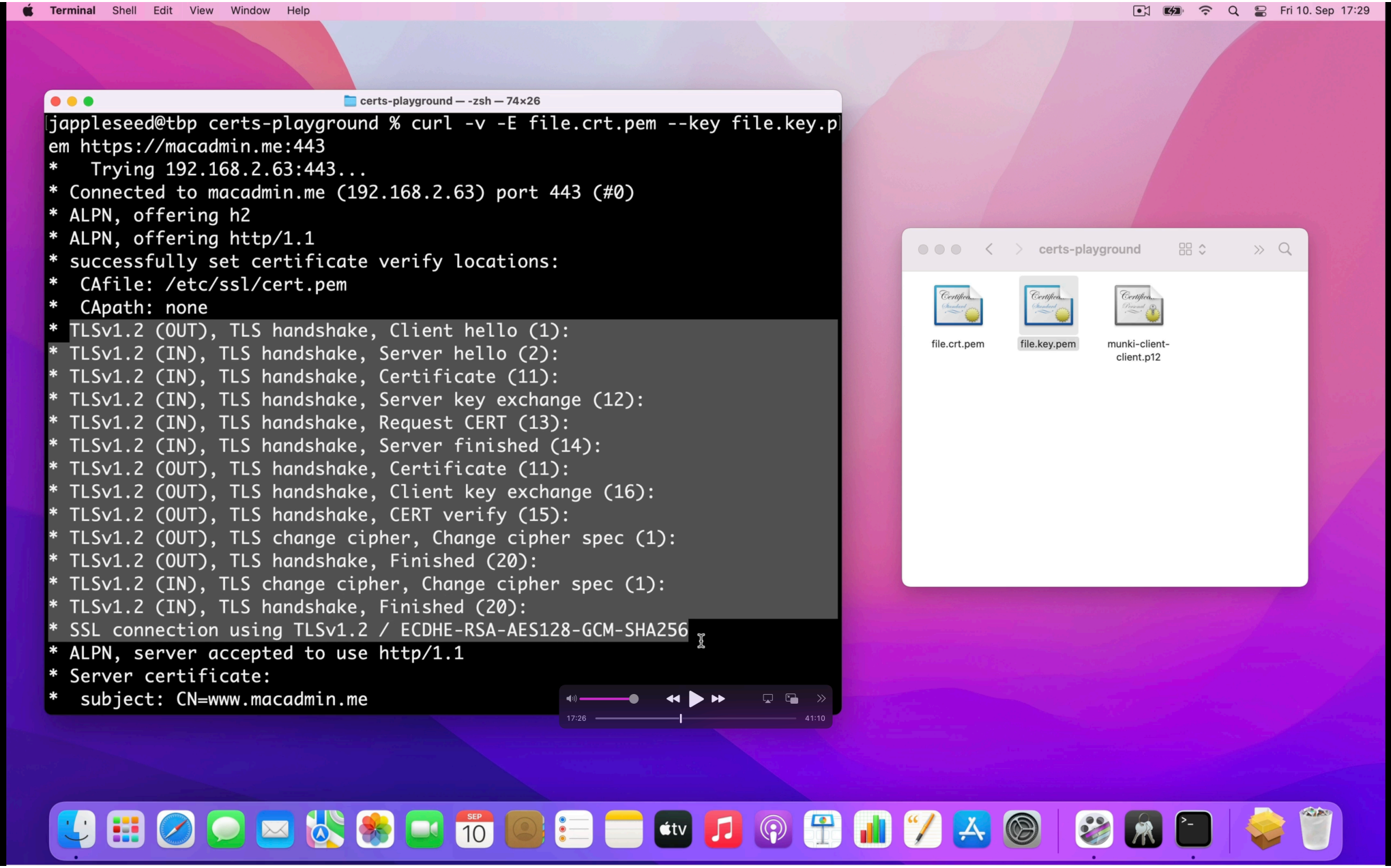
Case study + lab setup

5 + 1

Case study #01

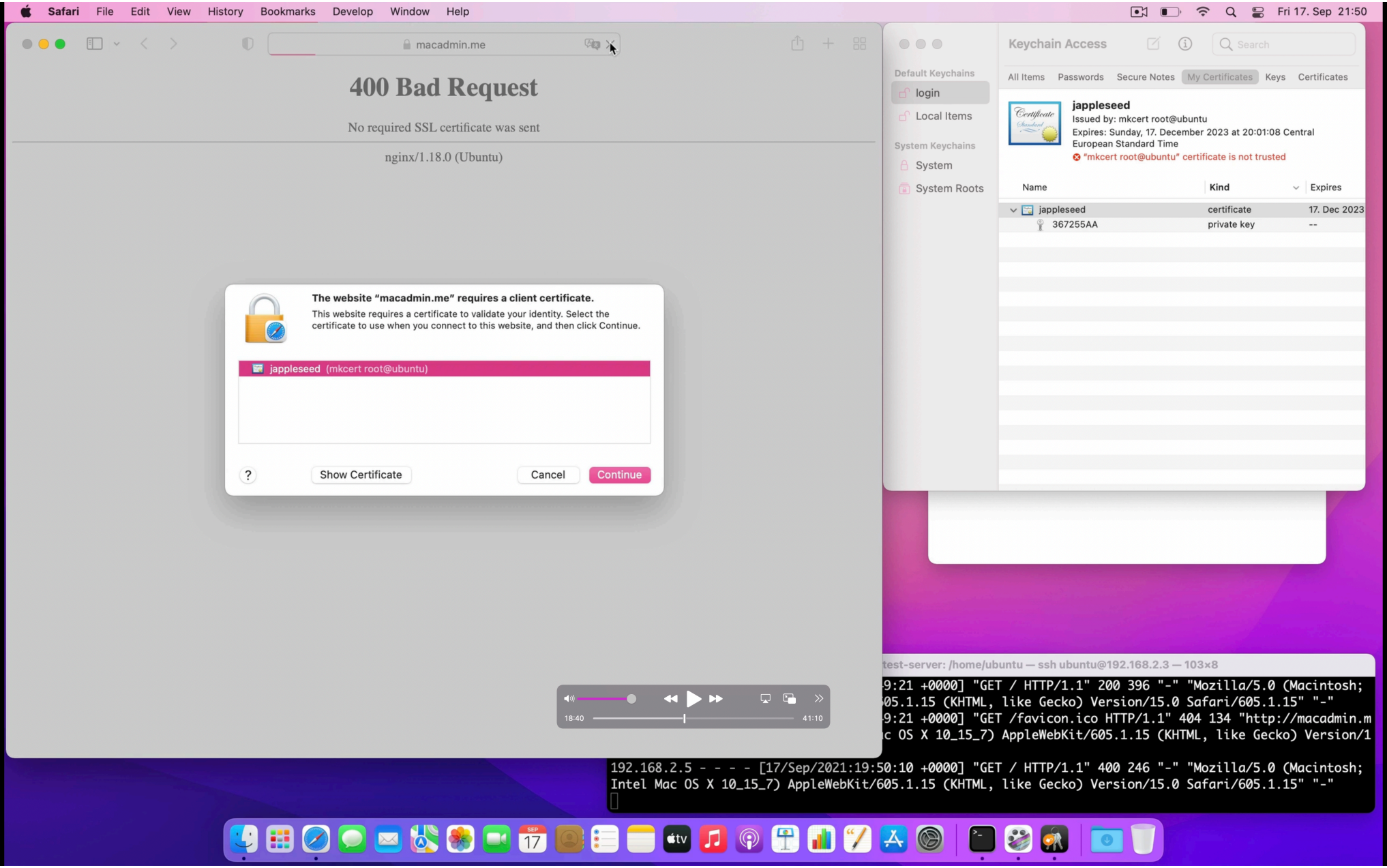
Using curl with client certificate

```
# This is the server SSL certificate
  ssl_certificate       /etc/nginx/certs/macadmin.me.pem;

# This is the server certificate key
  ssl_certificate_key   /etc/nginx/certs/macadmin.me.key;

# This is the CA cert against which the client/user will be validated
  # The Client certificate might be
  # created from a different CA
  ssl_client_certificate /etc/nginx/certs/root.pem;

# Enables mutual TLS/two way SSL to verify the client
  ssl_verify_client on;
```


Terminal Shell Edit View Window Help
Fri 10. Sep 17:29
certs-playground — -zsh — 74×26
jappleseed@tbp certs-playground % curl -v -E file.crt.pem --key file.key.p
em https://macadmin.me:443
*   Trying 192.168.2.63:443...
* Connected to macadmin.me (192.168.2.63) port 443 (#0)
* ALPN, offering h2
* ALPN, offering http/1.1
* successfully set certificate verify locations:
* CAfile: /etc/ssl/cert.pem
* CApath: none
* TLSv1.2 (OUT), TLS handshake, Client hello (1):
* TLSv1.2 (IN), TLS handshake, Server hello (2):
* TLSv1.2 (IN), TLS handshake, Certificate (11):
* TLSv1.2 (IN), TLS handshake, Server key exchange (12):
* TLSv1.2 (IN), TLS handshake, Request CERT (13):
* TLSv1.2 (IN), TLS handshake, Server finished (14):
* TLSv1.2 (OUT), TLS handshake, Certificate (11):
* TLSv1.2 (OUT), TLS handshake, Client key exchange (16):
* TLSv1.2 (OUT), TLS handshake, CERT verify (15):
* TLSv1.2 (OUT), TLS change cipher, Change cipher spec (1):
* TLSv1.2 (OUT), TLS handshake, Finished (20):
* TLSv1.2 (IN), TLS change cipher, Change cipher spec (1):
* TLSv1.2 (IN), TLS handshake, Finished (20):
* SSL connection using TLSv1.2 / ECDHE-RSA-AES128-GCM-SHA256
* ALPN, server accepted to use http/1.1
* Server certificate:
* subject: CN=www.macadmin.me
17:26
41:10
certs-playground
file.crt.pem
file.key.pem
munki-client-client.p12

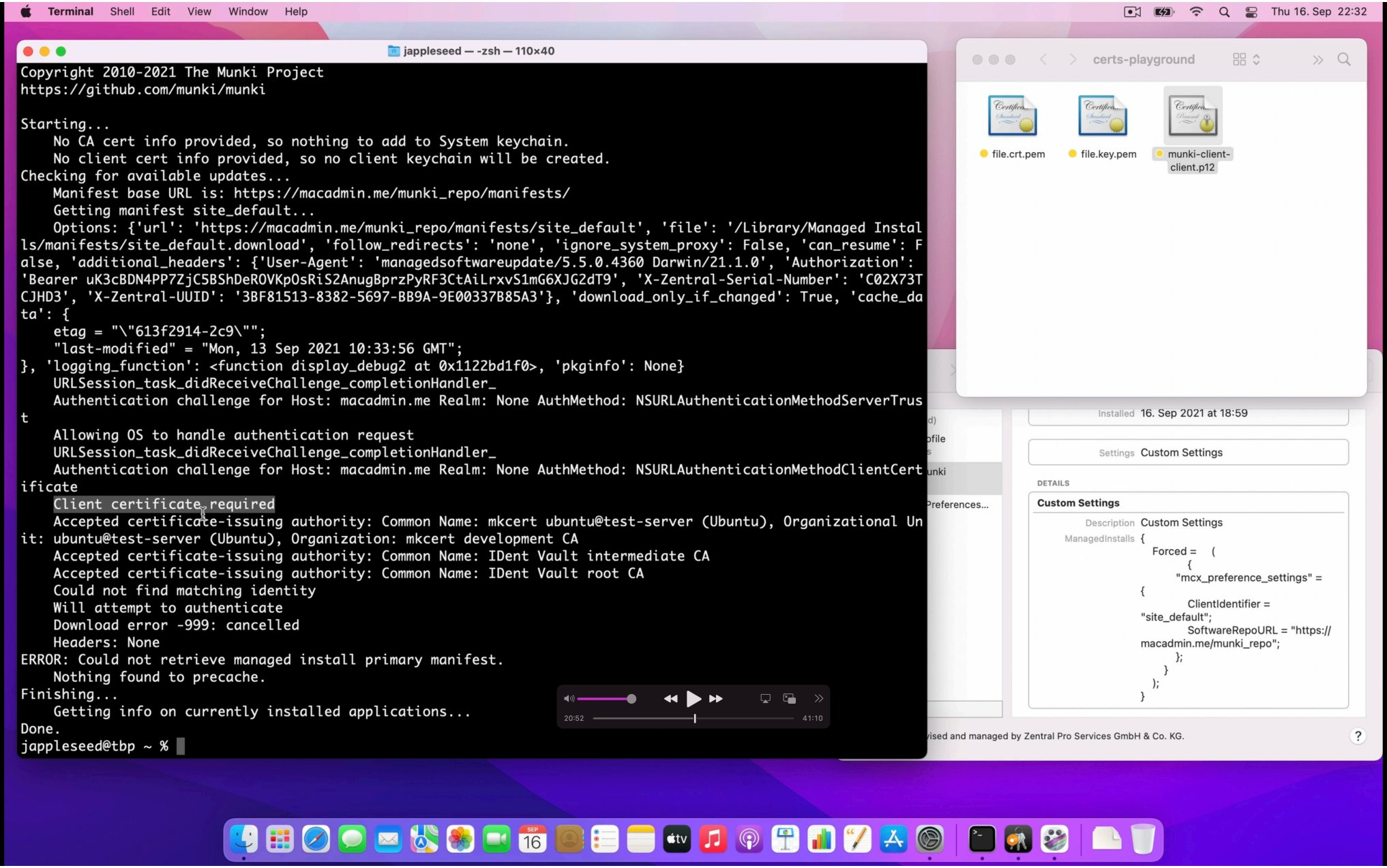
Case study #02

Browser access with client certificate


Safari File Edit View History Bookmarks Develop Window Help
Fri 17. Sep 21:50
macadmin.me
400 Bad Request
No required SSL certificate was sent
nginx/1.18.0 (Ubuntu)
The website "macadmin.me" requires a client certificate.
This website requires a certificate to validate your identity. Select the certificate to use when you connect to this website, and then click Continue.
jappleseed (mkcert root@ubuntu)
Show Certificate
Cancel
Continue
Keychain Access
Search
Default Keychains
login
Local Items
System Keychains
System
System Roots
All Items
Passwords
Secure Notes
My Certificates
Keys
Certificates
jappleseed
Issued by: mkcert root@ubuntu
Expires: Sunday, 17. December 2023 at 20:01:08 Central European Standard Time
"mkcert root@ubuntu" certificate is not trusted
Name
Kind
Expires
jappleseed
certificate
17. Dec 2023
367255AA
private key
--
18:40
41:10
test-server: /home/ubuntu — ssh ubuntu@192.168.2.3 — 103×8
9:21 +0000] "GET / HTTP/1.1" 200 396 "-" "Mozilla/5.0 (Macintosh;
05.1.15 (KHTML, like Gecko) Version/15.0 Safari/605.1.15" "-"
9:21 +0000] "GET /favicon.ico HTTP/1.1" 404 134 "http://macadmin.m
c OS X 10_15_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/1
192.168.2.5 - - - - [17/Sep/2021:19:50:10 +0000] "GET / HTTP/1.1" 400 246 "-" "Mozilla/5.0 (Macintosh;
Intel Mac OS X 10_15_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/15.0 Safari/605.1.15" "-"

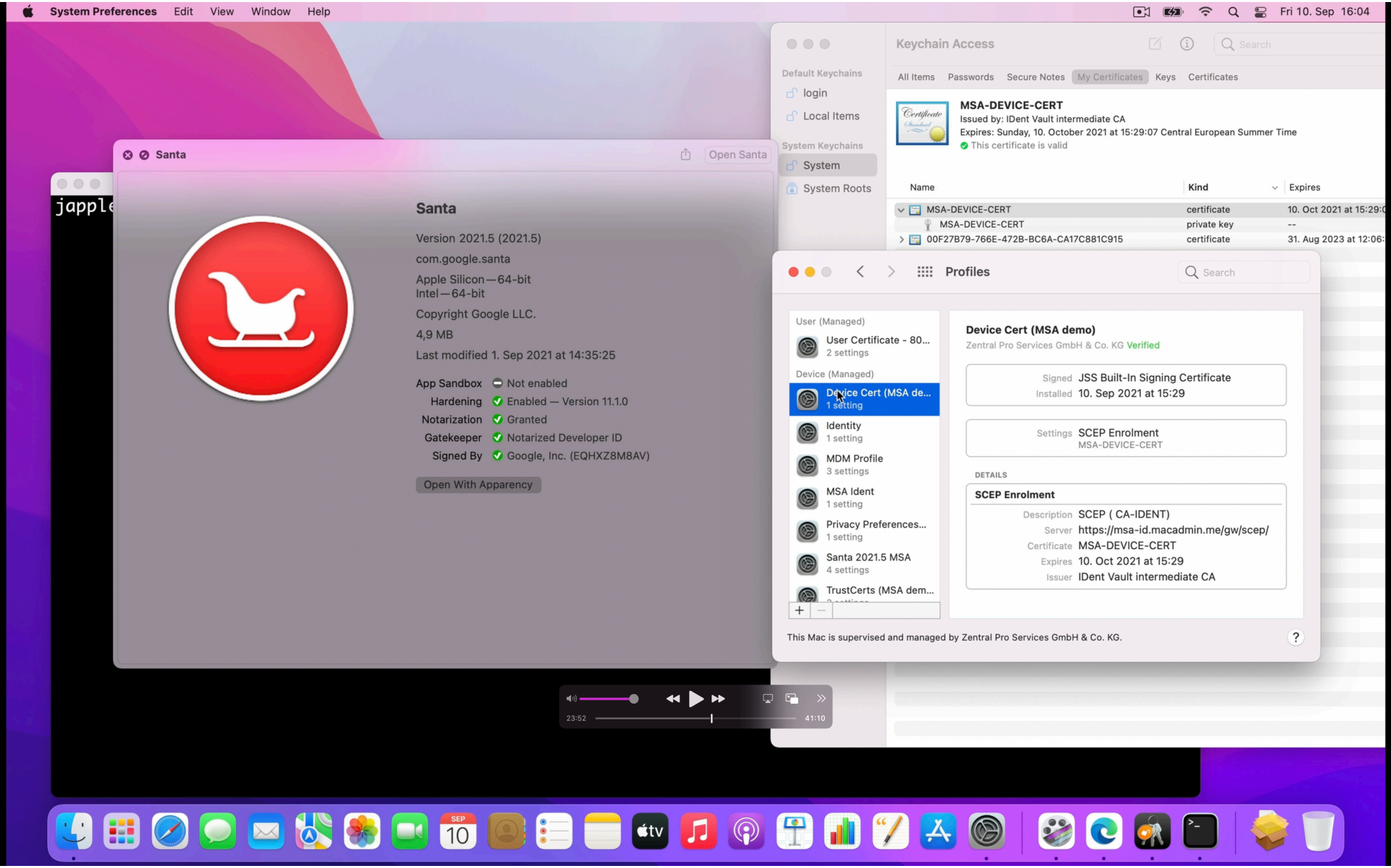
Case study #03

Munki access with client certificate


Terminal Shell Edit View Window Help
Thu 16. Sep 22:32
jappleseed — -zsh — 110×40
Copyright 2010-2021 The Munki Project
https://github.com/munki/munki
Starting...
No CA cert info provided, so nothing to add to System keychain.
No client cert info provided, so no client keychain will be created.
Checking for available updates...
Manifest base URL is: https://macadmin.me/munki_repo/manifests/
Getting manifest site_default...
Options: {'url': 'https://macadmin.me/munki_repo/manifests/site_default', 'file': '/Library/Managed Installs/manifests/site_default.download', 'follow_redirects': 'none', 'ignore_system_proxy': False, 'can_resume': False, 'additional_headers': {'User-Agent': 'managedsoftwareupdate/5.5.0.4360 Darwin/21.1.0', 'Authorization': 'Bearer uK3cBDN4PP7ZjC5BShDeROVKpOsRiS2AnugBprzPyRF3CtAiLrxvS1mG6XJG2dT9', 'X-Zentral-Serial-Number': 'C02X73TCJHD3', 'X-Zentral-UUID': '3BF81513-8382-5697-BB9A-9E00337B85A3'}, 'download_only_if_changed': True, 'cache_data': {
etag = "\"613f2914-2c9\"";
"last-modified" = "Mon, 13 Sep 2021 10:33:56 GMT";
}, 'logging_function': <function display_debug2 at 0x1122bd1f0>, 'pkginfo': None}
URLSession_task_didReceiveChallenge_completionHandler_
Authentication challenge for Host: macadmin.me Realm: None AuthMethod: NSURLAuthenticationMethodServerTrust
Allowing OS to handle authentication request
URLSession_task_didReceiveChallenge_completionHandler_
Authentication challenge for Host: macadmin.me Realm: None AuthMethod: NSURLAuthenticationMethodClientCertificate
Client certificate required
Accepted certificate-issuing authority: Common Name: mkcert ubuntu@test-server (Ubuntu), Organizational Unit: ubuntu@test-server (Ubuntu), Organization: mkcert development CA
Accepted certificate-issuing authority: Common Name: IDent Vault intermediate CA
Accepted certificate-issuing authority: Common Name: IDent Vault root CA
Could not find matching identity
Will attempt to authenticate
Download error -999: cancelled
Headers: None
ERROR: Could not retrieve managed install primary manifest.
Nothing found to precache.
Finishing...
Getting info on currently installed applications...
Done.
jappleseed@tbp ~ %
20:52
41:10
certs-playground
file.crt.pem
file.key.pem
munki-client-client.p12
Installed 16. Sep 2021 at 18:59
Settings Custom Settings
DETAILS
Custom Settings
Description Custom Settings
ManagedInstalls {
Forced = (
{
"mcx_preference_settings" = {
ClientIdentifier = "site_default";
SoftwareRepoURL = "https://macadmin.me/munki_repo";
};
}
);
}
vised and managed by Zentral Pro Services GmbH & Co. KG.

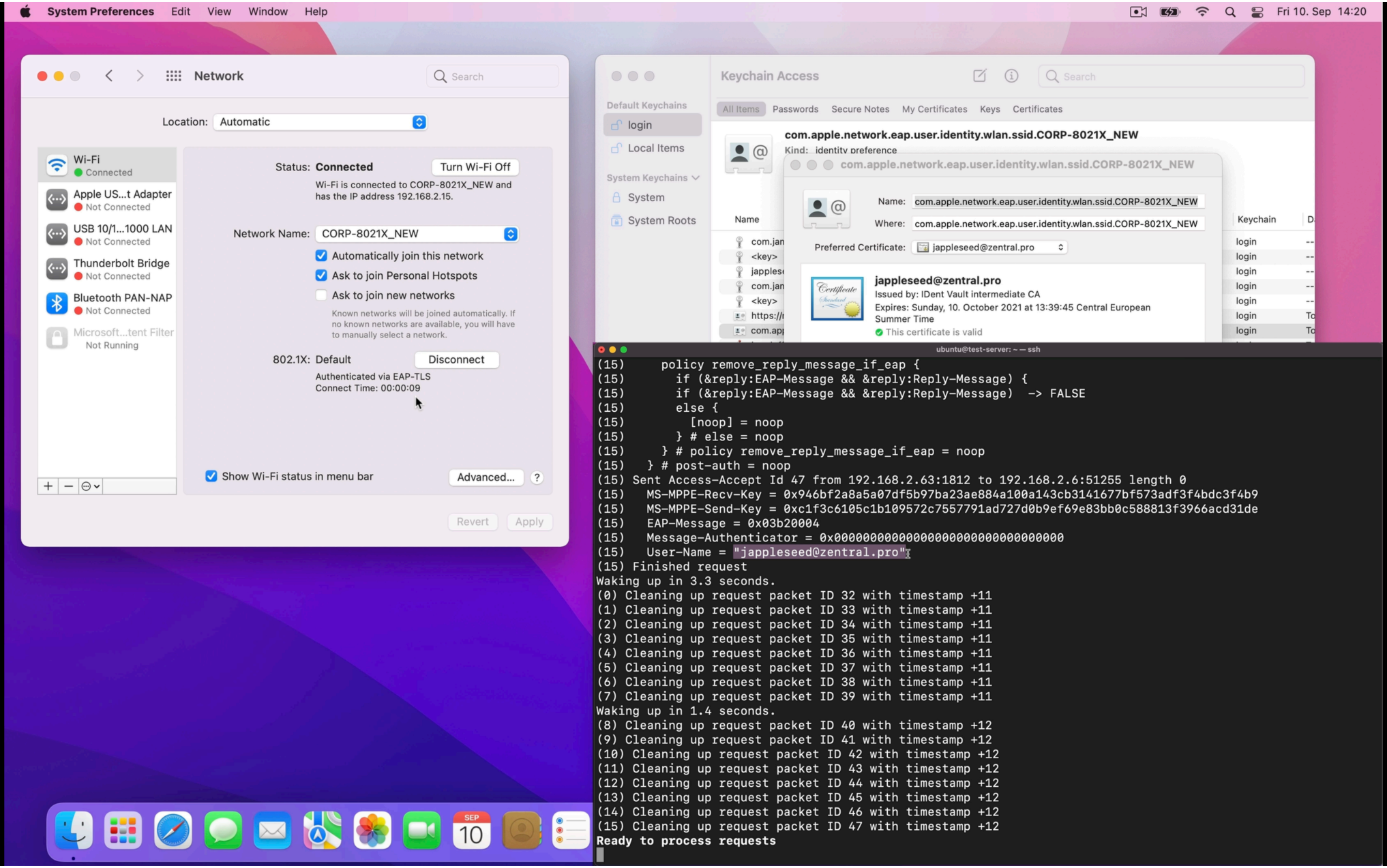
Case study #04

Santa access with client certificate


System Preferences Edit View Window Help
Fri 10. Sep 16:04
Keychain Access
Search
Default Keychains
login
Local Items
System Keychains
System
System Roots
All Items Passwords Secure Notes My Certificates Keys Certificates
MSA-DEVICE-CERT
Issued by: IDent Vault intermediate CA
Expires: Sunday, 10. October 2021 at 15:29:07 Central European Summer Time
This certificate is valid
Name Kind Expires
MSA-DEVICE-CERT certificate 10. Oct 2021 at 15:29:0
MSA-DEVICE-CERT private key --
00F27B79-766E-472B-BC6A-CA17C881C915 certificate 31. Aug 2023 at 12:06:
Santa
Open Santa
Santa
Version 2021.5 (2021.5)
com.google.santa
Apple Silicon—64-bit
Intel—64-bit
Copyright Google LLC.
4,9 MB
Last modified 1. Sep 2021 at 14:35:25
App Sandbox Not enabled
Hardening Enabled — Version 11.1.0
Notarization Granted
Gatekeeper Notarized Developer ID
Signed By Google, Inc. (EQHXZ8M8AV)
Open With Apparency
Profiles
Search
User (Managed)
User Certificate - 80...
2 settings
Device (Managed)
Device Cert (MSA de...
1 setting
Identity
1 setting
MDM Profile
3 settings
MSA Ident
1 setting
Privacy Preferences...
1 setting
Santa 2021.5 MSA
4 settings
TrustCerts (MSA dem...
Device Cert (MSA demo)
Zentral Pro Services GmbH & Co. KG Verified
Signed JSS Built-In Signing Certificate
Installed 10. Sep 2021 at 15:29
Settings SCEP Enrolment
MSA-DEVICE-CERT
DETAILS
SCEP Enrolment
Description SCEP (CA-IDENT)
Server https://msa-id.macadmin.me/gw/scep/
Certificate MSA-DEVICE-CERT
Expires 10. Oct 2021 at 15:29
Issuer IDent Vault intermediate CA
This Mac is supervised and managed by Zentral Pro Services GmbH & Co. KG.
23:52
41:10

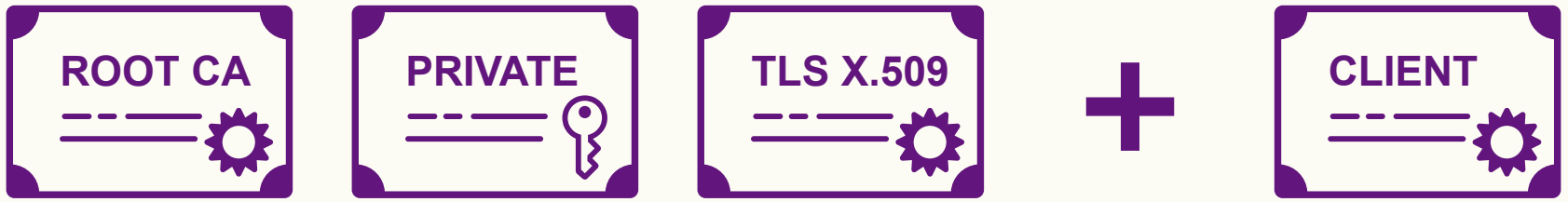
Case study #5

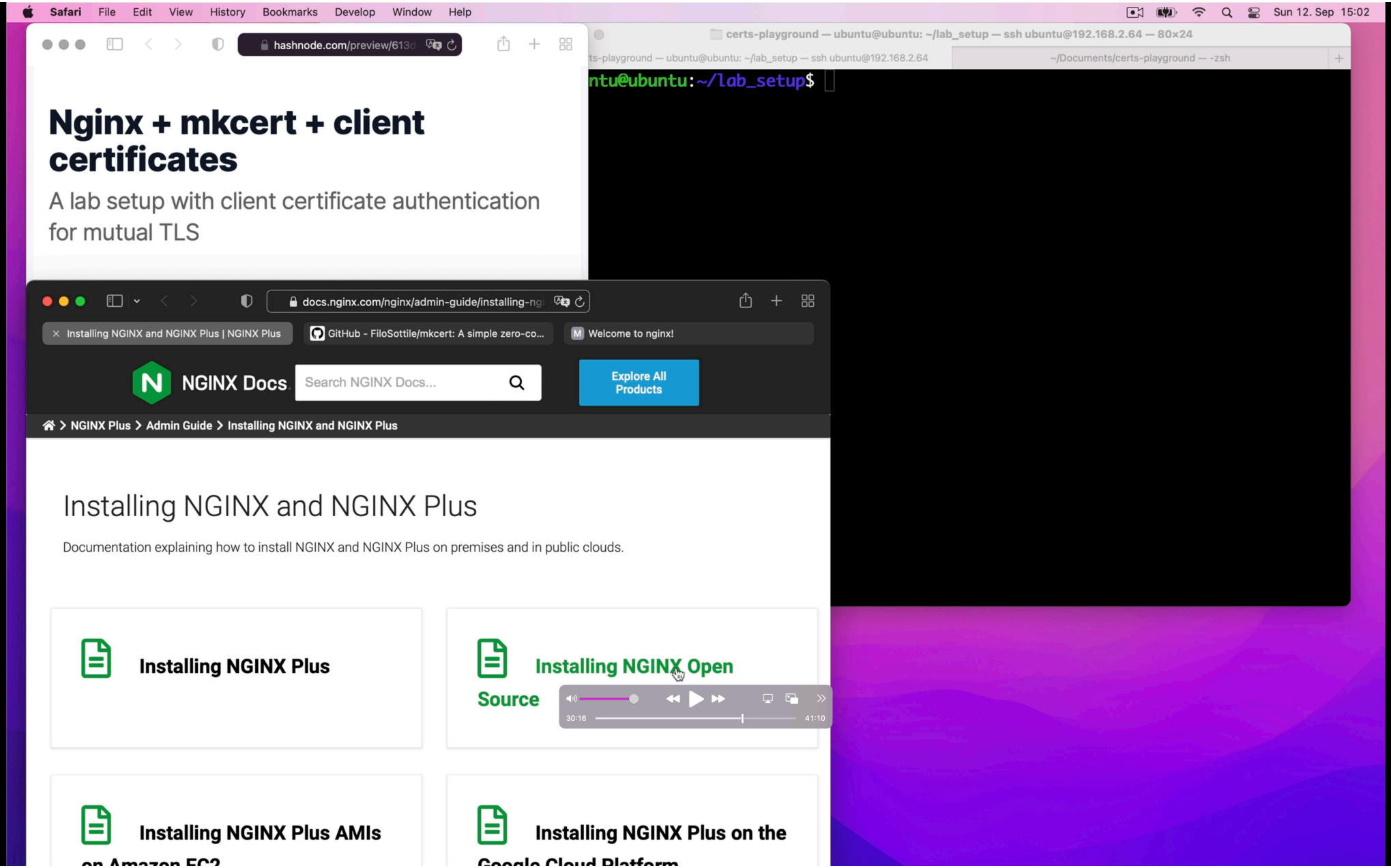
802.1X EAP-TLS setup
Certificate-based authentication to WiFi networks


System Preferences Edit View Window Help
Fri 10. Sep 14:20
Network
Search
Location: Automatic
Wi-Fi
Connected
Apple US...t Adapter
Not Connected
USB 10/1...1000 LAN
Not Connected
Thunderbolt Bridge
Not Connected
Bluetooth PAN-NAP
Not Connected
Microsoft...tent Filter
Not Running
Status: Connected
Turn Wi-Fi Off
Wi-Fi is connected to CORP-8021X_NEW and has the IP address 192.168.2.15.
Network Name: CORP-8021X_NEW
Automatically join this network
Ask to join Personal Hotspots
Ask to join new networks
Known networks will be joined automatically. If no known networks are available, you will have to manually select a network.
802.1X: Default
Disconnect
Authenticated via EAP-TLS
Connect Time: 00:00:09
Show Wi-Fi status in menu bar
Advanced...
Revert
Apply
Keychain Access
Search
Default Keychains
login
Local Items
System Keychains
System
System Roots
All Items
Passwords
Secure Notes
My Certificates
Keys
Certificates
com.apple.network.eap.user.identity.wlan.ssid.CORP-8021X_NEW
Kind: identity preference
com.apple.network.eap.user.identity.wlan.ssid.CORP-8021X_NEW
Name: com.apple.network.eap.user.identity.wlan.ssid.CORP-8021X_NEW
Where: com.apple.network.eap.user.identity.wlan.ssid.CORP-8021X_NEW
Preferred Certificate: jappleseed@zentral.pro
jappleseed@zentral.pro
Issued by: IDent Vault intermediate CA
Expires: Sunday, 10. October 2021 at 13:39:45 Central European Summer Time
This certificate is valid
Name
Keychain
ubuntu@test-server: ~ — ssh
(15) policy remove_reply_message_if_eap {
(15) if (&reply:EAP-Message && &reply:Reply-Message) {
(15) if (&reply:EAP-Message && &reply:Reply-Message) -> FALSE
(15) else {
(15) [noop] = noop
(15) } # else = noop
(15) } # policy remove_reply_message_if_eap = noop
(15) } # post-auth = noop
(15) Sent Access-Accept Id 47 from 192.168.2.63:1812 to 192.168.2.6:51255 length 0
(15) MS-MPPE-Recv-Key = 0x946bf2a8a5a07df5b97ba23ae884a100a143cb3141677bf573adf3f4bdc3f4b9
(15) MS-MPPE-Send-Key = 0xc1f3c6105c1b109572c7557791ad727d0b9ef69e83bb0c588813f3966acd31de
(15) EAP-Message = 0x03b20004
(15) Message-Authenticator = 0x00000000000000000000000000000000
(15) User-Name = "jappleseed@zentral.pro"
(15) Finished request
Waking up in 3.3 seconds.
(0) Cleaning up request packet ID 32 with timestamp +11
(1) Cleaning up request packet ID 33 with timestamp +11
(2) Cleaning up request packet ID 34 with timestamp +11
(3) Cleaning up request packet ID 35 with timestamp +11
(4) Cleaning up request packet ID 36 with timestamp +11
(5) Cleaning up request packet ID 37 with timestamp +11
(6) Cleaning up request packet ID 38 with timestamp +11
(7) Cleaning up request packet ID 39 with timestamp +11
Waking up in 1.4 seconds.
(8) Cleaning up request packet ID 40 with timestamp +12
(9) Cleaning up request packet ID 41 with timestamp +12
(10) Cleaning up request packet ID 42 with timestamp +12
(11) Cleaning up request packet ID 43 with timestamp +12
(12) Cleaning up request packet ID 44 with timestamp +12
(13) Cleaning up request packet ID 45 with timestamp +12
(14) Cleaning up request packet ID 46 with timestamp +12
(15) Cleaning up request packet ID 47 with timestamp +12
Ready to process requests

Practical Lab

Development CA for evaluation




Safari File Edit View History Bookmarks Develop Window Help
Sun 12. Sep 15:02
hashnode.com/preview/613d
Nginx + mkcert + client certificates
A lab setup with client certificate authentication for mutual TLS
certs-playground — ubuntu@ubuntu: ~/lab_setup — ssh ubuntu@192.168.2.64 — 80×24
ts-playground — ubuntu@ubuntu: ~/lab_setup — ssh ubuntu@192.168.2.64
~/Documents/certs-playground — -zsh
ntu@ubuntu:~/lab_setup$
docs.nginx.com/nginx/admin-guide/installing-ngi
Installing NGINX and NGINX Plus | NGINX Plus
GitHub - FiloSottile/mkcert: A simple zero-co...
Welcome to nginx!
NGINX Docs
Search NGINX Docs...
Explore All Products
NGINX Plus > Admin Guide > Installing NGINX and NGINX Plus
Installing NGINX and NGINX Plus
Documentation explaining how to install NGINX and NGINX Plus on premises and in public clouds.
Installing NGINX Plus
Installing NGINX Open Source
30:16
41:10
Installing NGINX Plus AMIs
Installing NGINX Plus on the

Provisioning

Certificate distribution

File system - most simple

PKCS12 (.p12) - option to embedded in MDM payload data

ADCS Connectors - hand over copy of the key material

SCEP - de facto standard for certificate provisioning

SCEP proxied - shields SCEP server on the Internet

EST - Enrollment over Secure Transport protocol

Conclusion

Why certificates based approach?

Identity - core component of security

Reliability - effective when used

Dynamic - use certain attributes and features

Secure - hard to impersonate

Future - application will increase

zentral
Development & Professional Services

Twitter: @head_min
Slack: @headmin

Zentral Pro Services
https://zentral.pro

Thank you

Twitter: @head_min
Slack: @headmin

Zentral Pro Services
https://zentral.pro