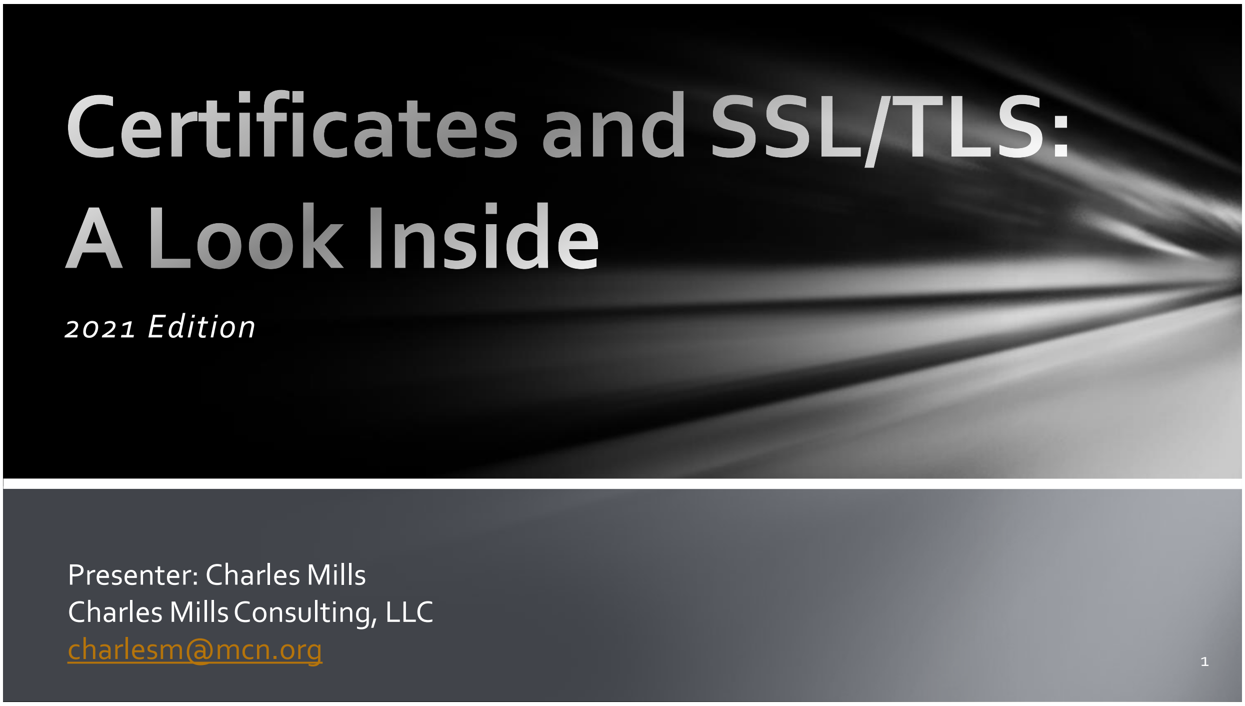
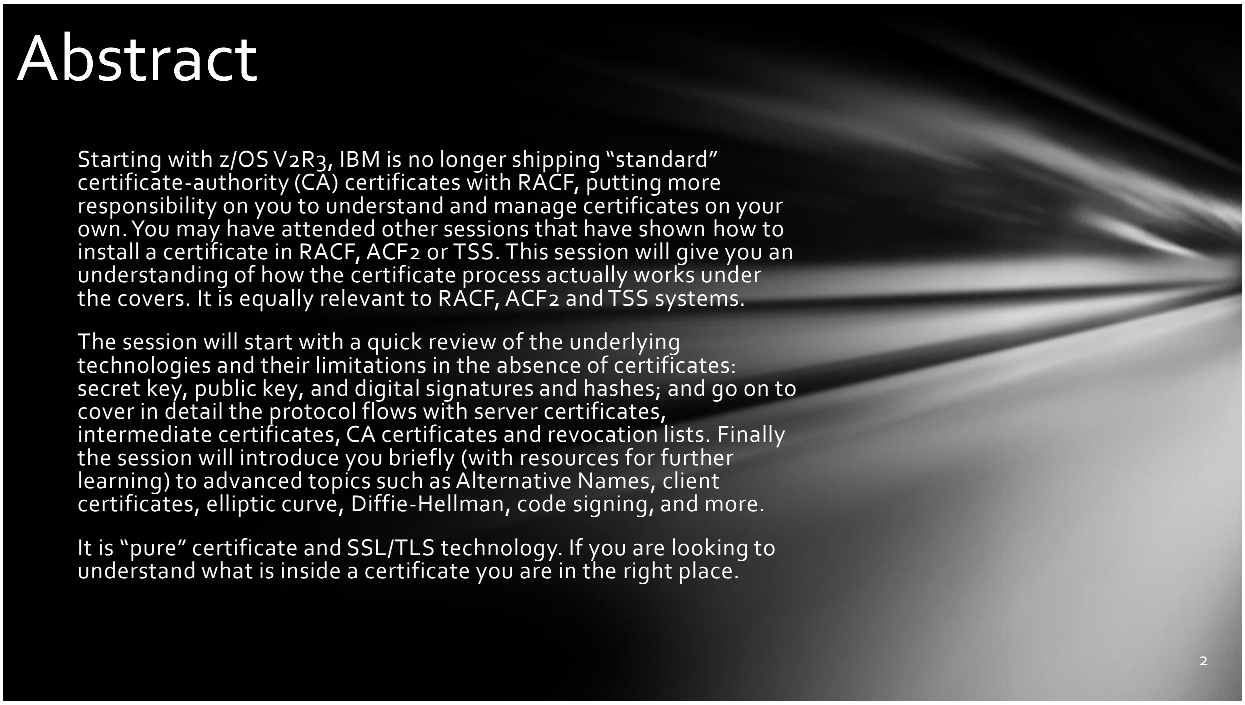


Certificates and SSL/TLS: A Look Inside

2021 Edition

Presenter: Charles Mills
Charles Mills Consulting, LLC
charlesm@mcn.org



Abstract

Starting with z/OS V2R3, IBM is no longer shipping "standard" certificate-authority (CA) certificates with RACF, putting more responsibility on you to understand and manage certificates on your own. You may have attended other sessions that have shown how to install a certificate in RACF, ACF2 or TSS. This session will give you an understanding of how the certificate process actually works under the covers. It is equally relevant to RACF, ACF2 and TSS systems.

The session will start with a quick review of the underlying technologies and their limitations in the absence of certificates: secret key, public key, and digital signatures and hashes; and go on to cover in detail the protocol flows with server certificates, intermediate certificates, CA certificates and revocation lists. Finally the session will introduce you briefly (with resources for further learning) to advanced topics such as Alternative Names, client certificates, elliptic curve, Diffie-Hellman, code signing, and more.

It is "pure" certificate and SSL/TLS technology. If you are looking to understand what is inside a certificate you are in the right place.

About the Presenter

Charles has been writing mainframe software for longer than he cares to admit. He developed security software for eight years at CorreLog, where he authored the zDefender and SyslogDefender products which were acquired by BMC. He is currently the Chief Development Officer for Cloud Compiling and also does freelance project-oriented development.

He has a PhD in certificate technology from the University of Hard Knocks.

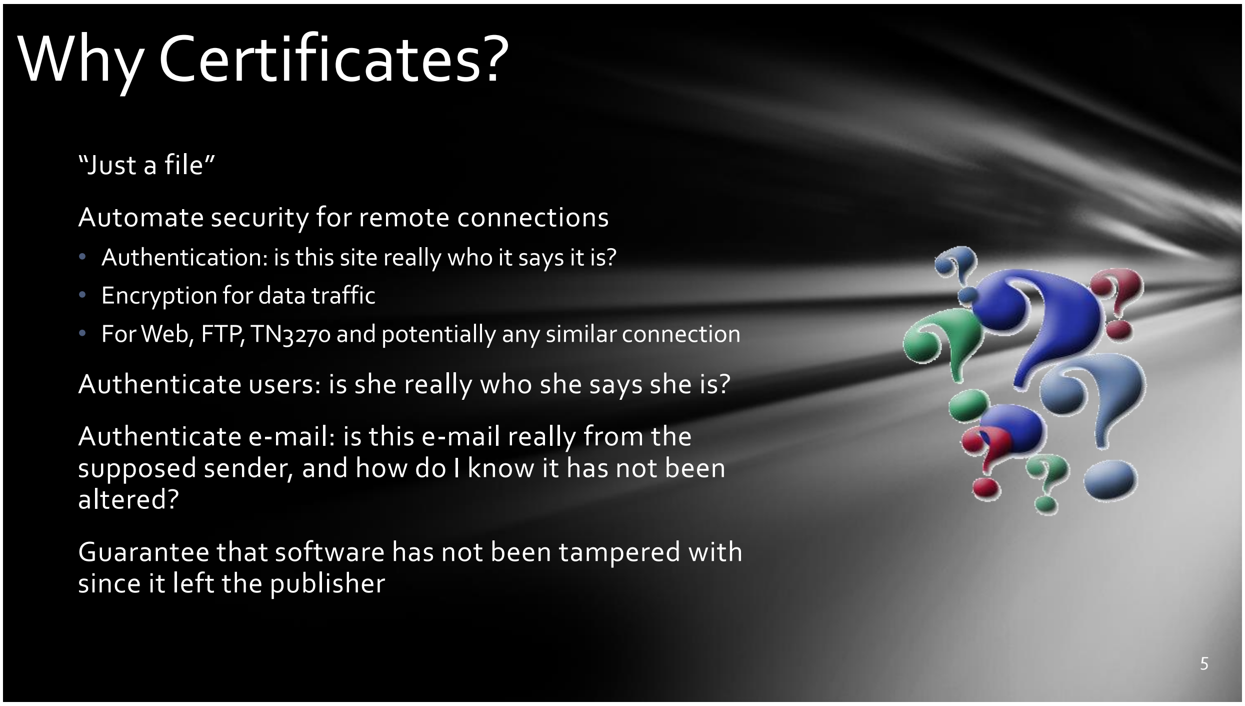
The University of Certificate Hard Knocks

Windows product

- My introduction to the nitty-gritty of certificates
- Implements both client and server ends of TLS protocol
- Built using OpenSSL
  - Open Source – "lightly" documented – forces one to learn as one goes
  - Result can be "the most dangerous code in the world" https://bit.ly/2Djr76W

z/OS product

- Built using IBM z/OS System SSL ("GSK")
- Designed to force you not to make mistakes
- Highly recommended



Why Certificates?

"Just a file"

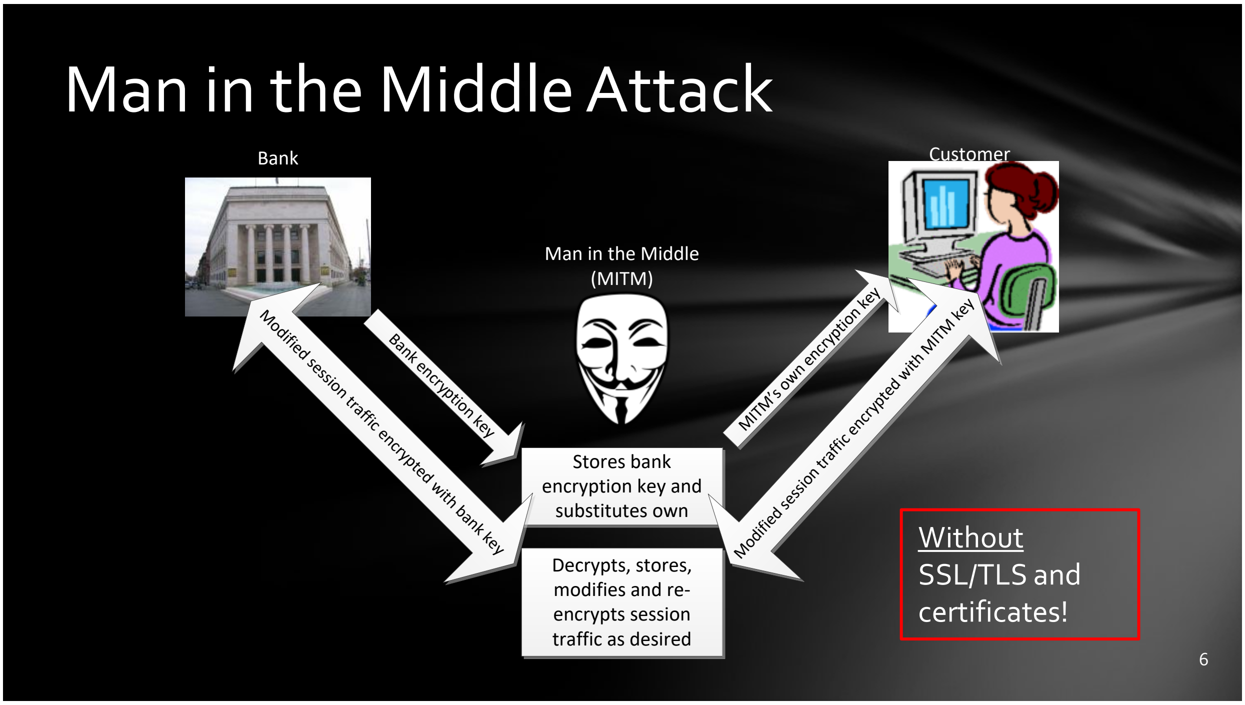
Automate security for remote connections

- Authentication: is this site really who it says it is?
- Encryption for data traffic
- For Web, FTP, TN3270 and potentially any similar connection

Authenticate users: is she really who she says she is?

Authenticate e-mail: is this e-mail really from the supposed sender, and how do I know it has not been altered?

Guarantee that software has not been tampered with since it left the publisher



Man in the Middle Attack

Bank

Man in the Middle (MITM)

Customer

Bank encryption key

Stores bank encryption key and substitutes own

MITM's own encryption key

Modified session traffic encrypted with bank key

Decrypts, stores, modifies and re-encrypts session traffic as desired

Modified session traffic encrypted with MITM key

Without SSL/TLS and certificates!

Brief history of SSL and TLS ("SSL/TLS")

1994: Netscape, first company to commercialize the Internet, perceives browser communication not secure enough for e-commerce

1994: Develops Secure Sockets Layer (SSL) version 1 (never released)

1995 and 1996: SSL versions 2 and 3 (both now deprecated)

1998: Netscape crashes and burns in Microsoft browser wars

1999: SSL v3 becomes IETF Transport Layer Security (TLS) v1.0

TLS now at Versions 1.2 and 1.3. TLS 1.3 was defined in RFC 8446 in August 2018.

Certificates are a fundamental component of SSL/TLS

Netscape

http://bit.ly/2DCRGiY
http://bit.ly/2DkGmus

Agenda

100 MPH review of underlying technologies

- With links for additional reference

Details of the certificate protocol flow

100 MPH introduction to some advanced features

- With links for additional reference

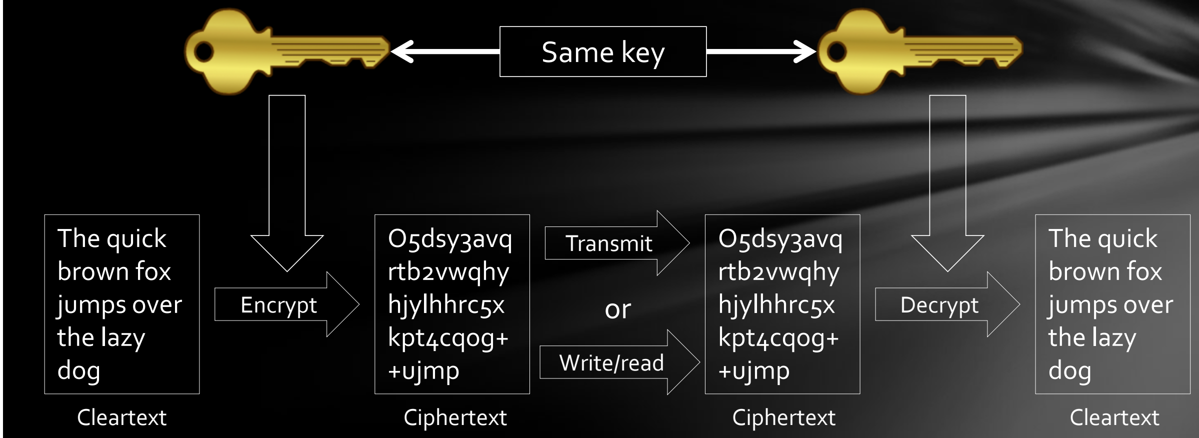
AlphaCoders.com

Reference links for more information

100 MPH Review of Underlying Technologies

9

Secret (Symmetric) Key Encryption



Same key

The quick brown fox jumps over the lazy dog

Cleartext

Encrypt

O5dsy3avq rtb2vwqhy hjylhhrc5x kpt4cqog+ +ujmp

Ciphertext

Transmit

or

Write/read

O5dsy3avq rtb2vwqhy hjylhhrc5x kpt4cqog+ +ujmp

Ciphertext

Decrypt

The quick brown fox jumps over the lazy dog

Cleartext

http://bit.ly/2ilor54

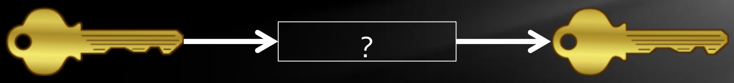
10

Secret Key Encryption

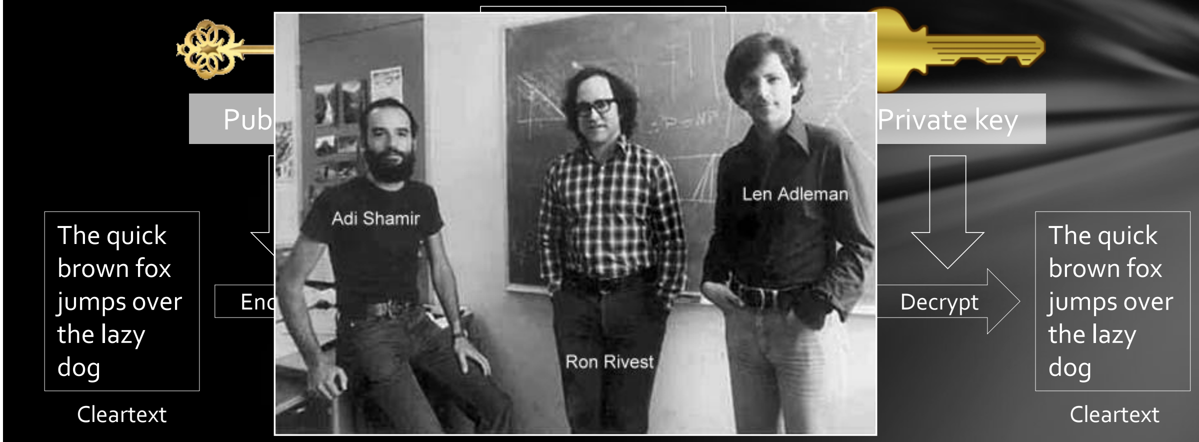
What's the big problem?

Key Management

- How do we get that secret key from one end to the other?
- How do we keep track of thousands of individual secret keys?
  - Chase Bank has 4 million customers



Public Key (Asymmetric) Encryption



The quick brown fox jumps over the lazy dog

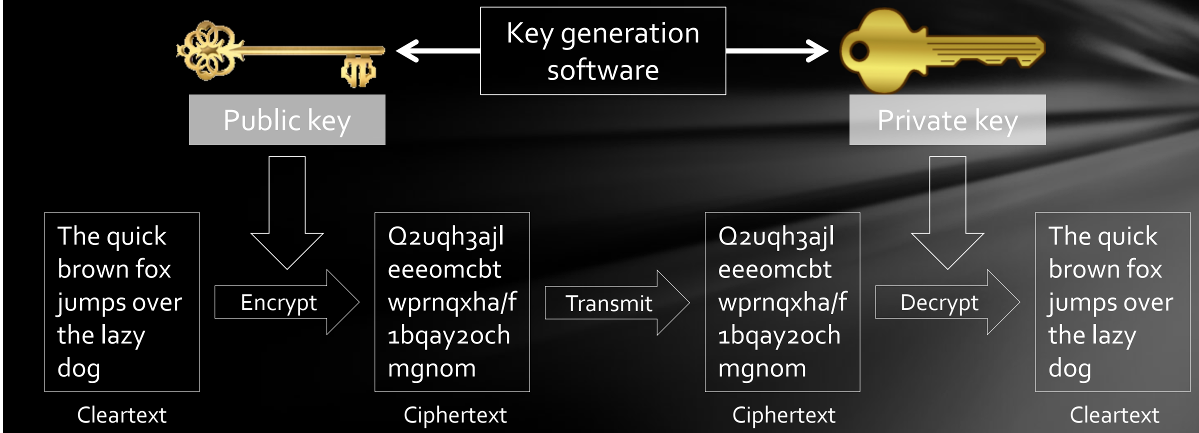
Cleartext

The quick brown fox jumps over the lazy dog

Cleartext

http://bit.ly/2io5WvQ

Public Key (Asymmetric) Encryption


Key generation software
Public key
Private key
The quick brown fox jumps over the lazy dog
Encrypt
Q2uqh3ajl eeeomcbt wprnqxha/f 1bqay2och mgnom
Transmit
Q2uqh3ajl eeeomcbt wprnqxha/f 1bqay2och mgnom
Decrypt
The quick brown fox jumps over the lazy dog
Cleartext
Ciphertext
Ciphertext
Cleartext

http://bit.ly/2io5WvQ

My Public Key

```
Public-Key: (2048 bit)
Modulus:
00:ad:3d:3a:cf:fd:39:8f:b0:d9:6d:8e:27:ad:37:
c7:74:a2:b3:7a:05:b0:de:f9:06:96:f7:c6:a1:16:
d5:2b:39:28:30:d2:63:3c:96:f5:3e:d1:9f:9b:9a:
1f:3e:29:71:be:7d:6b:c3:a3:90:de:ce:41:b0:e8:
5d:fe:ce:05:0d:d5:55:7f:fa:58:df:3b:5b:25:98:
8e:cb:c2:d1:6e:0d:be:44:88:87:9f:b1:a0:cf:de:
ae:7d:e3:fd:d1:81:64:2b:48:f1:7a:83:d7:e9:66:
9f:32:3a:9a:26:d5:41:50:3e:8a:a4:9c:18:9a:c1:
21:ea:9b:b5:23:b1:57:27:55:e0:85:a0:d6:0e:c4:
3b:ea:8e:03:b7:4e:28:e0:c8:57:de:db:fe:a4:dc:
32:11:09:aa:d8:6d:04:e0:f6:d5:e2:08:c4:87:30:
29:3a:bd:0f:2b:45:7d:b8:6e:8c:71:22:ff:8c:3c:
68:7d:64:87:f7:87:a5:66:2c:d2:71:e2:97:84:48:
26:82:58:e4:0b:d6:59:e3:57:0a:07:24:77:e3:39:
3a:07:04:f6:ac:23:e1:33:28:ba:f3:5b:7c:df:91:
27:a4:79:a1:e5:6c:e9:c7:23:74:81:a7:cc:7f:75:
c4:9e:d4:7e:27:af:23:9f:32:87:2e:f1:87:e7:38:
0f:31
Exponent: 65537 (0x10001)
```

But absolutely imperative to keep that private key private!

Public Key Encryption

What are the big problems?

Keys are HUGE! As you saw on the previous slide

- 4096 bits is over 1200 decimal digits
- Essence of public key is the difficulty of factoring huge numbers

Need a program to generate a pair of keys

Key management (are you noticing a theme?)

Unidirectional

Very slow

Don't say Mills said Public Key was no good – we will see how certificates solve all of these problems

16

Digital Hash

Function that takes a possibly very long "message" and returns a relatively short fixed-length binary value

Must be relatively easy to calculate

Deterministic: same message yields same hash

Possible but highly unlikely two slightly different messages have same hash

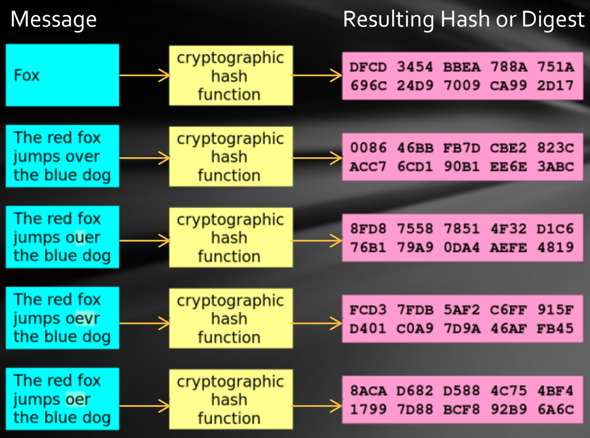
Almost impossible to construct a message with a predictable hash

Same hash = same message

Also called message digest

Examples: SHA-2 (MD5, SHA-1)

http://bit.ly/2mNglor

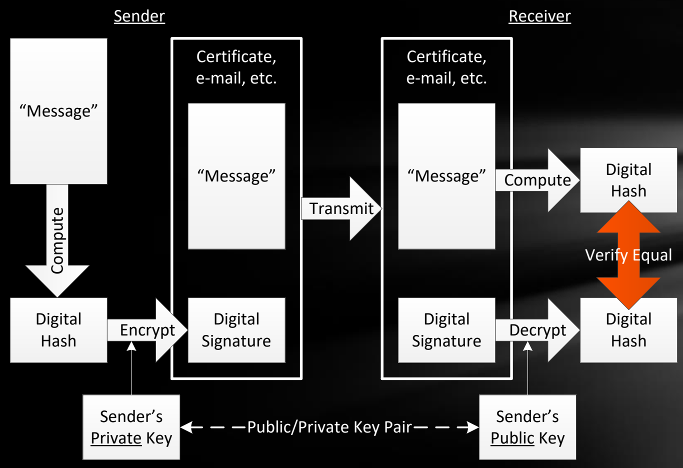


Message
Resulting Hash or Digest
Fox
cryptographic hash function
DFCD 3454 BBEA 788A 751A 696C 24D9 7009 CA99 2D17
The red fox jumps over the blue dog
cryptographic hash function
0086 46BB FB7D CBE2 823C ACC7 6CD1 90B1 EE6E 3ABC
The red fox jumps ouer the blue dog
cryptographic hash function
8FD8 7558 7851 4F32 D1C6 76B1 79A9 0DA4 AEFE 4819
The red fox jumps oevr the blue dog
cryptographic hash function
FCD3 7FDB 5AF2 C6FF 915F D401 C0A9 7D9A 46AF FB45
The red fox jumps oer the blue dog
cryptographic hash function
8ACA D682 D588 4C75 4BF4 1799 7D88 BCF8 92B9 6A6C

Source: Wikipedia
By User:Jorge Stolfi based on Image:Hash_function.svg by Helix84 - Original work for Wikipedia,
Public Domain, https://commons.wikimedia.org/w/index.php?curid=5290240

18

Digital Signatures



Public key in reverse: encrypt hash with private key, decrypt with public key

Verifies message not tampered with: any change to message would change the hash

Authenticates: only owner of private key could have signed (encrypted)

Non-repudiation: only owner of private key could have sent

http://bit.ly/2rTHa54

Digital Signatures and Trust

You receive a "message" (could be anything) purportedly from me with an attached digital hash encrypted with some private key. You also compute your own digital hash for the message.

If you can decrypt the attached digital hash with my public key (well known) and it is the same as the digital hash you computed, then the message is from me and is unaltered. If you trust me then you can trust the message.

By extension, if the message contains a public key, then you can trust any message signed with that public/private key pair.

This – the "chain of trust" – is the essence of certificates.

B Franklin

Certificate Authority

Company or group within a company that issues certificates

Uses self-created "root" certificate to sign them

May be well-known CA

- IdenTrust (owned by 8 large banks)
- DigiCert (20% market share)
- ~~Comodo~~ Sectigo (17% market share)
- Symantec (acquired Verisign*; CA business sold to DigiCert)
- GoDaddy
- GlobalSign
- Entrust*
- Let's Encrypt – Free! Highly automated. Transparent (public log). Ninety days only!

Or department or individual

Well-known CA required for public-use SSL/TLS

http://bit.ly/2oKougM

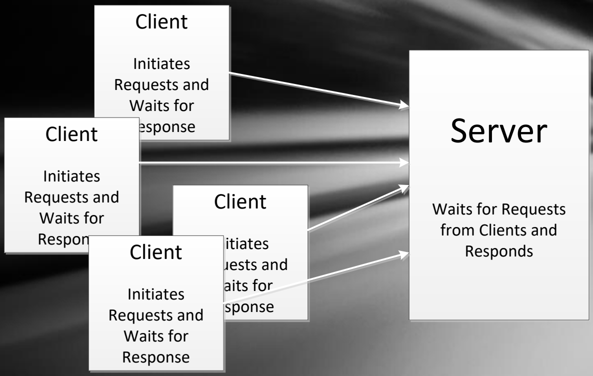
*CA Certificates that formerly shipped with RACF. IBM also shipped Thawte, acquired by Verisign (acquired by Symantec, acquired by DigiCert).

Client and Server

Nothing to do with color or size of boxes

Often software, not hardware

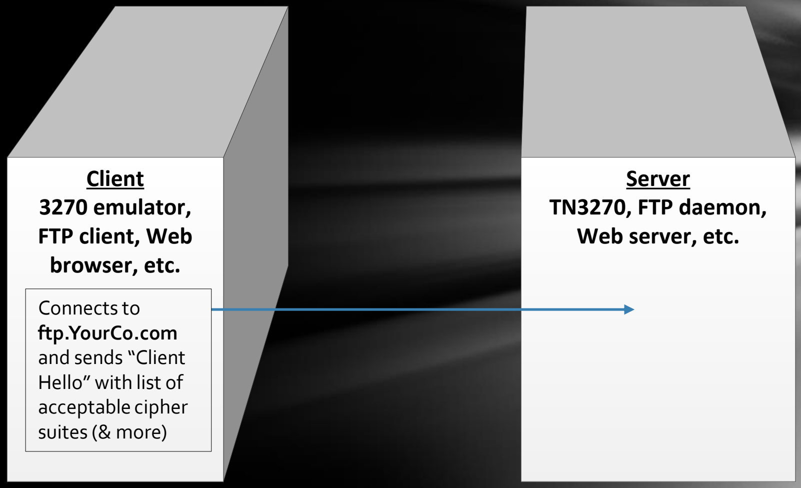
The predominant architecture for complex applications (Web browser, FTP, e-mail, 3270 emulation)


Client
Initiates Requests and Waits for Response
Client
Initiates Requests and Waits for Respon
Client
itiates uests and aits for sponse
Client
Initiates Requests and Waits for Response
Server
Waits for Requests from Clients and Responds

http://bit.ly/2DjfNWk

TLS certificate protocol flow

Client initiates connection to server*



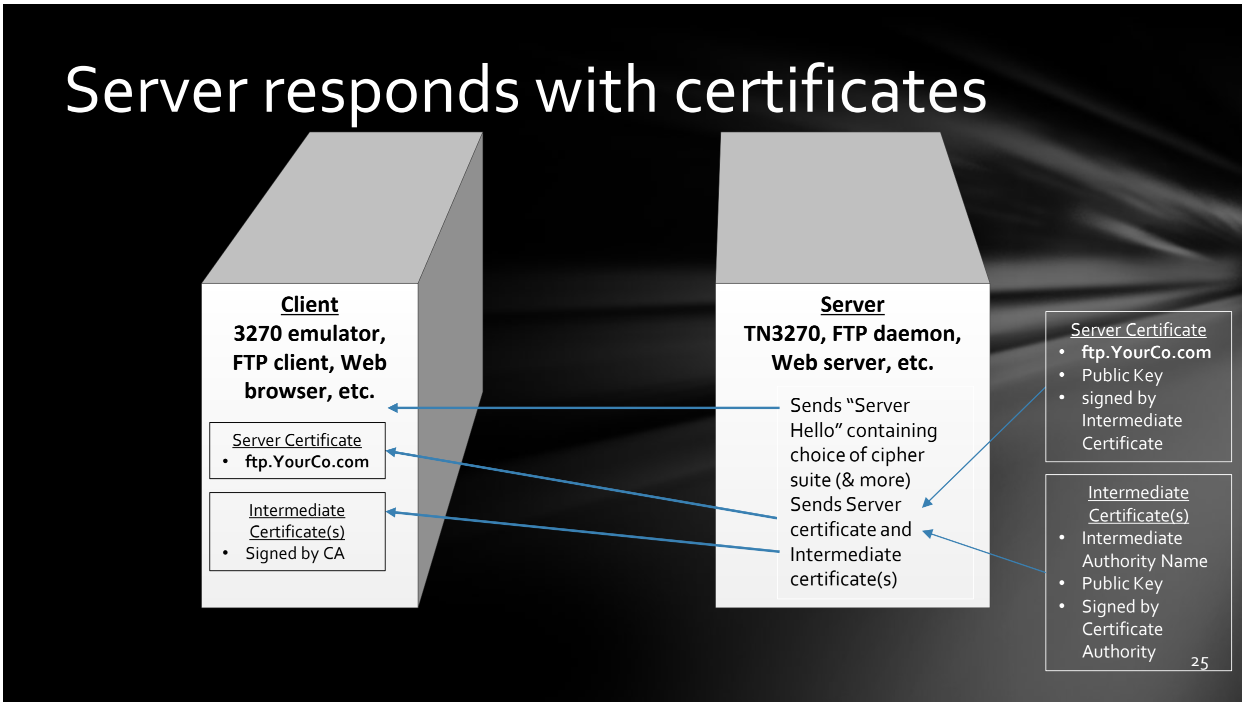
Client
3270 emulator, FTP client, Web browser, etc.

Connects to **ftp.YourCo.com** and sends "Client Hello" with list of acceptable cipher suites (& more)

Server
TN3270, FTP daemon, Web server, etc.

https://bit.ly/2Ukx7V0

*this is the TLS 1.2 and below handshake. TLS 1.3 more complex but similar.



Server responds with certificates

Client

3270 emulator, FTP client, Web browser, etc.

Server Certificate
- **ftp.YourCo.com**

Intermediate Certificate(s)
- Signed by CA

Server

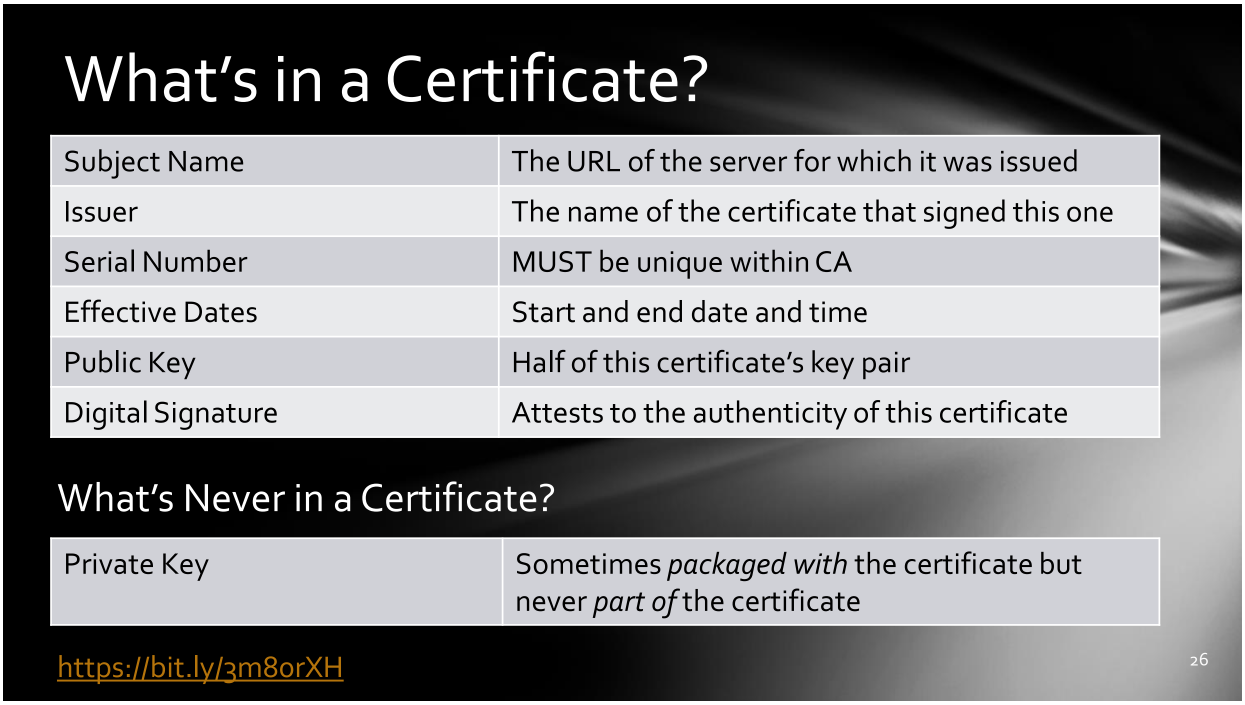
TN3270, FTP daemon, Web server, etc.

Sends "Server Hello" containing choice of cipher suite (& more)
Sends Server certificate and Intermediate certificate(s)

Server Certificate
- **ftp.YourCo.com**
- Public Key
- signed by Intermediate Certificate

Intermediate Certificate(s)
- Intermediate Authority Name
- Public Key
- Signed by Certificate Authority

25



What’s in a Certificate?

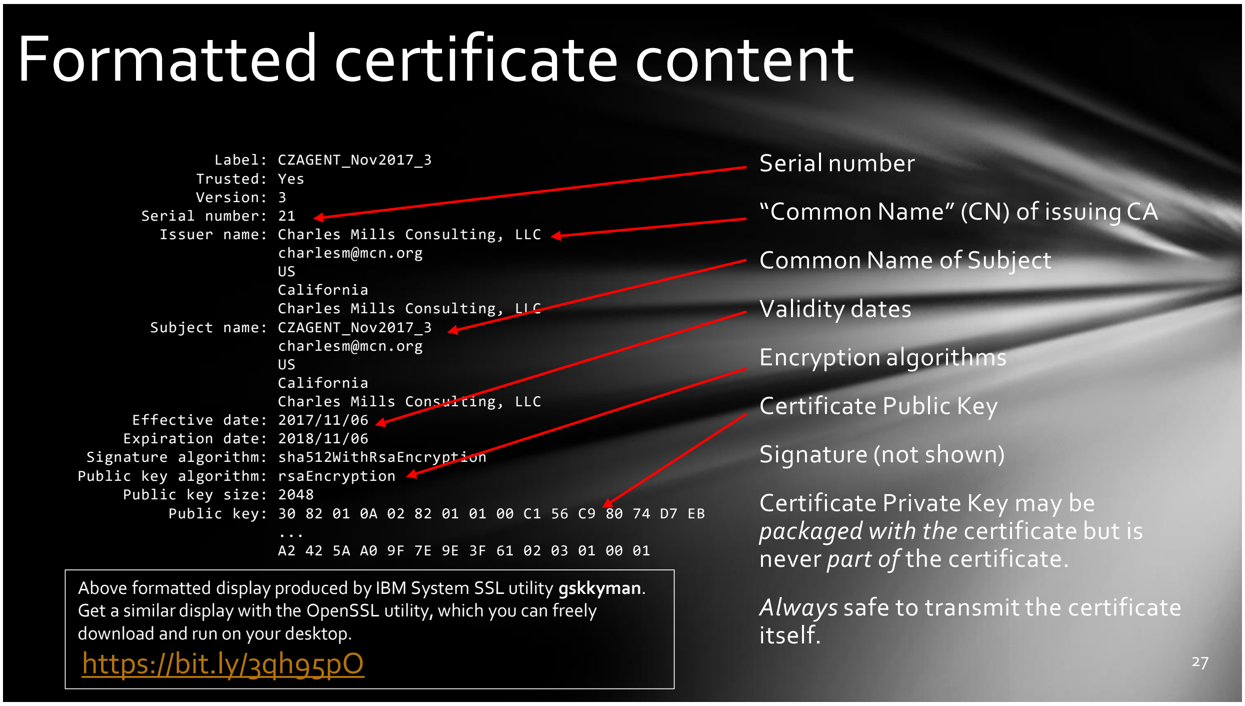
Subject Name	The URL of the server for which it was issued
Issuer	The name of the certificate that signed this one
Serial Number	MUST be unique within CA
Effective Dates	Start and end date and time
Public Key	Half of this certificate’s key pair
Digital Signature	Attests to the authenticity of this certificate

What’s Never in a Certificate?

Private Key	Sometimes *packaged with* the certificate but never *part of* the certificate

https://bit.ly/3m8orXH

26



Formatted certificate content

```
               Label: CZAGENT_Nov2017_3
             Trusted: Yes
             Version: 3
       Serial number: 21
         Issuer name: Charles Mills Consulting, LLC
                      charlesm@mcn.org
                      US
                      California
                      Charles Mills Consulting, LLC
        Subject name: CZAGENT_Nov2017_3
                      charlesm@mcn.org
                      US
                      California
                      Charles Mills Consulting, LLC
      Effective date: 2017/11/06
     Expiration date: 2018/11/06
 Signature algorithm: sha512WithRsaEncryption
Public key algorithm: rsaEncryption
     Public key size: 2048
          Public key: 30 82 01 0A 02 82 01 01 00 C1 56 C9 80 74 D7 EB
                      ...
                      A2 42 5A A0 9F 7E 9E 3F 61 02 03 01 00 01
```

Above formatted display produced by IBM System SSL utility **gskkyman**. Get a similar display with the OpenSSL utility, which you can freely download and run on your desktop.

https://bit.ly/3qh95pO

Serial number

“Common Name” (CN) of issuing CA

Common Name of Subject

Validity dates

Encryption algorithms

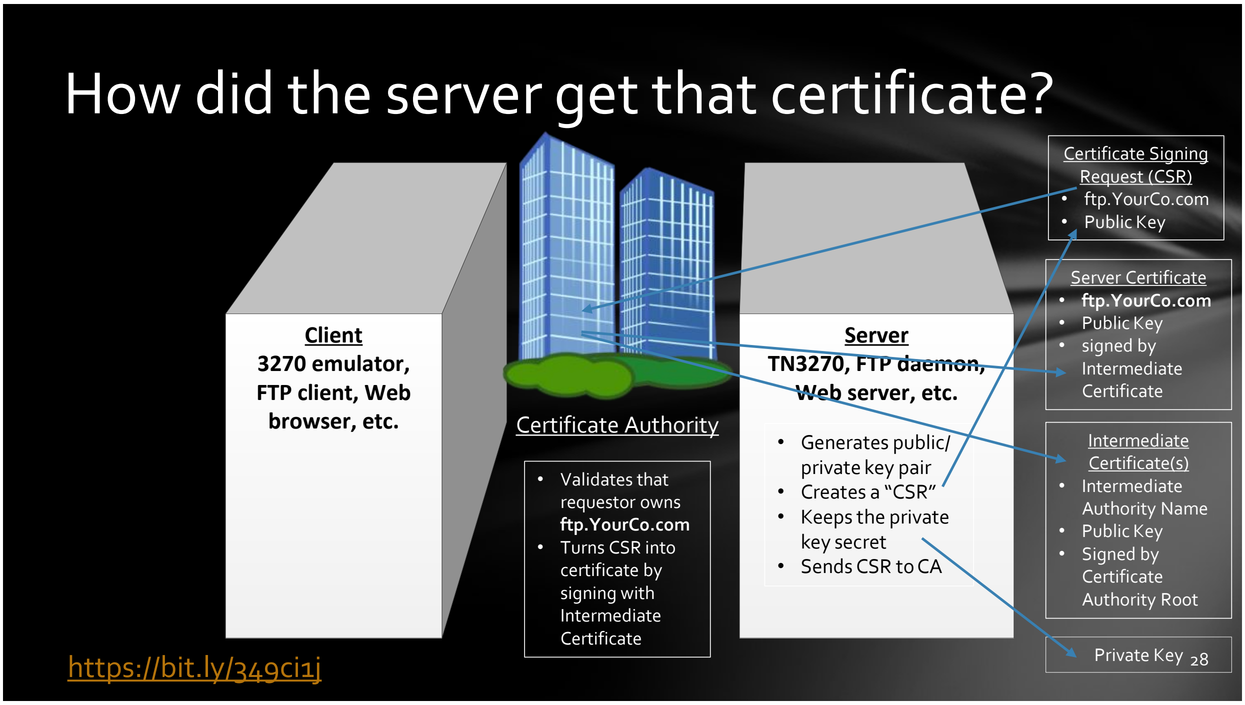
Certificate Public Key

Signature (not shown)

Certificate Private Key may be *packaged with the* certificate but is never *part of* the certificate.

Always safe to transmit the certificate itself.

27



How did the server get that certificate?

Client

3270 emulator, FTP client, Web browser, etc.

Certificate Authority

- Validates that requestor owns **ftp.YourCo.com**
- Turns CSR into certificate by signing with Intermediate Certificate

Server

TN3270, FTP daemon, Web server, etc.

- Generates public/ private key pair
- Creates a “CSR”
- Keeps the private key secret
- Sends CSR to CA

Certificate Signing Request (CSR)

- ftp.YourCo.com
- Public Key

Server Certificate

- **ftp.YourCo.com**
- Public Key
- signed by Intermediate Certificate

Intermediate Certificate(s)

- Intermediate Authority Name
- Public Key
- Signed by Certificate Authority Root

Private Key

https://bit.ly/349ci1j

28

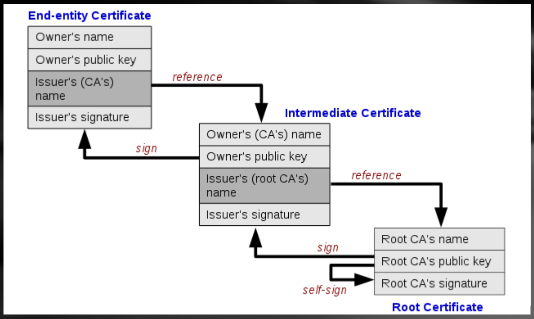
Why Intermediate Certificates?

The compromise of a CA root key would render root and all certificates issued by CA untrustworthy – a disaster!

Certificate Authorities store their root keys off-line to help prevent compromise

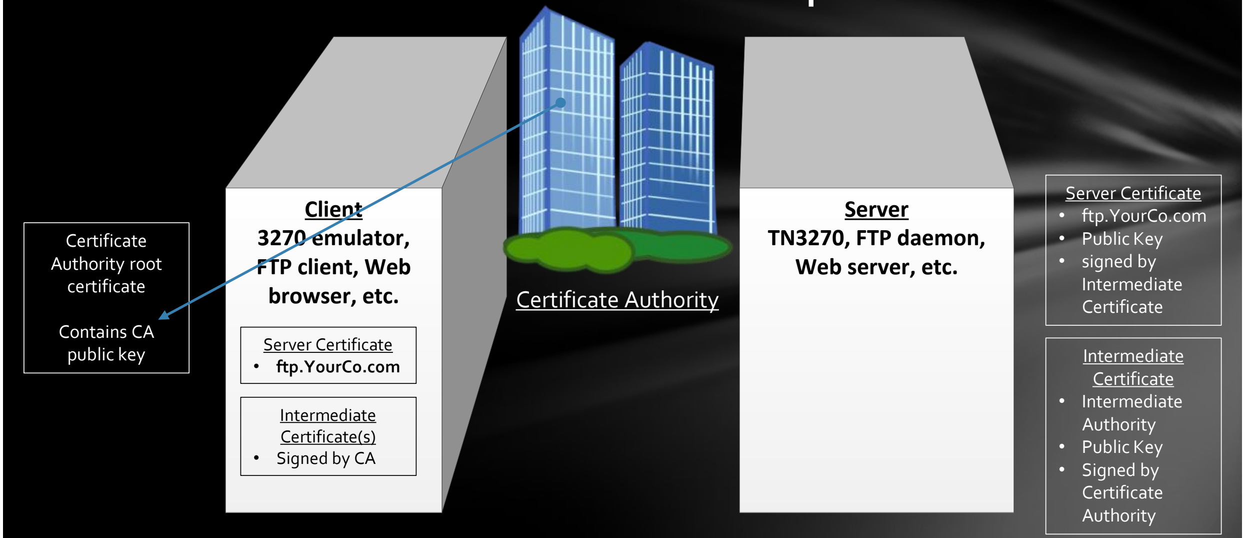
They use medium-term intermediate certificates – signed by their root certificate – to issue end-user certificates

Intermediate certificates are signed by the root certificate: "Chain of trust"



Source: Wikipedia
By Yanpas - Own work, CC BY-SA 4.0,
https://commons.wikimedia.org/w/index.php?curid=46369922

Client has CA root certificate pre-installed



Certificate Authority root certificate

Contains CA public key

Client
3270 emulator, FTP client, Web browser, etc.

Server Certificate
- ftp.YourCo.com

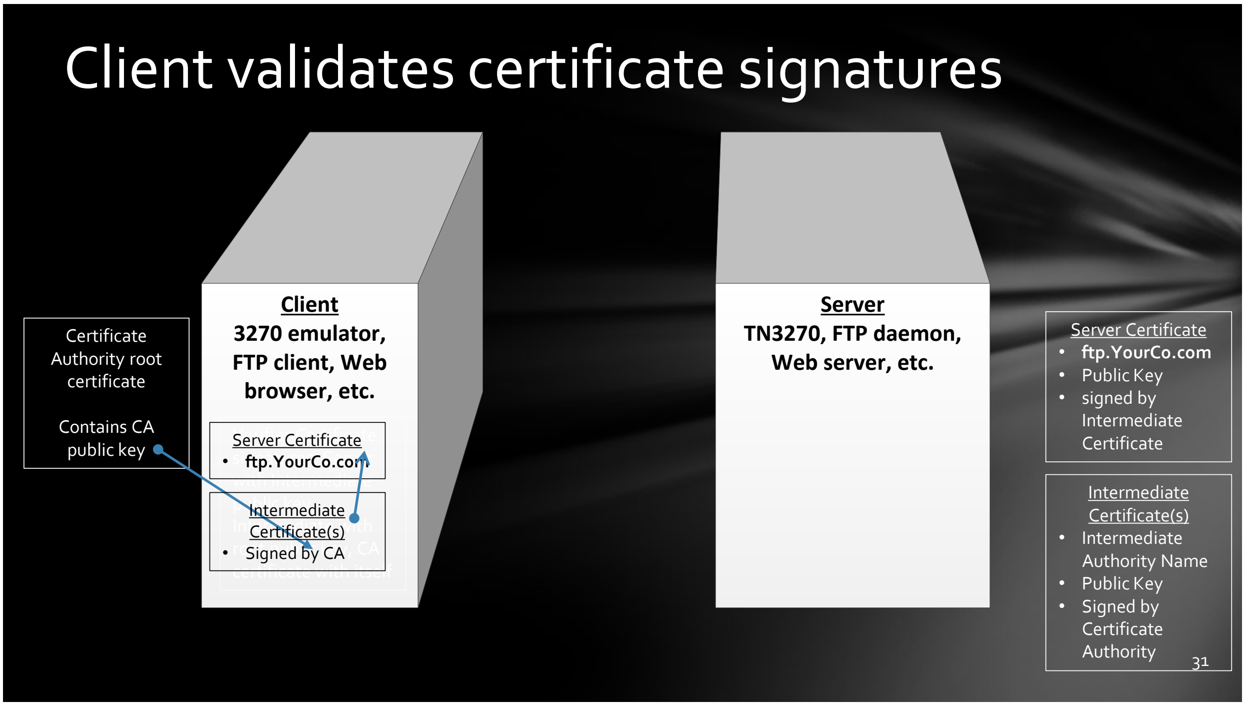
Intermediate Certificate(s)
- Signed by CA

Certificate Authority

Server
TN3270, FTP daemon, Web server, etc.

Server Certificate
- ftp.YourCo.com
- Public Key
- signed by Intermediate Certificate

Intermediate Certificate
- Intermediate Authority
- Public Key
- Signed by Certificate Authority



Client validates certificate signatures

Certificate Authority root certificate

Contains CA public key

Client

3270 emulator, FTP client, Web browser, etc.

Server Certificate
- **ftp.YourCo.com**

Intermediate Certificate(s)
- Signed by CA

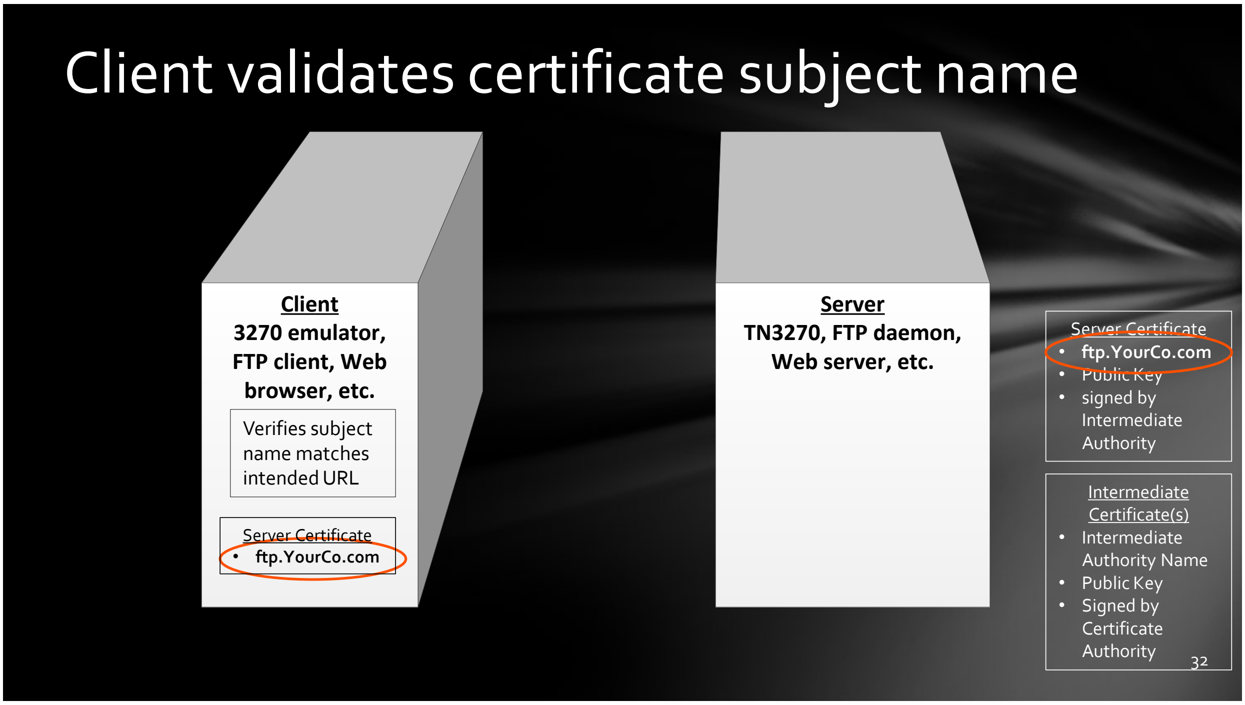
Server

TN3270, FTP daemon, Web server, etc.

Server Certificate
- **ftp.YourCo.com**
- Public Key
- signed by Intermediate Certificate

Intermediate Certificate(s)
- Intermediate Authority Name
- Public Key
- Signed by Certificate Authority

31



Client validates certificate subject name

Client

3270 emulator, FTP client, Web browser, etc.

Verifies subject name matches intended URL

Server Certificate
- **ftp.YourCo.com**

Server

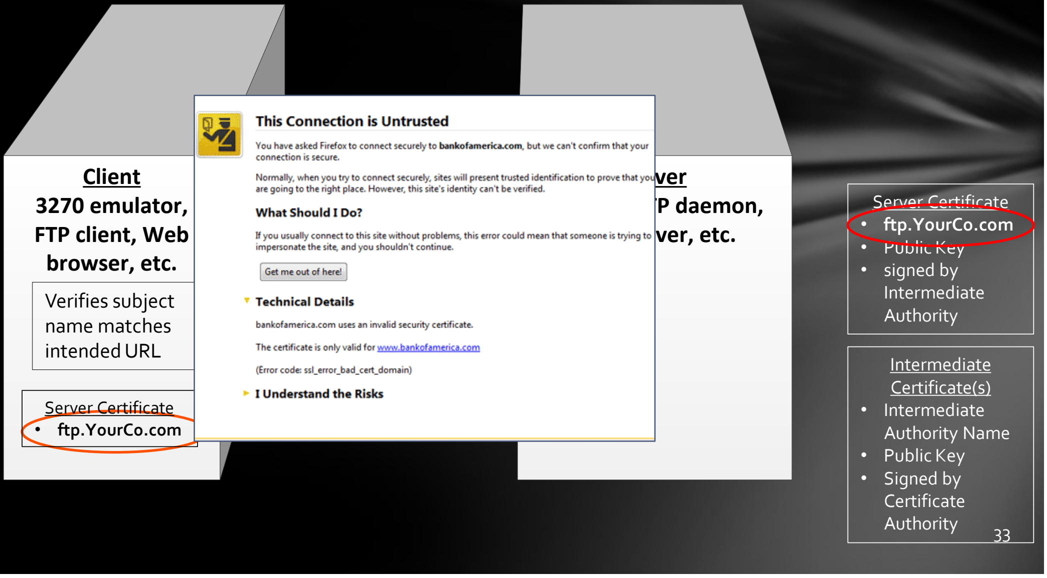
TN3270, FTP daemon, Web server, etc.

Server Certificate
- **ftp.YourCo.com**
- Public Key
- signed by Intermediate Authority

Intermediate Certificate(s)
- Intermediate Authority Name
- Public Key
- Signed by Certificate Authority

32

Client validates certificate subject name



Client
3270 emulator, FTP client, Web browser, etc.

Verifies subject name matches intended URL

Server Certificate
- ftp.YourCo.com

ver
P daemon,
ver, etc.

This Connection is Untrusted

You have asked Firefox to connect securely to **bankofamerica.com**, but we can't confirm that your connection is secure.

Normally, when you try to connect securely, sites will present trusted identification to prove that you are going to the right place. However, this site's identity can't be verified.

What Should I Do?

If you usually connect to this site without problems, this error could mean that someone is trying to impersonate the site, and you shouldn't continue.

Technical Details

bankofamerica.com uses an invalid security certificate.

The certificate is only valid for www.bankofamerica.com

(Error code: ssl_error_bad_cert_domain)

I Understand the Risks

Server Certificate
- ftp.YourCo.com
- Public Key
- signed by Intermediate Authority

Intermediate Certificate(s)
- Intermediate Authority Name
- Public Key
- Signed by Certificate Authority

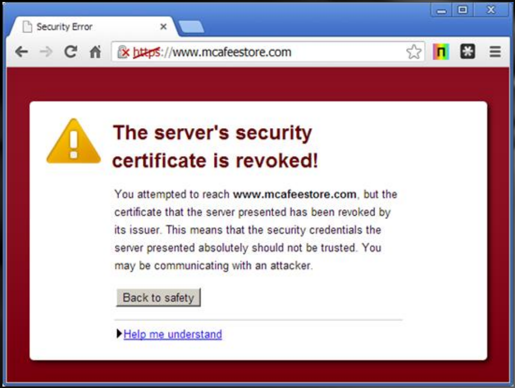
Certificate Revocation

Why would a certificate be revoked?

- Issued in error
- Key compromised
- CA root key compromised (disaster!)

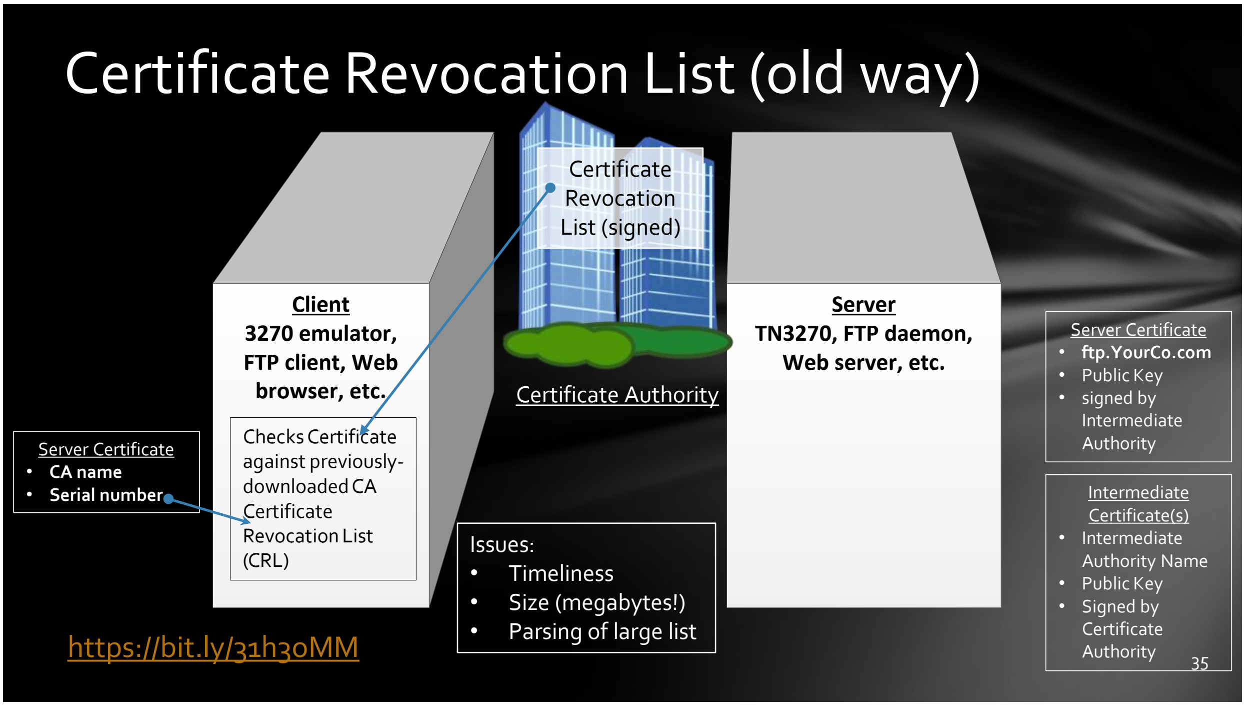
Clients should check for server certificate revocation

- Some clients do not, and some users ignore the error – bad idea!



The server's security certificate is revoked!

You attempted to reach **www.mcafeestore.com**, but the certificate that the server presented has been revoked by its issuer. This means that the security credentials the server presented absolutely should not be trusted. You may be communicating with an attacker.



Certificate Revocation List (old way)

Client
3270 emulator, FTP client, Web browser, etc.

Checks Certificate against previously-downloaded CA Certificate Revocation List (CRL)

Certificate Revocation List (signed)

Certificate Authority

Server
TN3270, FTP daemon, Web server, etc.

Server Certificate
- CA name
- Serial number

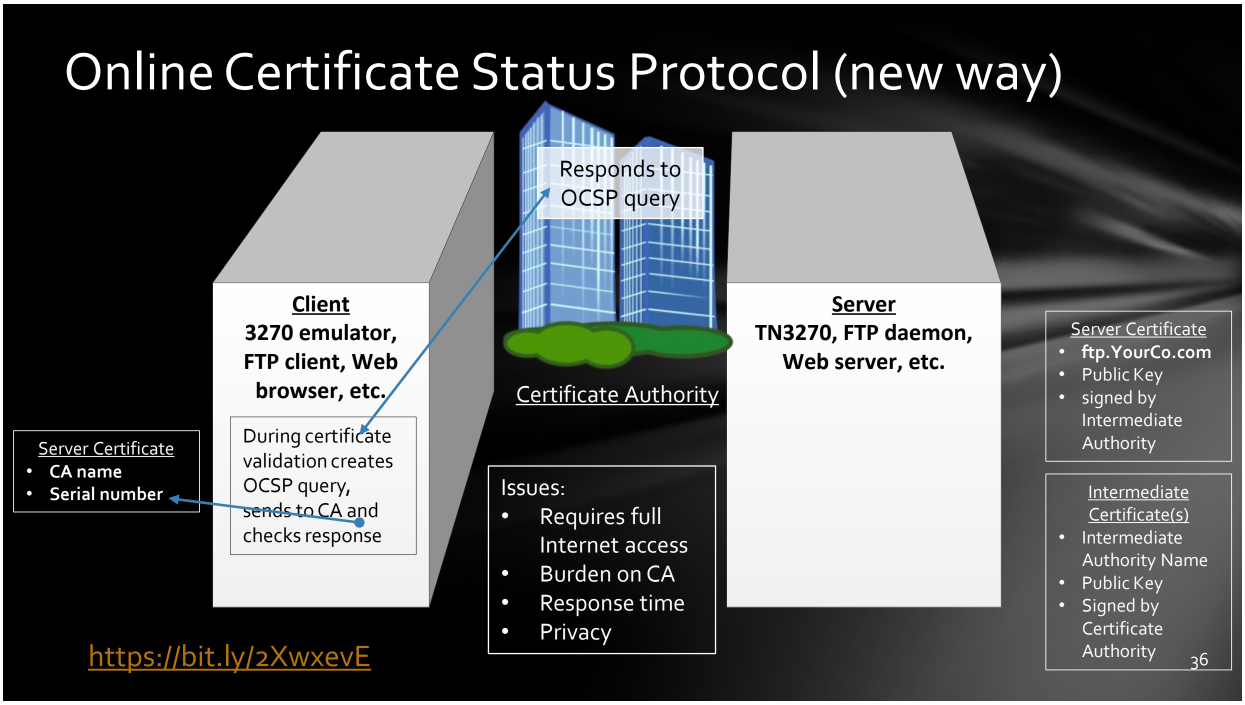
Server Certificate
- **ftp.YourCo.com**
- Public Key
- signed by Intermediate Authority

Intermediate Certificate(s)
- Intermediate Authority Name
- Public Key
- Signed by Certificate Authority

Issues:
- Timeliness
- Size (megabytes!)
- Parsing of large list

https://bit.ly/31h30MM

35



Online Certificate Status Protocol (new way)

Client
3270 emulator, FTP client, Web browser, etc.

During certificate validation creates OCSP query, sends to CA and checks response

Responds to OCSP query

Certificate Authority

Server
TN3270, FTP daemon, Web server, etc.

Server Certificate
- CA name
- Serial number

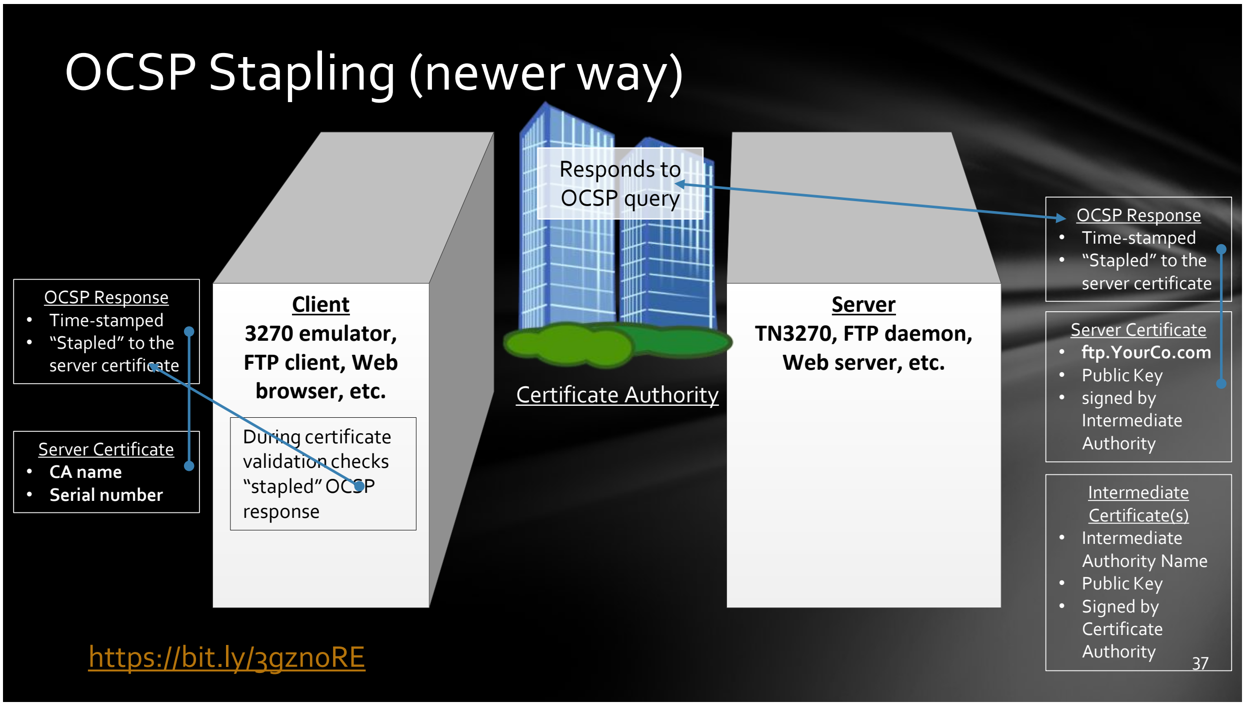
Server Certificate
- **ftp.YourCo.com**
- Public Key
- signed by Intermediate Authority

Intermediate Certificate(s)
- Intermediate Authority Name
- Public Key
- Signed by Certificate Authority

Issues:
- Requires full Internet access
- Burden on CA
- Response time
- Privacy

https://bit.ly/2XwxevE

36



OCSP Stapling (newer way)

Client

3270 emulator, FTP client, Web browser, etc.

During certificate validation checks "stapled" OCSP response

Responds to OCSP query

Certificate Authority

Server

TN3270, FTP daemon, Web server, etc.

OCSP Response
- Time-stamped
- "Stapled" to the server certificate

Server Certificate
- **CA name**
- **Serial number**

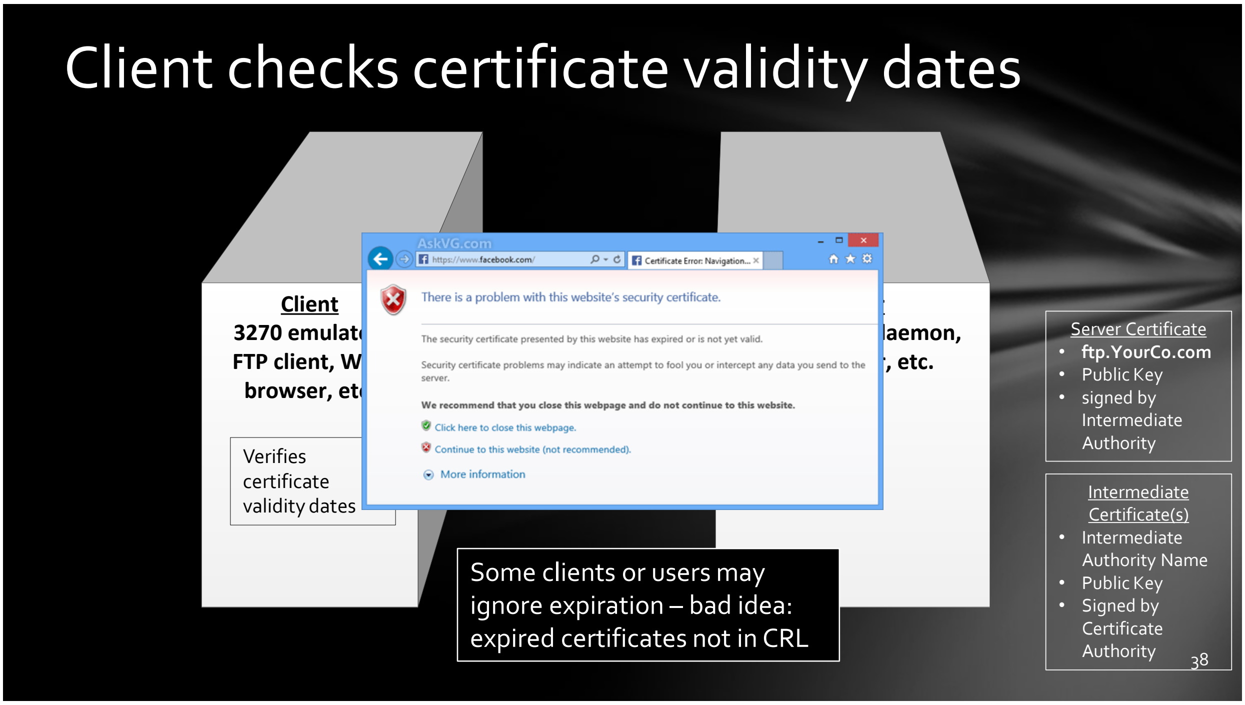
OCSP Response
- Time-stamped
- "Stapled" to the server certificate

Server Certificate
- **ftp.YourCo.com**
- Public Key
- signed by Intermediate Authority

Intermediate Certificate(s)
- Intermediate Authority Name
- Public Key
- Signed by Certificate Authority

https://bit.ly/3gznoRE

37



Client checks certificate validity dates

Client

3270 emulat
FTP client, W
browser, et

Verifies certificate validity dates

laemon,
, etc.

AskVG.com

https://www.facebook.com/

Certificate Error: Navigation...

There is a problem with this website's security certificate.

The security certificate presented by this website has expired or is not yet valid.

Security certificate problems may indicate an attempt to fool you or intercept any data you send to the server.

We recommend that you close this webpage and do not continue to this website.

Click here to close this webpage.

Continue to this website (not recommended).

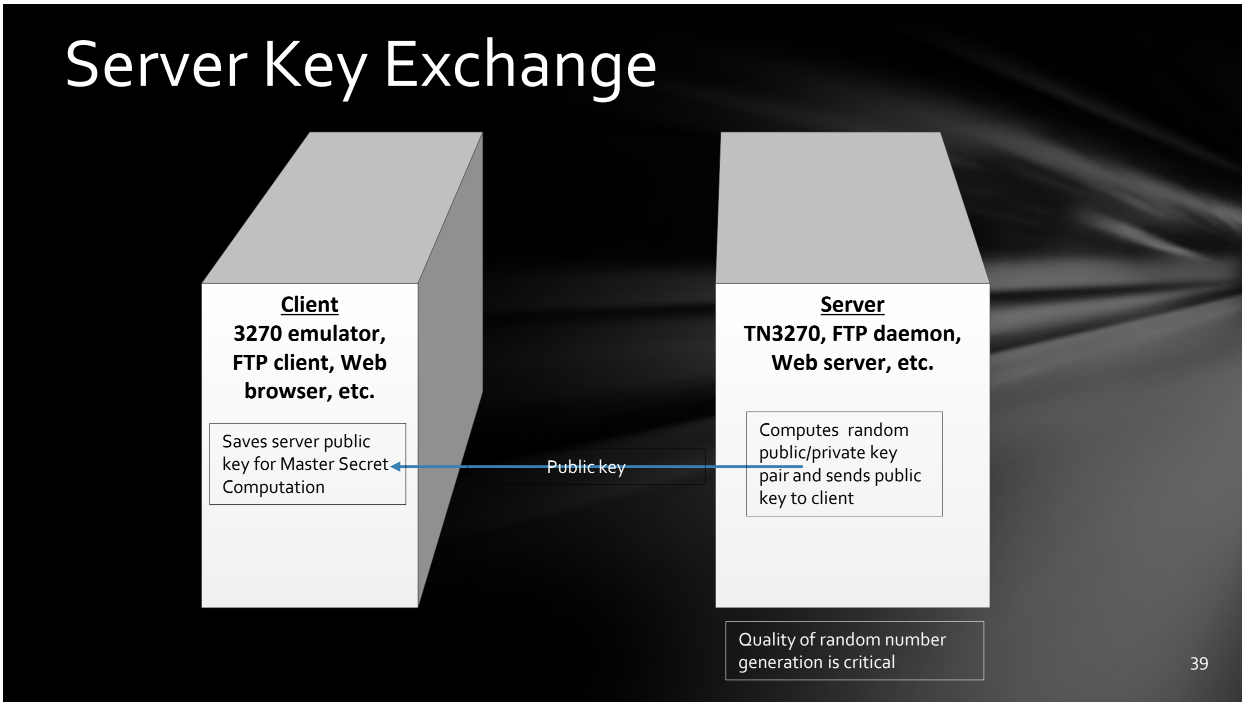
More information

Some clients or users may ignore expiration – bad idea: expired certificates not in CRL

Server Certificate
- **ftp.YourCo.com**
- Public Key
- signed by Intermediate Authority

Intermediate Certificate(s)
- Intermediate Authority Name
- Public Key
- Signed by Certificate Authority

38



Server Key Exchange

Client
3270 emulator, FTP client, Web browser, etc.

Saves server public key for Master Secret Computation

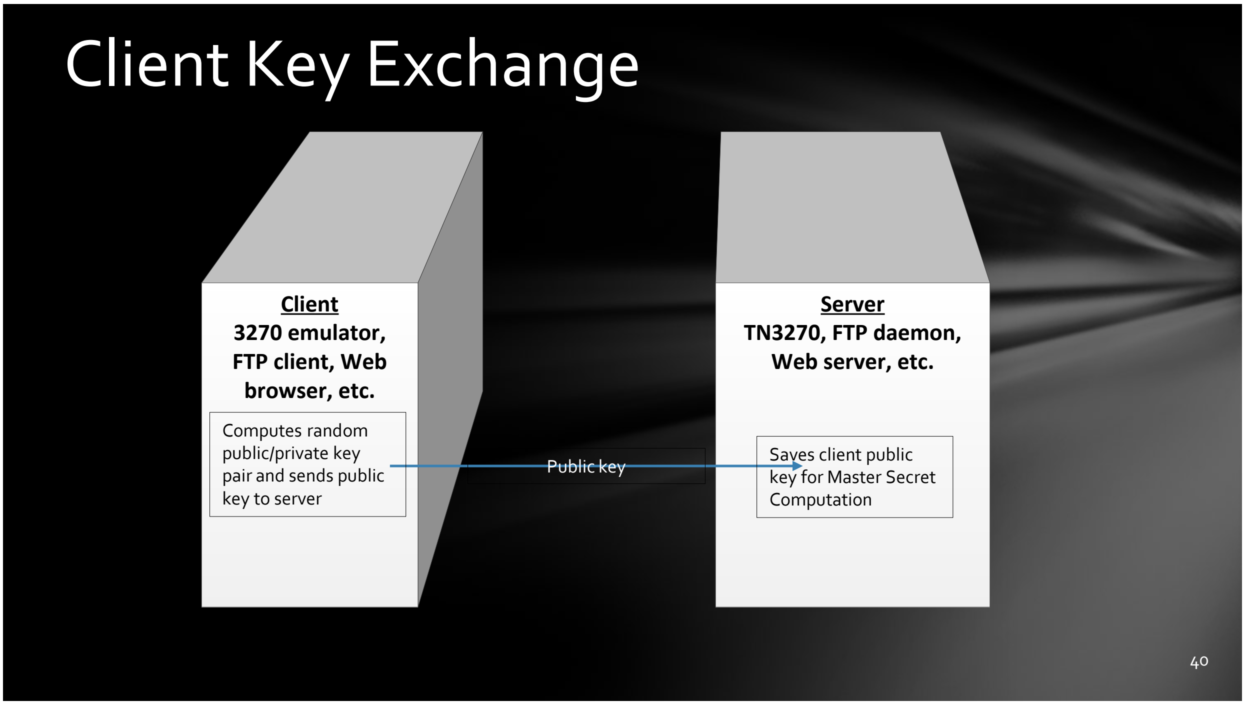
Public key

Server
TN3270, FTP daemon, Web server, etc.

Computes random public/private key pair and sends public key to client

Quality of random number generation is critical

39



Client Key Exchange

Client
3270 emulator, FTP client, Web browser, etc.

Computes random public/private key pair and sends public key to server

Public key

Server
TN3270, FTP daemon, Web server, etc.

Saves client public key for Master Secret Computation

40

Client and Server Compute Master Secret

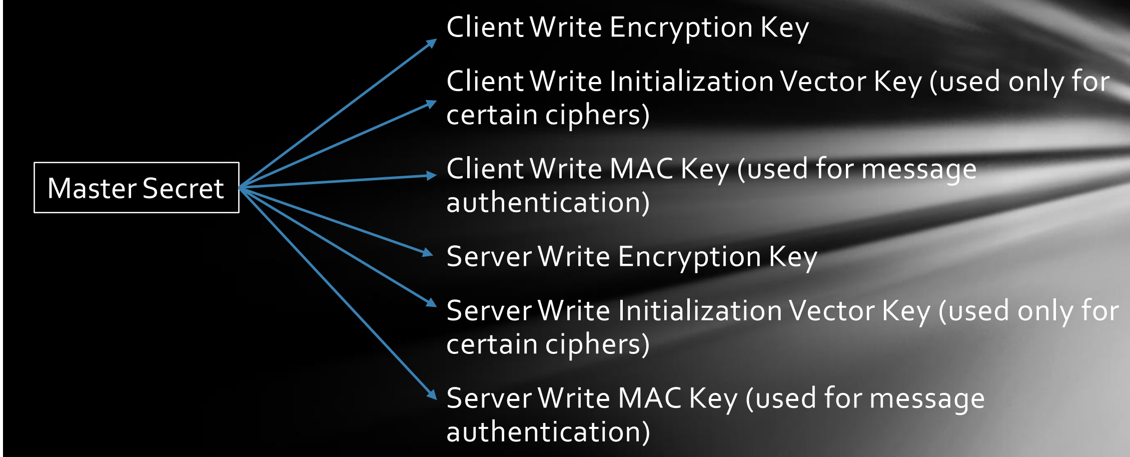
Client and Server each compute a “Premaster Secret”

- Computed from random numbers in Client Hello, Server Hello, its own private key, and the partner’s public key
- Both parties perform the same computation and should get the same result (even though different inputs!)
- Length varies depending on cipher suite

Client and Server each derive the same “Master Secret” from the Premaster Secret

- Always 48 bits

Up to Six session keys derived from master secret

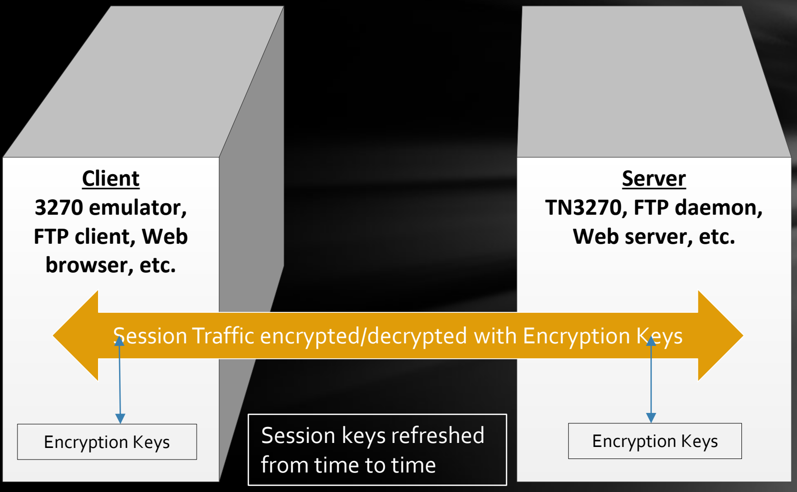


Master Secret

- Client Write Encryption Key
- Client Write Initialization Vector Key (used only for certain ciphers)
- Client Write MAC Key (used for message authentication)
- Server Write Encryption Key
- Server Write Initialization Vector Key (used only for certain ciphers)
- Server Write MAC Key (used for message authentication)

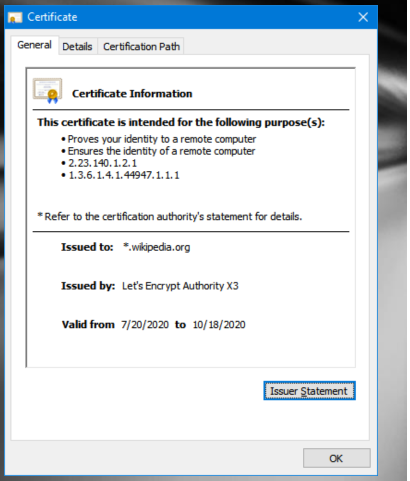
https://bit.ly/3ho66OB

Data traffic at last!


Client
3270 emulator, FTP client, Web browser, etc.
Server
TN3270, FTP daemon, Web server, etc.
Session Traffic encrypted/decrypted with Encryption Keys
Encryption Keys
Session keys refreshed from time to time
Encryption Keys

Certificates Solve the Crucial Problems

- Authentication
- Encryption
- Secure key delivery
- Automation of key delivery
- Bi-directionality
- Speed
- Man in the Middle attack



Certificate Information

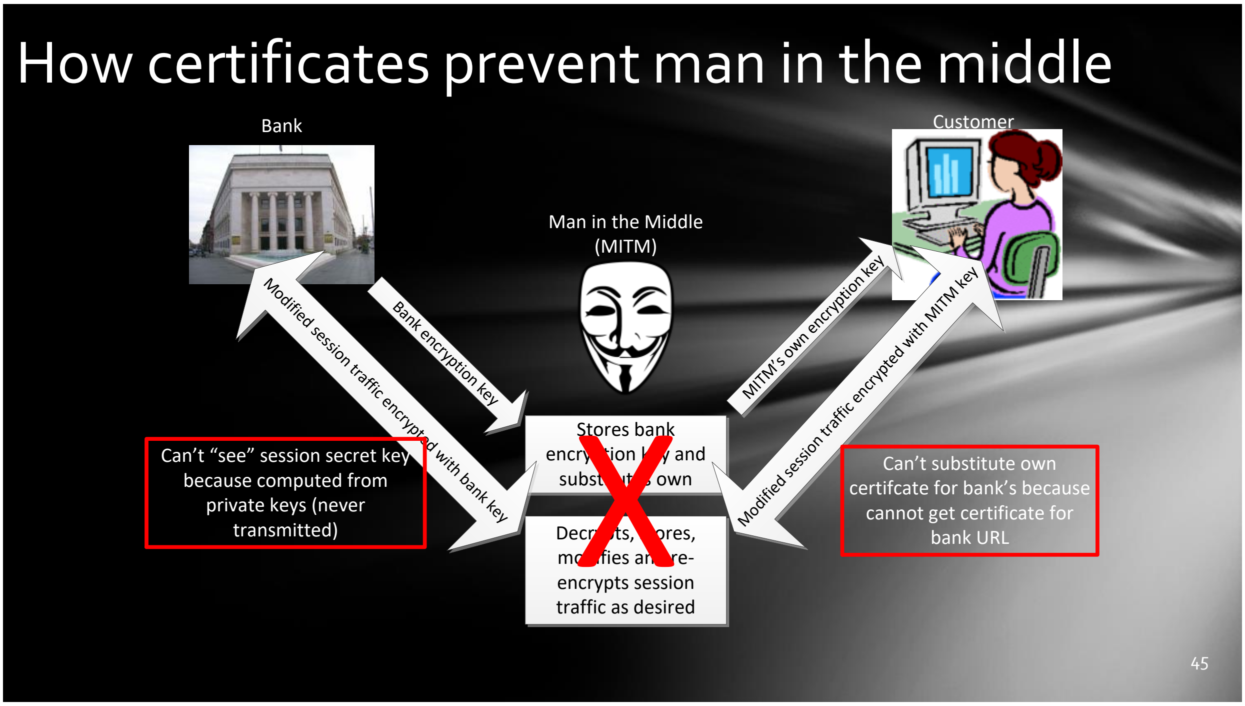
This certificate is intended for the following purpose(s):
- Proves your identity to a remote computer
- Ensures the identity of a remote computer
- 2.23.140.1.2.1
- 1.3.6.1.4.1.44947.1.1.1

* Refer to the certification authority's statement for details.

Issued to: *.wikipedia.org

Issued by: Let's Encrypt Authority X3

Valid from 7/20/2020 **to** 10/18/2020



How certificates prevent man in the middle

Bank

Customer

Man in the Middle (MITM)

Bank encryption key

Modified session traffic encrypted with bank key

MITM's own encryption key

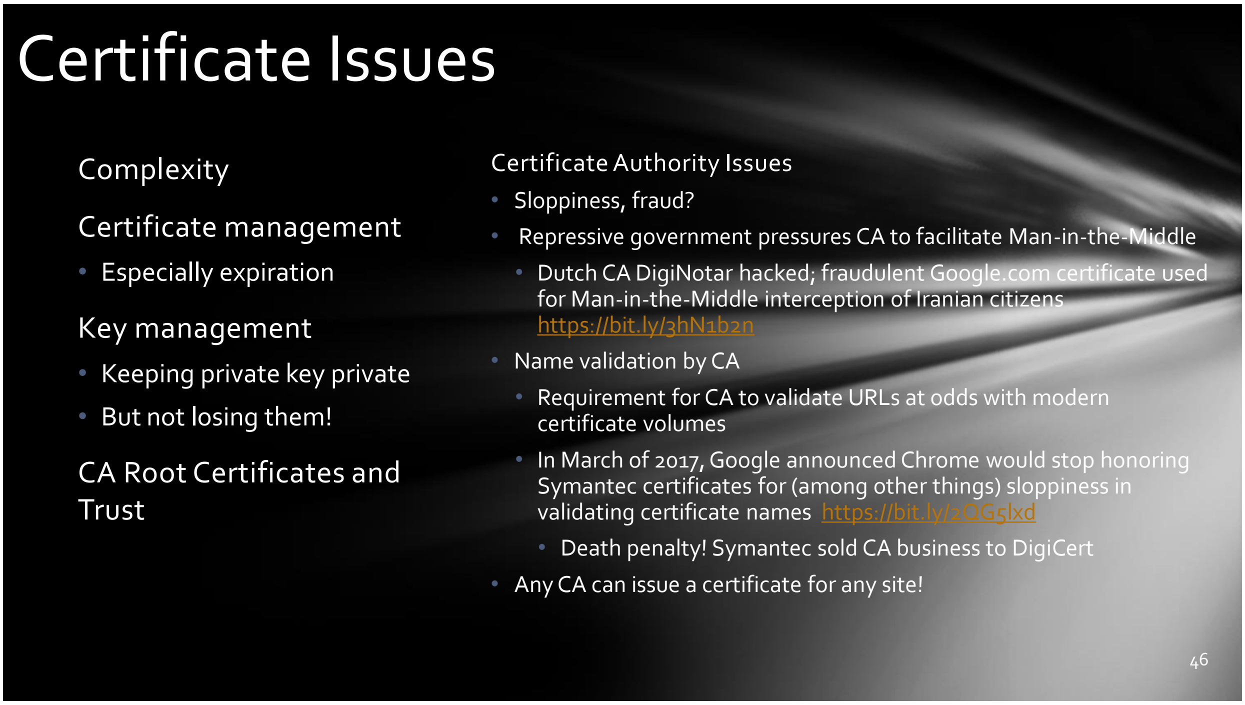
Modified session traffic encrypted with MITM key

Stores bank encryption key and substitutes own

Decrypts, stores, modifies and re-encrypts session traffic as desired

Can't "see" session secret key because computed from private keys (never transmitted)

Can't substitute own certifcate for bank's because cannot get certificate for bank URL



Certificate Issues

Complexity

Certificate management

- Especially expiration

Key management

- Keeping private key private
- But not losing them!

CA Root Certificates and Trust

Certificate Authority Issues

- Sloppiness, fraud?
- Repressive government pressures CA to facilitate Man-in-the-Middle
  - Dutch CA DigiNotar hacked; fraudulent Google.com certificate used for Man-in-the-Middle interception of Iranian citizens https://bit.ly/3hN1b2n
- Name validation by CA
  - Requirement for CA to validate URLs at odds with modern certificate volumes
  - In March of 2017, Google announced Chrome would stop honoring Symantec certificates for (among other things) sloppiness in validating certificate names https://bit.ly/2OG5lxd
    - Death penalty! Symantec sold CA business to DigiCert
- Any CA can issue a certificate for any site!

Questions?

More questions?
charlesm@mcn.org

100 MPH Overview of Some Advanced Features

Self-signed certificates

Misunderstood concept

Self-signing is not inherently bad – all CA root certificates are self-signed

Means the certificate signs itself, not that the company that issued the certificate is its own CA

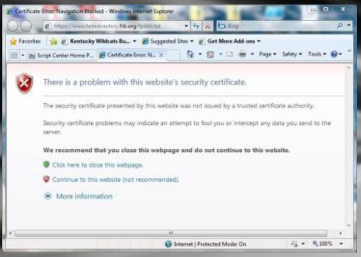
Generally frowned upon for end-point certificates

Provide encryption

Provide authentication only if pre-installed on client

Nothing wrong with your company being its own CA

- Saves money, time and trouble
- Works only for internal clients – external users do not have CA root certificate
- Possibly more secure to control it all yourself



There is a problem with this website's security certificate.

The security certificate presented by this website was not issued by a trusted certificate authority.

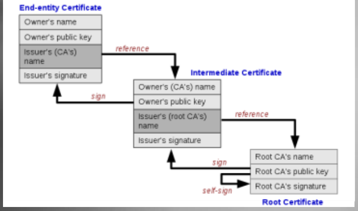
Security certificate problems may indicate an attempt to fool you or intercept any data you send to the server.

We recommend that you close this webpage and do not continue to this website.

Click here to close this webpage.

Continue to this website (not recommended).

More information



Source: Wikipedia
By Yanpas - Own work, CC BY-SA 4.0,
https://commons.wikimedia.org/w/index.php?curid=46369922

Alternative Names

Certificates support multiple "subject alternative names" (SANs) in addition to the main "common name"

Thus one certificate could be valid for YourCo.com, MyCo.com and HerCo.com

Using an Alternative Name for the server URL is now preferred to Common Name (RFC 2818)

Sometimes called a Multi-Domain or SAN Certificate

CA's charge more for multiple names but that is a business issue, not a technical issue

http://bit.ly/2B8AL4Z

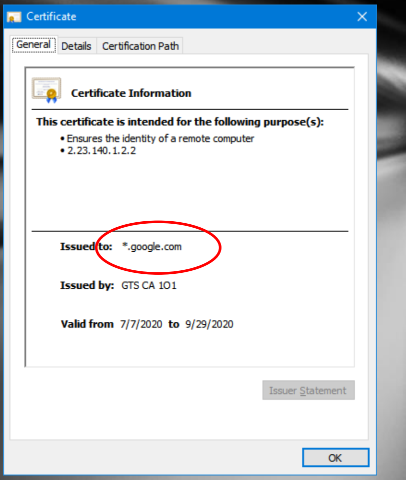
Subject Name Wildcards

Certificates support wildcard subject names (Common or Alternative)

Asterisk may be last or only character of leftmost sub-domain name: `*.YourCo.Com` or `w*.YourCo.Com`

- Or last dotted address octet: `192.168.17.*` (infrequent)

One certificate for www.YourCo.com, ftp.YourCo.com and mail.YourCo.com



Certificate Information

This certificate is intended for the following purpose(s):
- Ensures the identity of a remote computer
- 2.23.140.1.2.2

Issued to: *.google.com

Issued by: GTS CA 1O1

Valid from 7/7/2020 to 9/29/2020

http://bit.ly/2Djbw6g

51

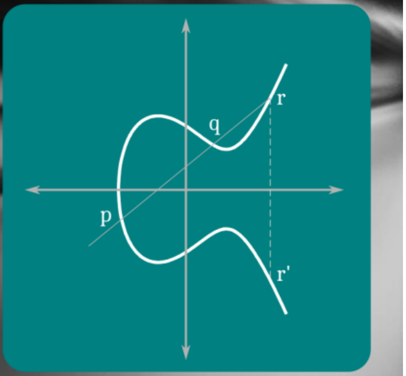
Elliptic Curve Encryption

The problem with RSA encryption

- Principle is that multiplication is fast; factoring is slow
- As computers have gotten faster we have compensated by going to larger and larger RSA keys
- Problem is that the larger the key, the less the difference in time between multiplication and factoring – so diminishing returns

Elliptic Curve Encryption too complex for one slide

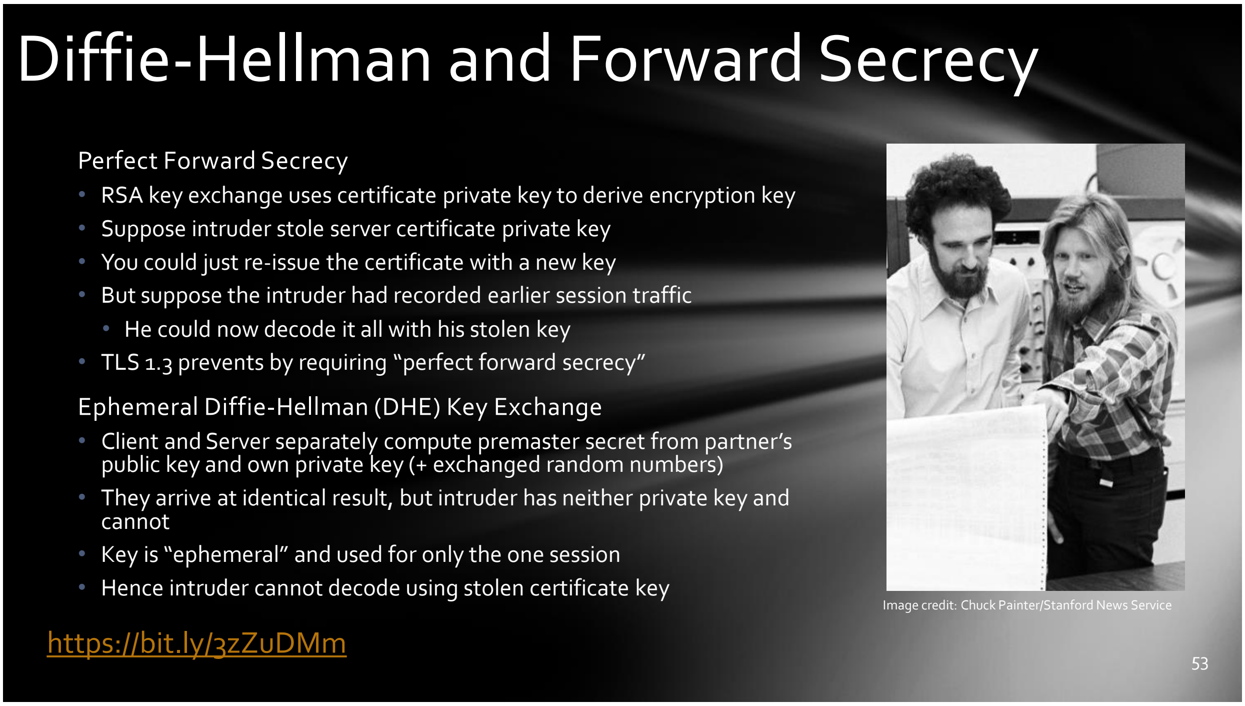
- Relatively fast to compute a transformation based on an elliptic curve
- Very slow to reverse that transformation
- Smaller keys give security equivalent to large RSA keys
- Time ratio constant for larger keys



Public domain. Source freesvg.org

https://bit.ly/3d18Jy7

52



Diffie-Hellman and Forward Secrecy

Perfect Forward Secrecy

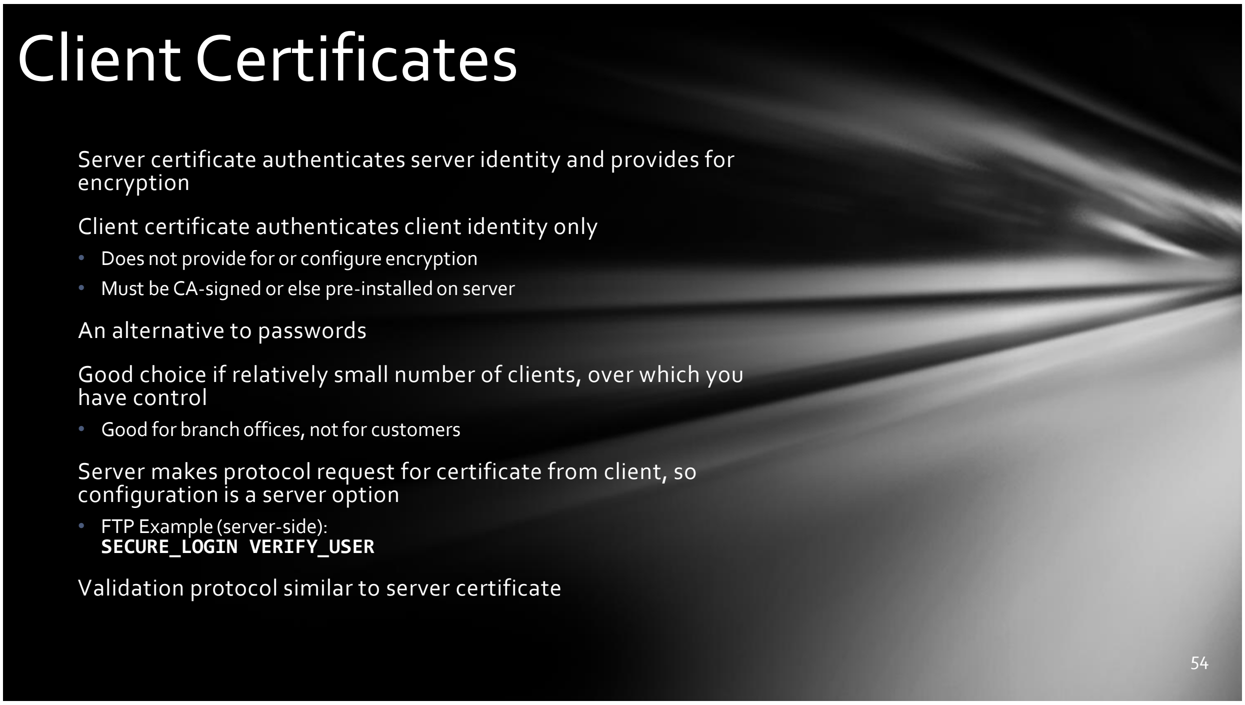
- RSA key exchange uses certificate private key to derive encryption key
- Suppose intruder stole server certificate private key
- You could just re-issue the certificate with a new key
- But suppose the intruder had recorded earlier session traffic
  - He could now decode it all with his stolen key
- TLS 1.3 prevents by requiring "perfect forward secrecy"

Ephemeral Diffie-Hellman (DHE) Key Exchange

- Client and Server separately compute premaster secret from partner's public key and own private key (+ exchanged random numbers)
- They arrive at identical result, but intruder has neither private key and cannot
- Key is "ephemeral" and used for only the one session
- Hence intruder cannot decode using stolen certificate key

Image credit: Chuck Painter/Stanford News Service

https://bit.ly/3zZuDMm



Client Certificates

Server certificate authenticates server identity and provides for encryption

Client certificate authenticates client identity only

- Does not provide for or configure encryption
- Must be CA-signed or else pre-installed on server

An alternative to passwords

Good choice if relatively small number of clients, over which you have control

- Good for branch offices, not for customers

Server makes protocol request for certificate from client, so configuration is a server option

- FTP Example (server-side):
  `SECURE_LOGIN VERIFY_USER`

Validation protocol similar to server certificate

Code signing with certificates

Verifies that software is authentic

Does not prove that code is good, merely authentic!

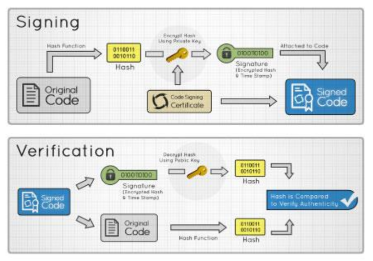
Verifies software has not been altered/tampered with

Requires special code-signing software

May be CA-signed or software-vendor signed

Time-stamping

- Allows for fact that certificate may expire after software is published but before it is installed



Source itcs.com

Constraints and Key Usage

Basic constraint

- CA key or not

Key usage

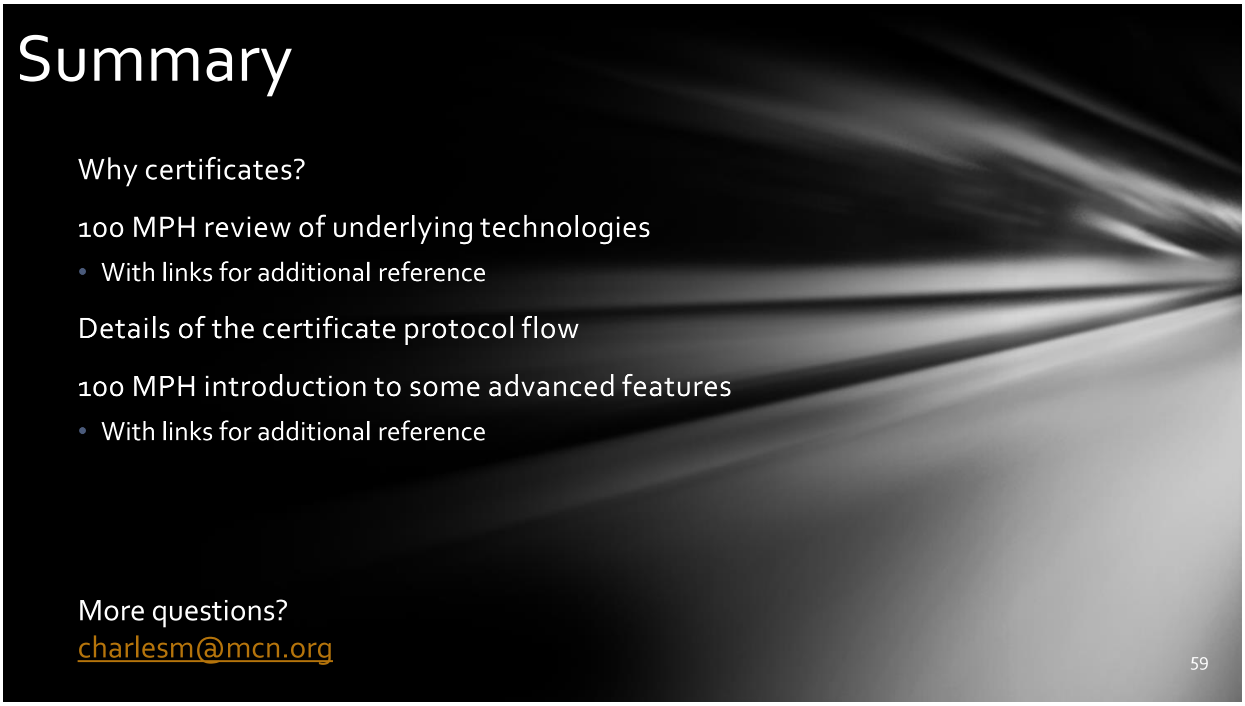
- Signatures
- Etc.

Extended key usage

- Server
- Client
- Code signing
- Email
- Etc.

```
X509v3 extensions:
    X509v3 Basic Constraints:
        CA:FALSE
X509v3 Extended Key Usage:
        TLS Web Server Authentication,
        TLS Web Client Authentication
```

http://bit.ly/2B7lZvy



Summary

Why certificates?

100 MPH review of underlying technologies

- With links for additional reference

Details of the certificate protocol flow

100 MPH introduction to some advanced features

- With links for additional reference

More questions?
charlesm@mcn.org

59

Thank you

More questions?
charlesm@mcn.org

60