Securing Internet Communication: TLS

CS 161: Computer Security

Prof. Vern Paxson

TAs: Paul Bramsen, Apoorva Dornadula,
David Fifield, Mia Gil Epner, David Hahn, Warren He,
Grant Ho, Frank Li, Nathan Malkin, Mitar Milutinovic,
Rishabh Poddar, Rebecca Portnoff, Nate Wang

https://inst.eecs.berkeley.edu/~cs161/

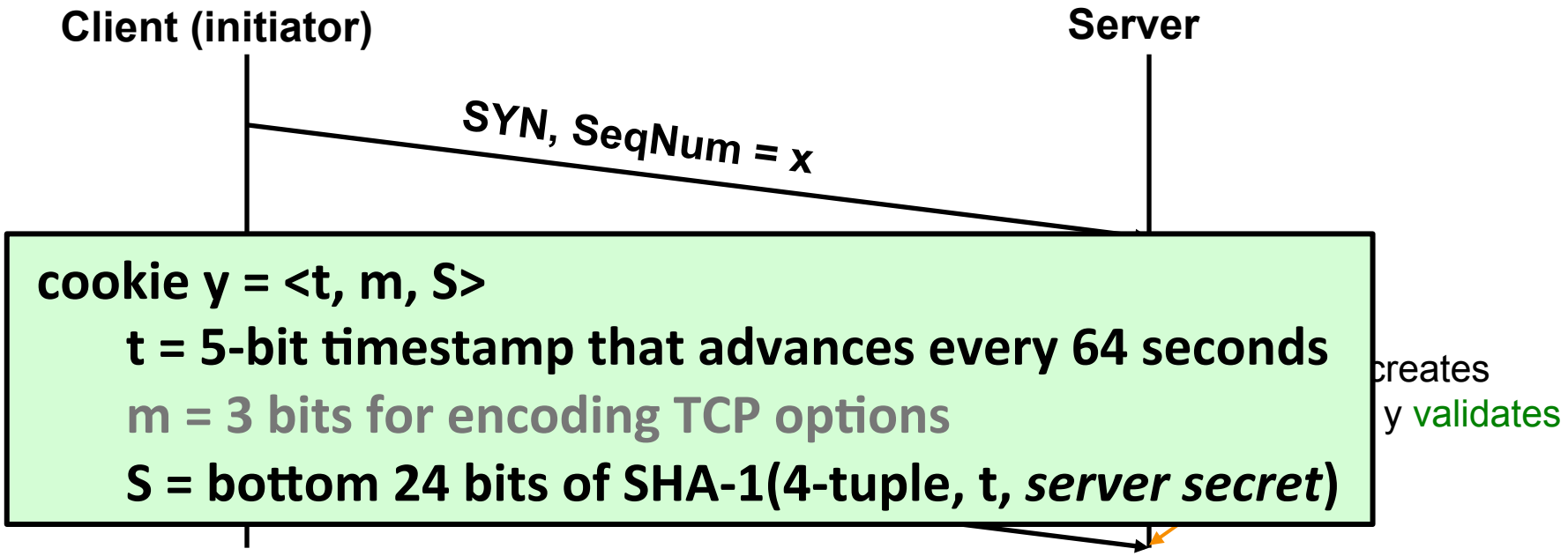
April 6, 2017

Today's Lecture

- Finish discussion of Denial-of-Service (DoS)
- Begin discussion of crypto technology in practice
- Goal #1: overview of the most prominent Internet security protocol
  - SSL/TLS: transport-level (process-to-process) on top of TCP
    - Secures the web via HTTPS
  - (Next lecture: DNSSEC, securing domain name lookups)
- Goal #2: cement understanding of crypto building blocks & how they're used together

Practical Defense: *SYN Cookies*

- Server: when SYN arrives, encode critical state entirely within SYN-ACK's sequence # y !
  - y = *encoding* of necessary state, using server secret
- When ACK of SYN-ACK arrives, server only creates state ***if*** value of **y** from it agrees w/ secret



Cookies: Discussion

- Illustrates general strategy: rather than *holding* state, *encode* it so that it is returned when needed
- For SYN cookies, attacker must complete 3-way handshake in order to burden server
  - *Can't use spoofed source addresses*
- Note #1: strategy requires that you have enough bits to encode all the critical state
  - (This is just barely the case for SYN cookies)
- Note #2: if it's expensive to generate *or check* the cookie, then it's not a win

TCP SYN Flooding, con't

- Approach #4: spread service across lots of different physical servers
  - This is a **general defense** against a wide range of DoS threats (including application-layer)
  - If servers are at different places around the network, protects against *network-layer* DoS too
- But: costs $$
- And: some services are not easy to divide up
  - Such as when need to modify common database
    - E.g. a multi-player real-time game

Application-Layer DoS

- Rather than exhausting network or memory resources, attacker can overwhelm a service's processing capacity
- There are many ways to do so, often at little expense to attacker compared to target (*asymmetry*)

The link sends a request to the web server that requires heavy processing by its backend database.

Application-Layer DoS, con't

- Rather than exhausting network or memory resources, attacker can overwhelm a service's processing capacity
- There are many ways to do so, often at little expense to attacker compared to target (asymmetry)
- Defenses against such attacks?
- Approach #1: Only let legit users to issue expensive requests
  - Relies on being able to identify/authenticate them
  - Note: that *this itself might be expensive*!
- Approach #2: Look for clusters of similar activity
  - Arms race w/ attacker AND costs *collateral damage*
- Approach #3: distribute service across multiple physical servers ($$$)

Securing
Internet Communication

Channel vs. Object Security

- Channel security = securing a means of communication
- Object security = securing data values
- CIA applies to both of them
  - But with different design implications
- TLS provides *channel security*

Building Secure End-to-End Channels

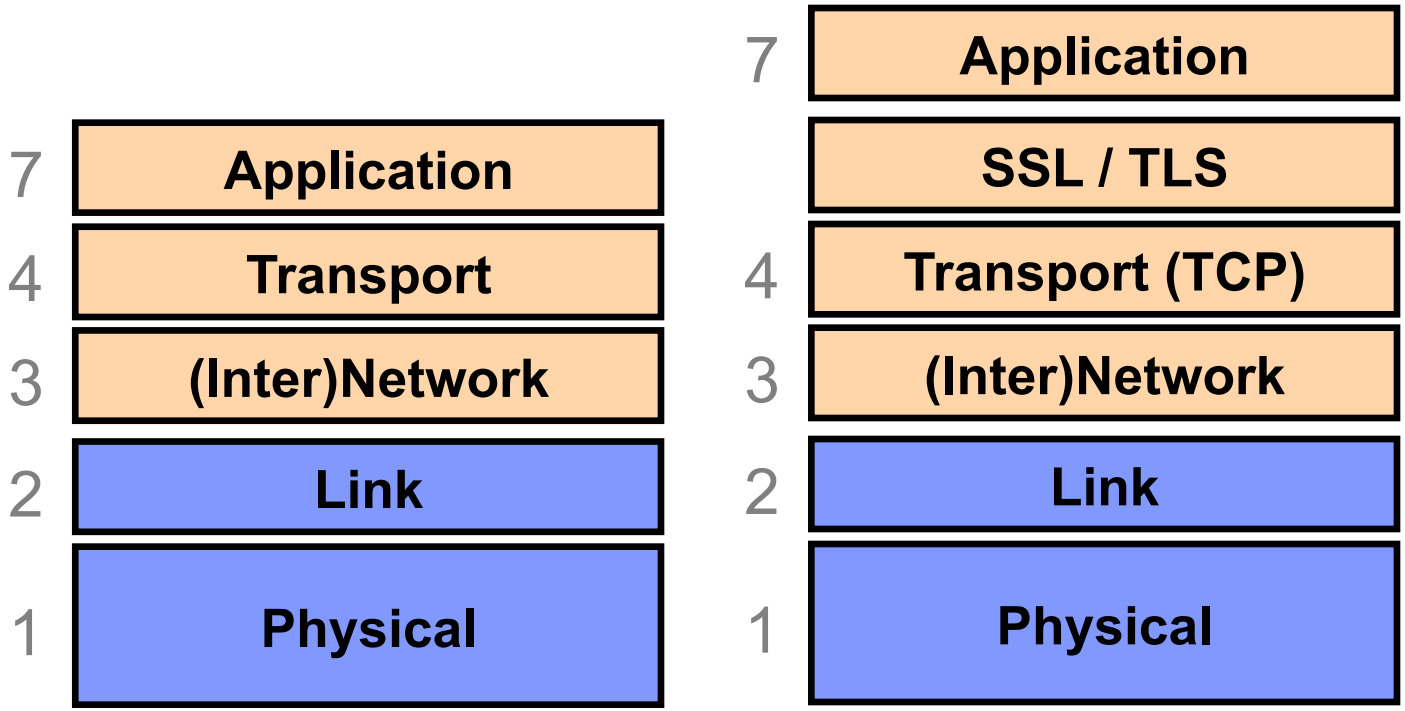
- *End-to-end* = communication protections achieved all the way from originating client to intended server
  - *With no need to trust intermediaries*
- Dealing with threats:
  - Eavesdropping?
    - Encryption (including session keys)
  - Manipulation (injection, MITM)?
    - Integrity (use of a MAC); *replay protection*
  - Impersonation?
    - Signatures

(What's missing? *Availability …*)

Building A Secure End-to-End Channel: SSL/TLS

- SSL = *Secure Sockets Layer* (predecessor)
- TLS = *Transport Layer Security* (standard)
  - Both terms used interchangeably
- Notion: provide means to secure *any* application that uses TCP

SSL/TLS In Network Layering



Building A Secure End-to-End Channel: SSL/TLS

- SSL = *Secure Sockets Layer* (predecessor)
- TLS = *Transport Layer Security* (standard)
  - Both terms used interchangeably
- Notion: provide means to secure *any* application that uses TCP
  - Secure = encryption/confidentiality + integrity + authentication (of server, but typ. *not* of client)
  - E.g., puts the ‘s’ in “https”

Home | University of Califo...

www.berkeley.edu

Search

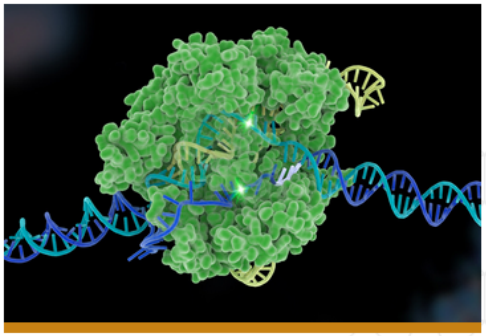
Eight graduate student semi-finalists compete in the Grad Slam competition

Turbocharging photosynthesis

BERKELEY BLOG

This NIH program is crucial to global health — and its future is in danger

by Arthur Reingold



European Patent Office to grant UC a broad patent on CRISPR-Cas9

@UCBERKELEY

Largest single $$ gift to acquire #art in UC Berkeley history brings work of major 20th-century artist to campus… twitter.com/i/web/status/8…

25 minutes ago

Creating an AR/VR lab at Berkeley

Berkeley
UNIVERSITY OF CALIFORNIA

Visit Jobs News Blog Events Give

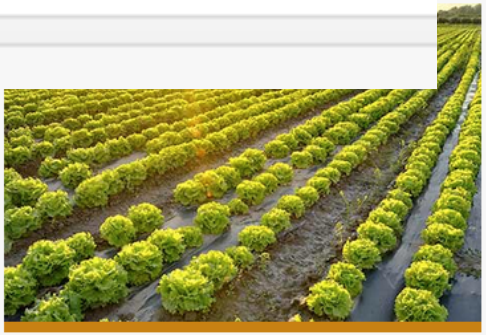
Libraries
Cal Athletics
Equity, Inclusion & Diversity
Accreditation
Websites A-Z
Directory
Map
Contact us

Regular web surfing – `http:` URL

Home | University of Ca

www.berkeley.edu

Eight graduate student semi-finalists compete in the Grad Slam competition

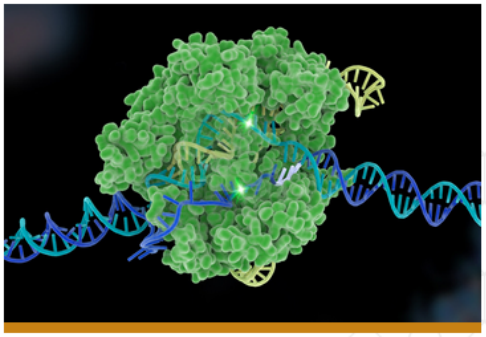


Turbocharging photosynthesis

BERKELEY BLOG

This NIH program is crucial to global health — and its future is in danger

by Arthur Reingold



European Patent Office to grant UC a broad patent on CRISPR-Cas9

@UCBERKELEY

Largest single $$ gift to acquire #art in UC Berkeley history brings work of major 20th-century artist to campus… twitter.com/i/web/status/8…

25 minutes ago

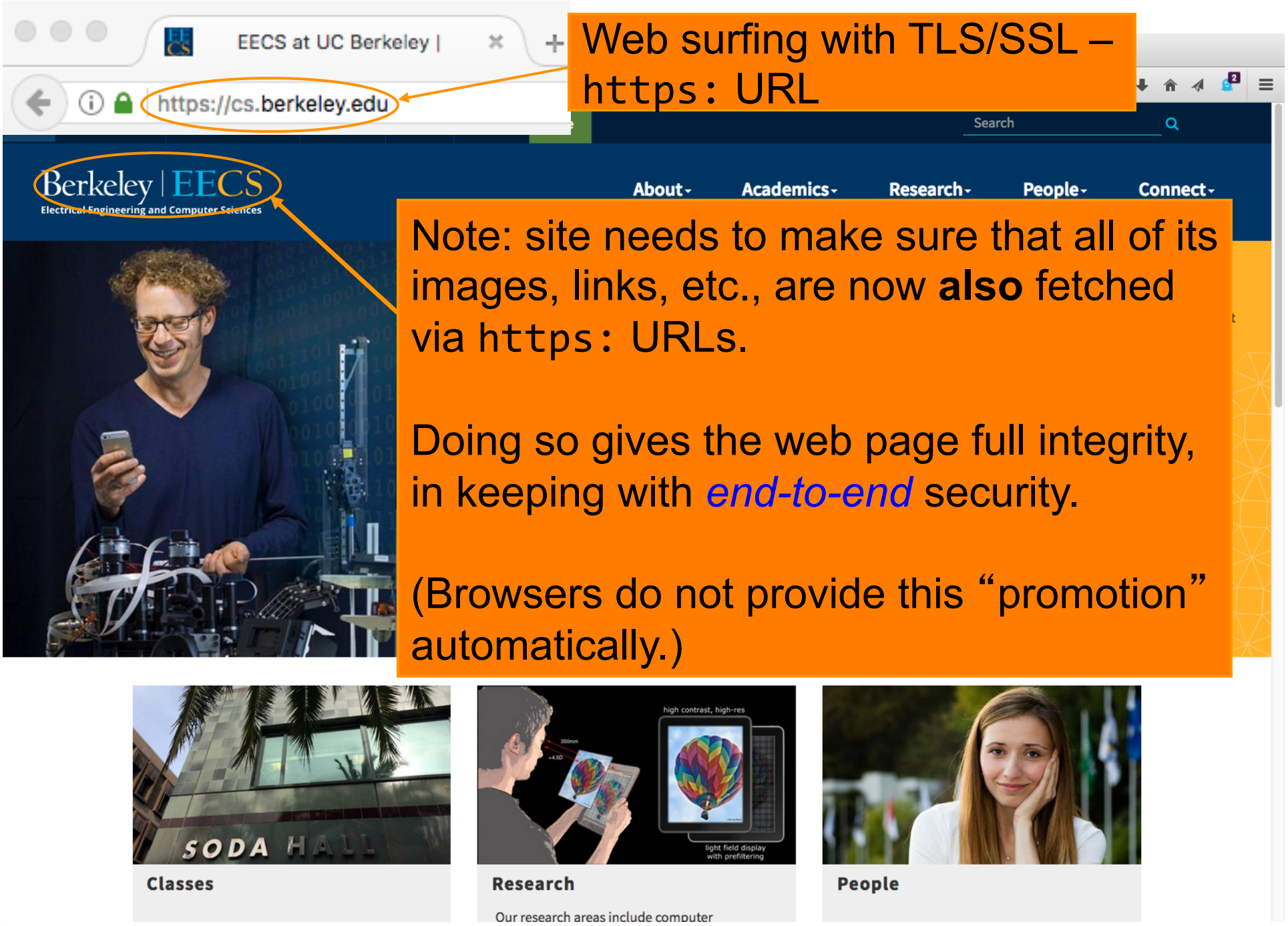
Creating an AR/VR lab at Berkeley

Berkeley
UNIVERSITY OF CALIFORNIA

Visit Jobs News Blog Events Give

Libraries
Cal Athletics
Equity, Inclusion & Diversity
Accreditation

Websites A-Z
Directory
Map
Contact us



Web surfing with TLS/SSL – `https:` URL

Note: site needs to make sure that all of its images, links, etc., are now **also** fetched via `https:` URLs.

Doing so gives the web page full integrity, in keeping with *end-to-end* security.

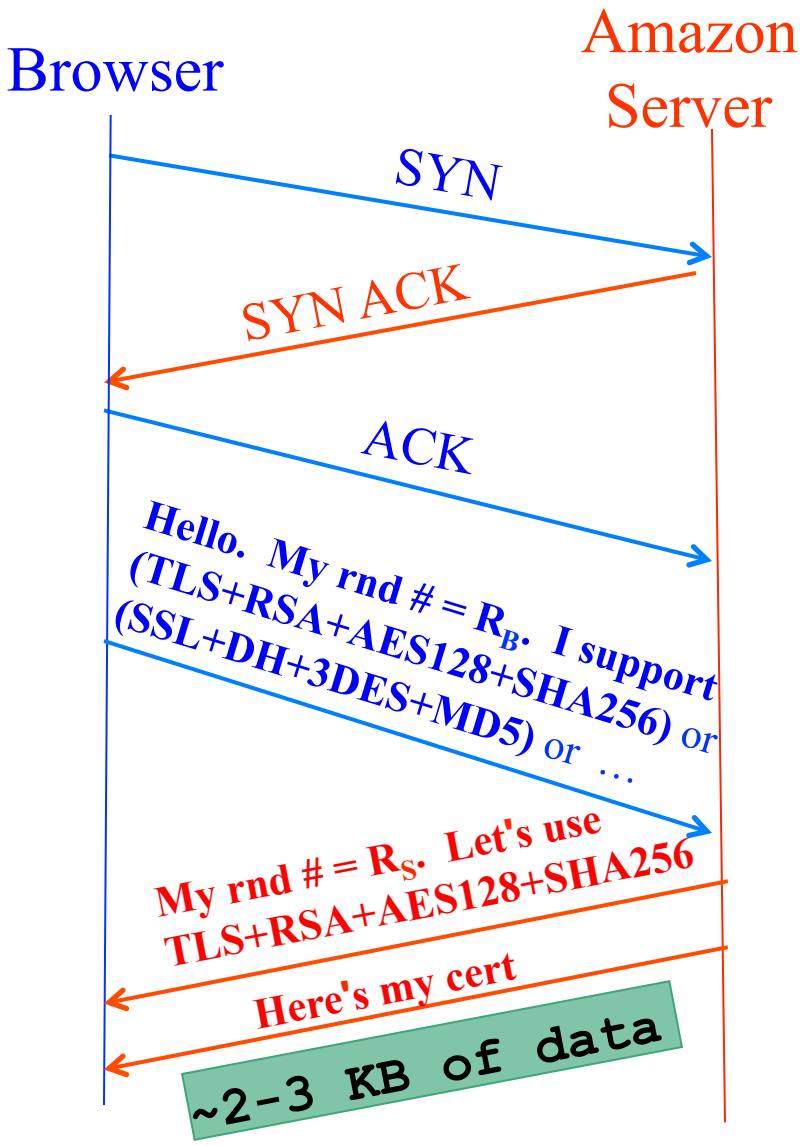
(Browsers do not provide this "promotion" automatically.)

Building A Secure End-to-End Channel: SSL / TLS

- SSL = *Secure Sockets Layer* (predecessor)
- TLS = *Transport Layer Security* (standard)
  - Both terms used interchangeably
- Notion: provide means to secure *any* application that uses TCP
  - Secure = encryption/confidentiality + integrity + authentication (of server, but typ. not of client)
  - E.g., puts the 's' in "https"
- API similar to "socket" interface used for regular network programming
  - Fairly easy to convert an app to be secured

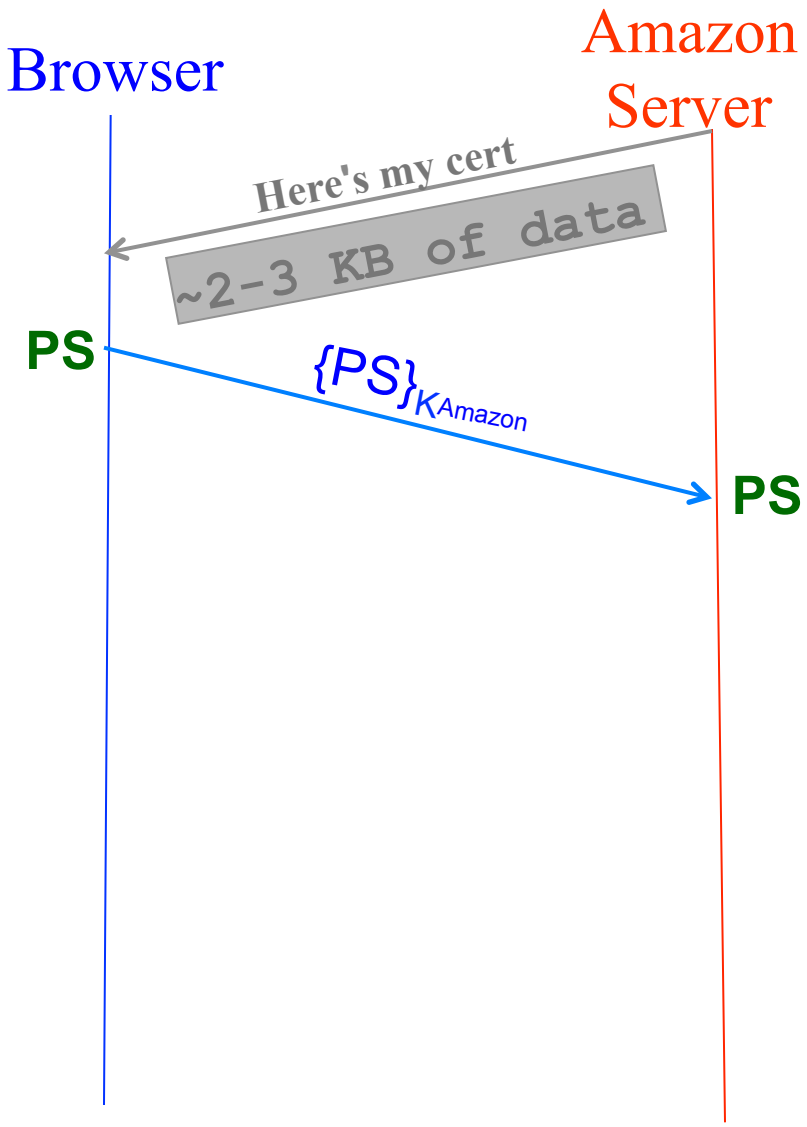
HTTPS Connection (SSL / TLS)

- Browser (client) connects via TCP to Amazon's **HTTPS** server
- Client picks 256-bit random number R_B, sends over list of crypto protocols it supports
- Server picks 256-bit random number R_S, selects *cipher suite* to use for this session
- Server sends over its certificate
- (all of this is in the clear)
- ***Client now validates cert***



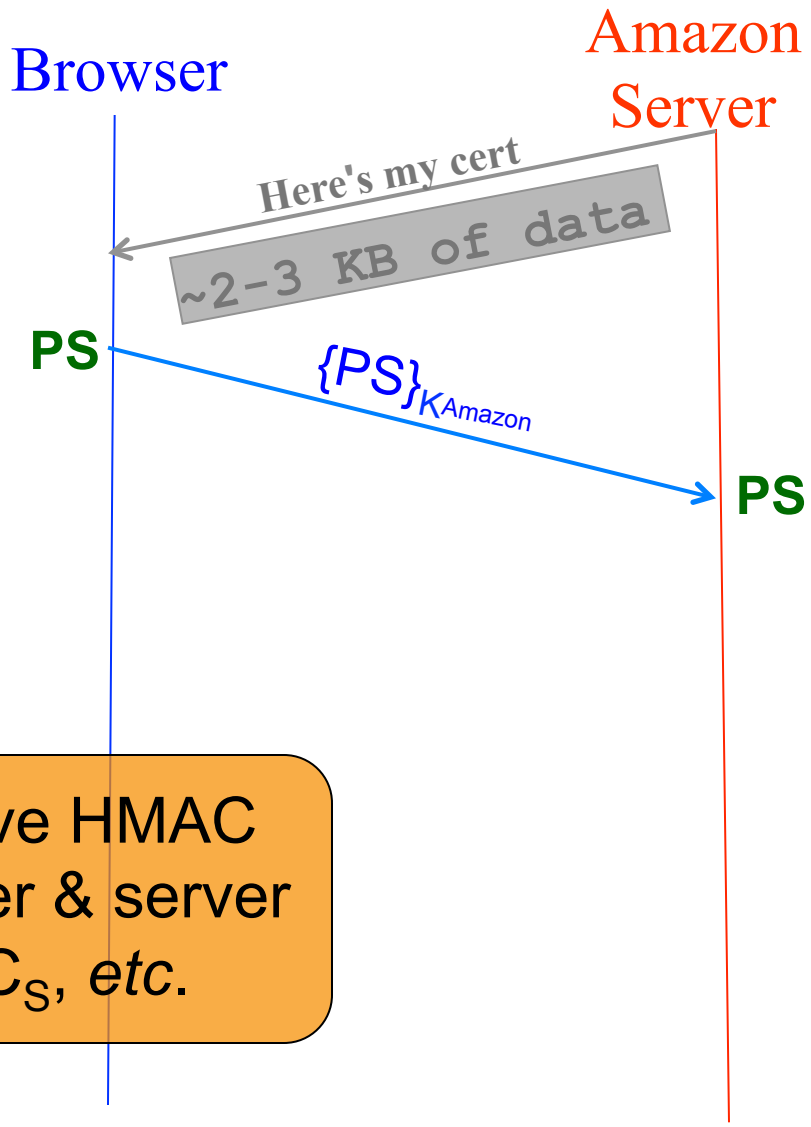
HTTPS Connection (SSL / TLS), con't

- For RSA, browser constructs long (368 bits) "Premaster Secret" **PS**
- Browser sends PS encrypted using Amazon's public RSA key K_{Amazon}
- Using PS, R_B, and R_S, browser & server derive symm. *cipher keys* (C_B, C_S) & MAC *integrity keys* (I_B, I_S)
  - One pair to use in each direction



HTTPS Connection (SSL / TLS), con't

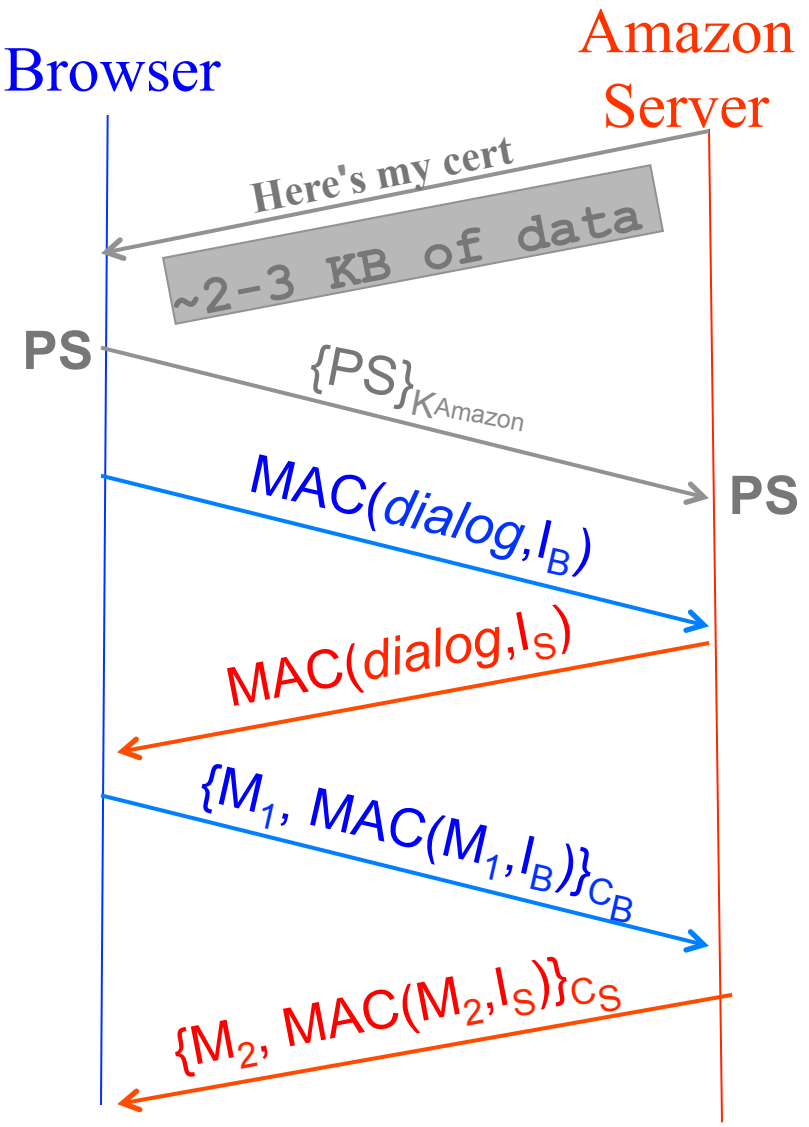
- For RSA, browser constructs long (368 bits) "Premaster Secret" **PS**
- Browser sends PS encrypted using Amazon's public RSA key K_{Amazon}
- Using PS, R_B, and R_S, browser & server derive symm. *cipher keys* (C_B, C_S) & MAC *integrity keys* (I_B, I_S)
  - One pair to use in each direction



PS is used as the key for iterative HMAC invocations on R_B || R_S. Browser & server use the output to generate C_B, C_S, *etc*.

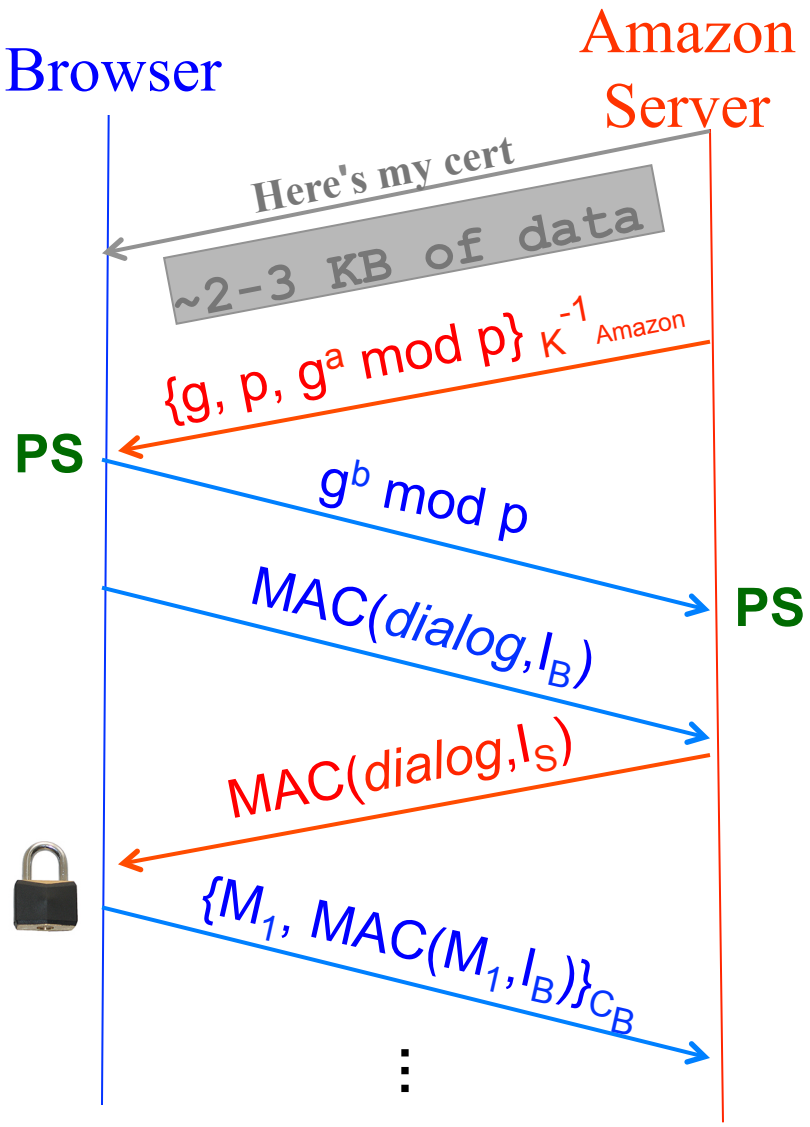
HTTPS Connection (SSL / TLS), con't

- For RSA, browser constructs long (368 bits) "Premaster Secret" **PS**
- Browser sends PS encrypted using Amazon's public RSA key K_{Amazon}
- Using PS, R_B, and R_S, browser & server derive symm. *cipher keys* (C_B, C_S) & MAC *integrity keys* (I_B, I_S)
  - One pair to use in each direction
- Browser & server exchange MACs computed over entire dialog so far
- If good MAC, Browser displays 
- All subsequent communication encrypted w/ symmetric cipher (e.g., AES128) cipher keys, MACs
  - Messages also numbered to thwart replay attacks



Alternative: Key Exchange via Diffie-Hellman

- For Diffie-Hellman, server generates random a, sends public params and g^a mod p
  - Signed with server's public key
- Browser verifies signature
- Browser generates random b, computes **PS** = g^{ab} mod p, sends to server
- Server also computes **PS** = g^{ab} mod p
- Remainder is as before: from PS, R_B, and R_S, browser & server derive symm. *cipher keys* (C_B, C_S) and MAC *integrity keys* (I_B, I_S), etc…

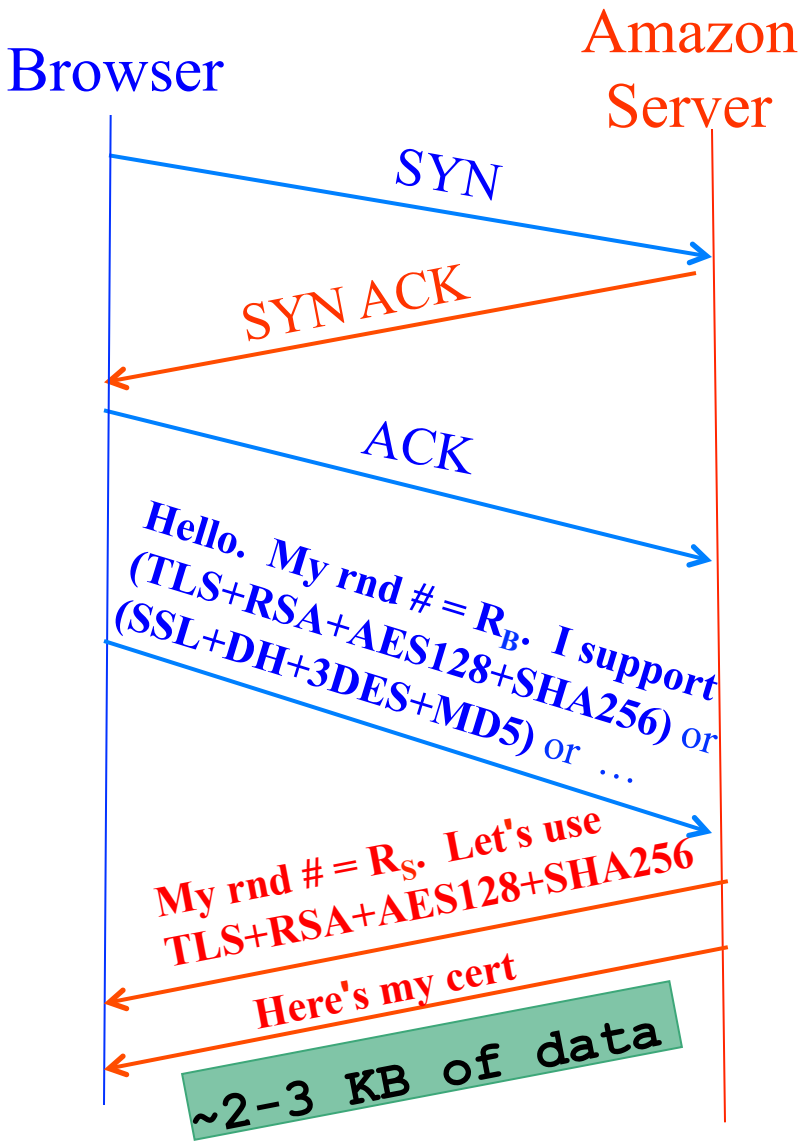


5 Minute Break

Questions Before We Proceed?

HTTPS Connection (SSL / TLS)

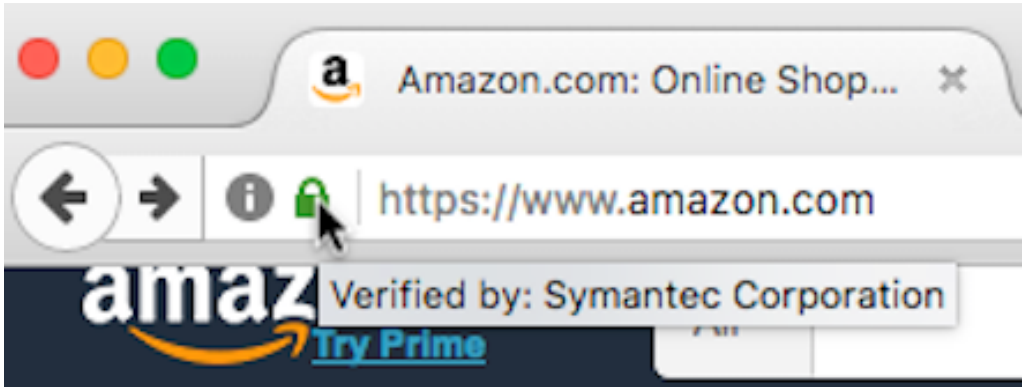
- Browser (client) connects via TCP to Amazon's **HTTPS** server
- Client picks 256-bit random number R_B, sends over list of crypto protocols it supports
- Server picks 256-bit random number R_S, selects *cipher suite* use for this session
- Server sends over its certificate
- (all of this is in the clear)
- ***Client now validates cert***

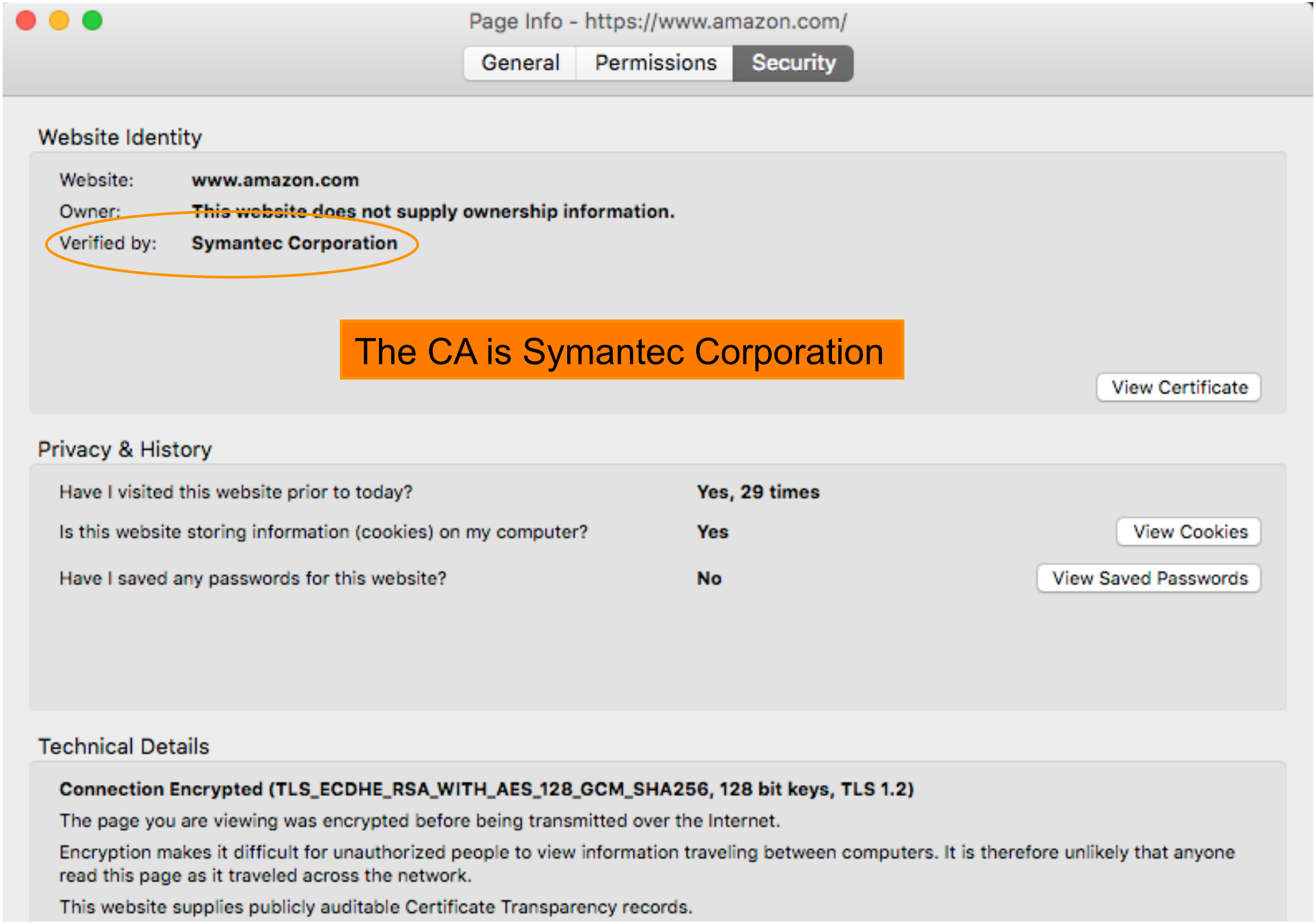


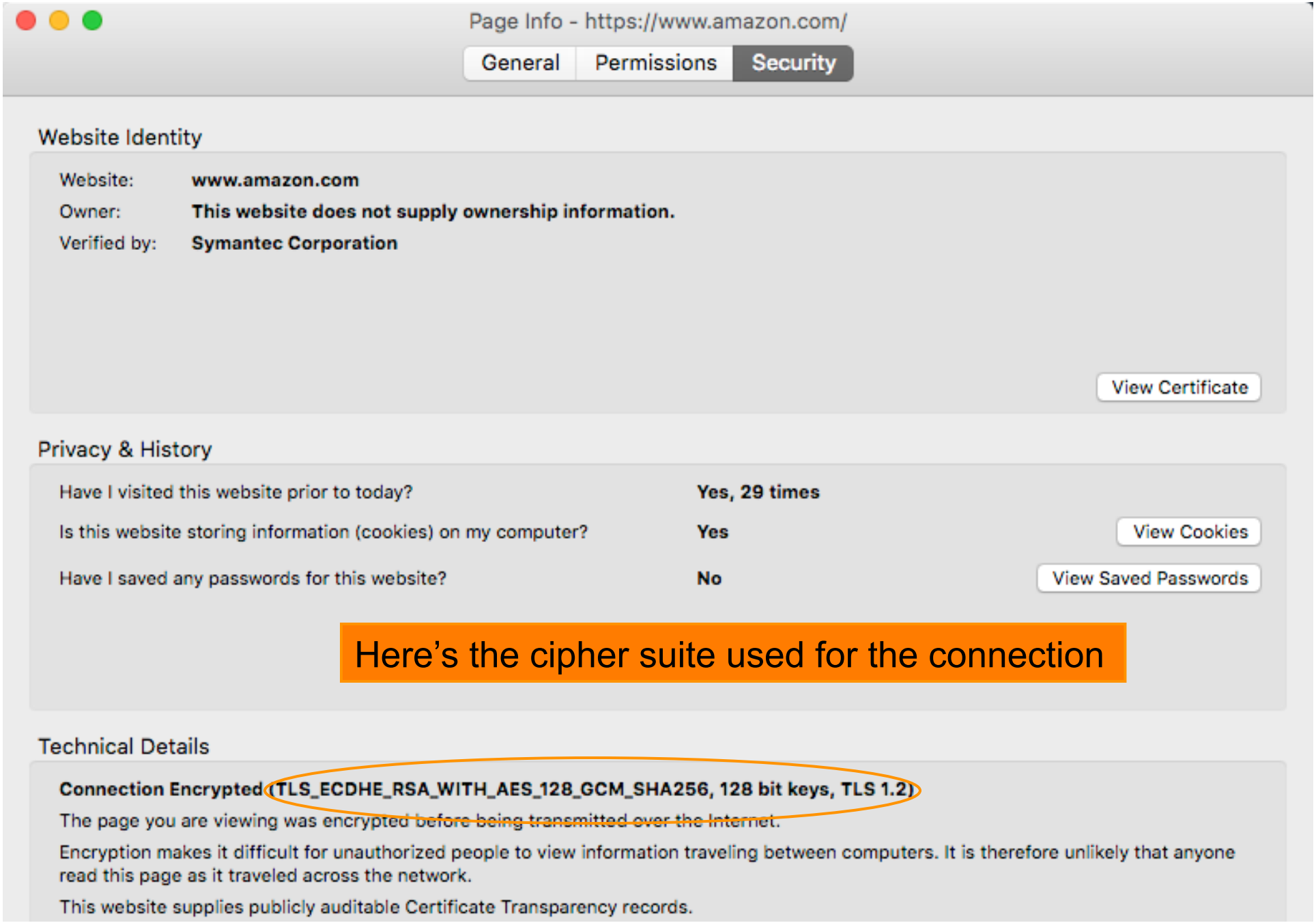
Certificates

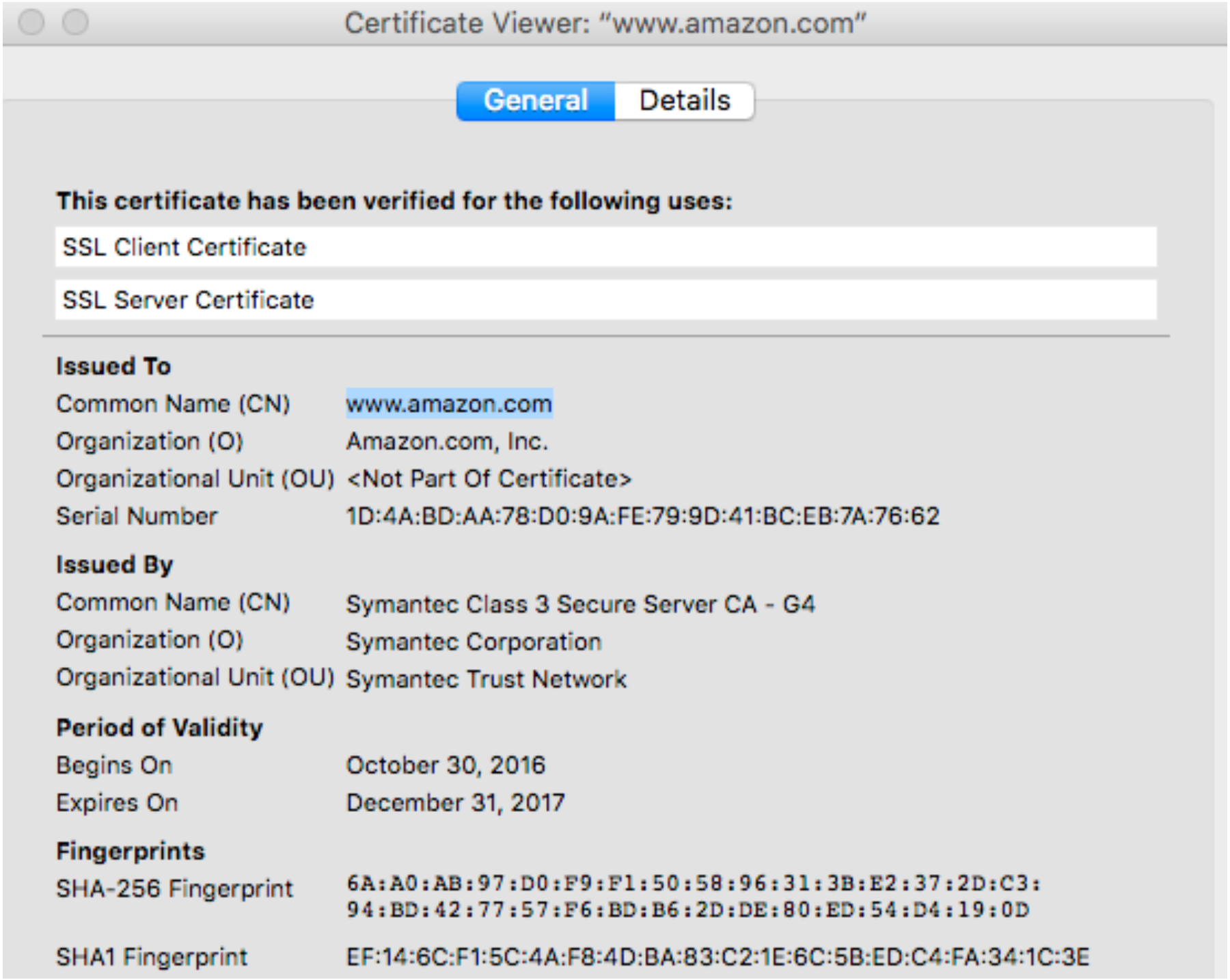
- Cert = signed statement about someone's public key
  - Note that a cert does not say anything about the identity of who **gives** you the cert
  - It simply states a given public key K_{Bob} belongs to Bob …
    - … and backs up this statement with a digital signature made using a different public/private key pair, say from Alice
- Bob then can prove his identity to you by *you sending him* something encrypted with K_{Bob} …
  - … which he then demonstrates he can read
- … or by *signing* something he demonstrably uses
- Works provided you trust that you have a valid copy of *Alice's* public key …
  - … and you trust Alice to use prudence when she signs other people's keys, such as Bob's

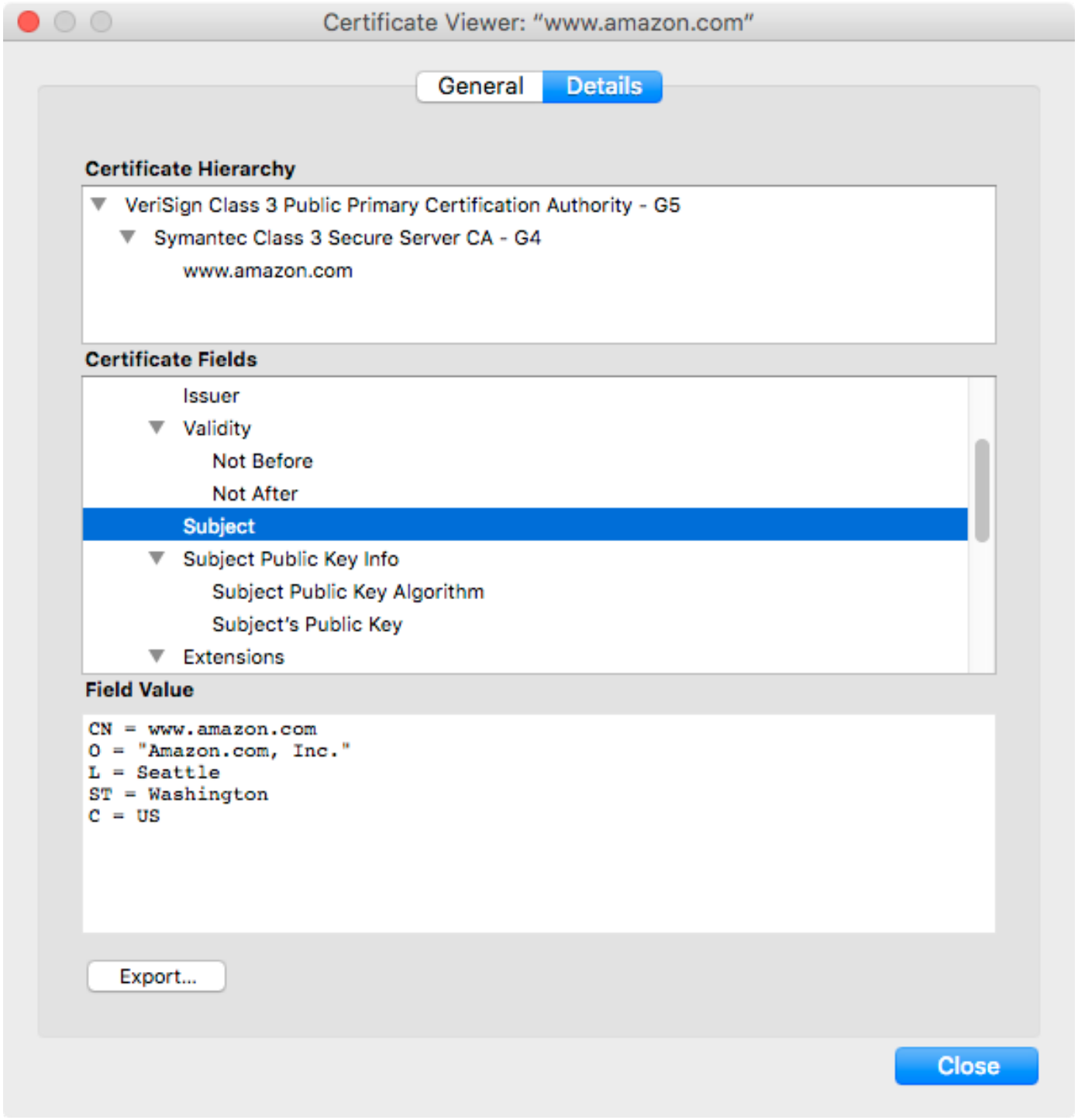
What's Inside Amazon's Cert?

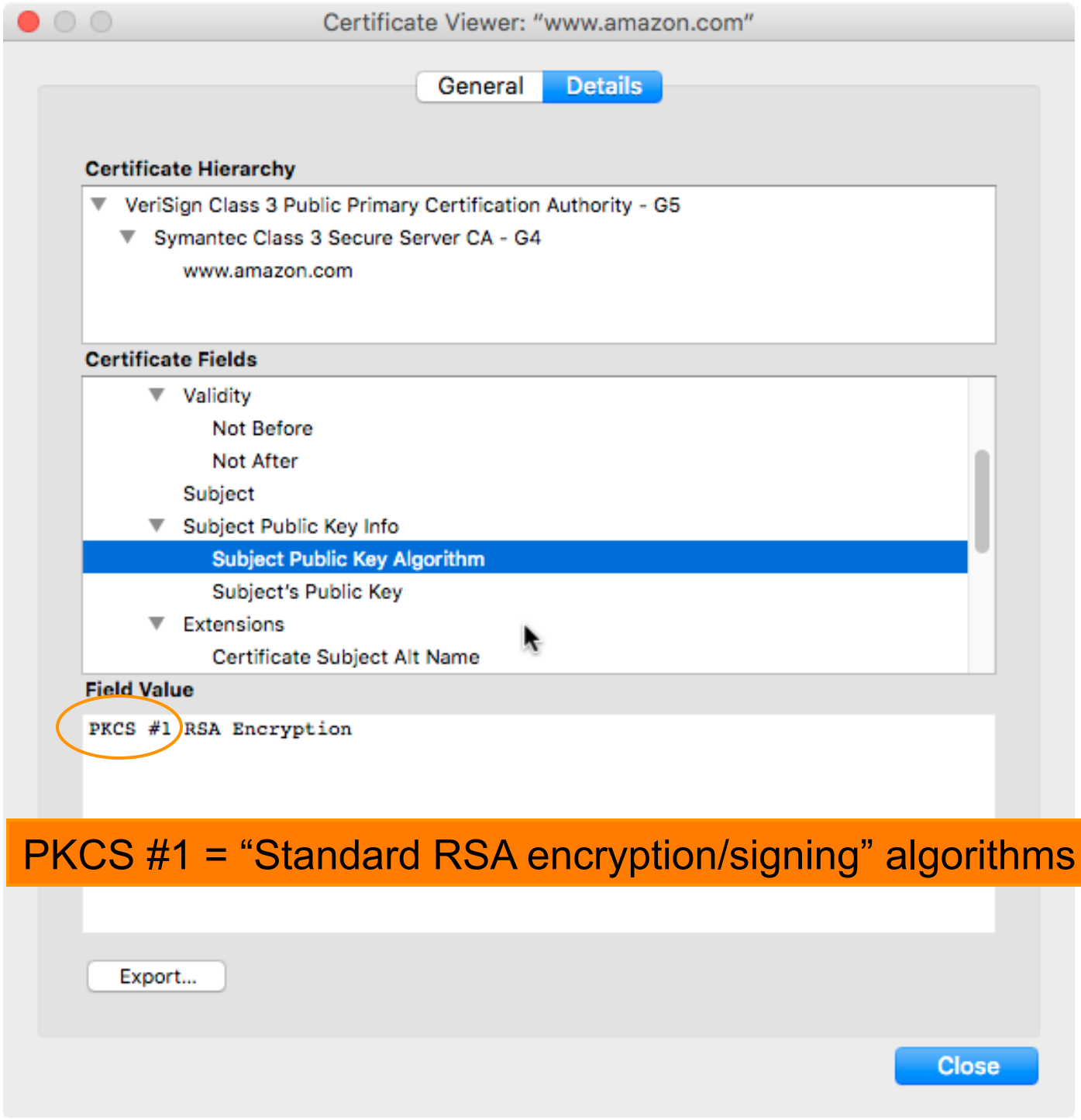

Amazon.com: Online Shop...
https://www.amazon.com
Verified by: Symantec Corporation
Try Prime


Page Info - https://www.amazon.com/
General
Permissions
Security
Website Identity
Website: www.amazon.com
Owner: This website does not supply ownership information.
Verified by: Symantec Corporation
The CA is Symantec Corporation
View Certificate
Privacy & History
Have I visited this website prior to today? Yes, 29 times
Is this website storing information (cookies) on my computer? Yes
View Cookies
Have I saved any passwords for this website? No
View Saved Passwords
Technical Details
Connection Encrypted (TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256, 128 bit keys, TLS 1.2)
The page you are viewing was encrypted before being transmitted over the Internet.
Encryption makes it difficult for unauthorized people to view information traveling between computers. It is therefore unlikely that anyone read this page as it traveled across the network.
This website supplies publicly auditable Certificate Transparency records.

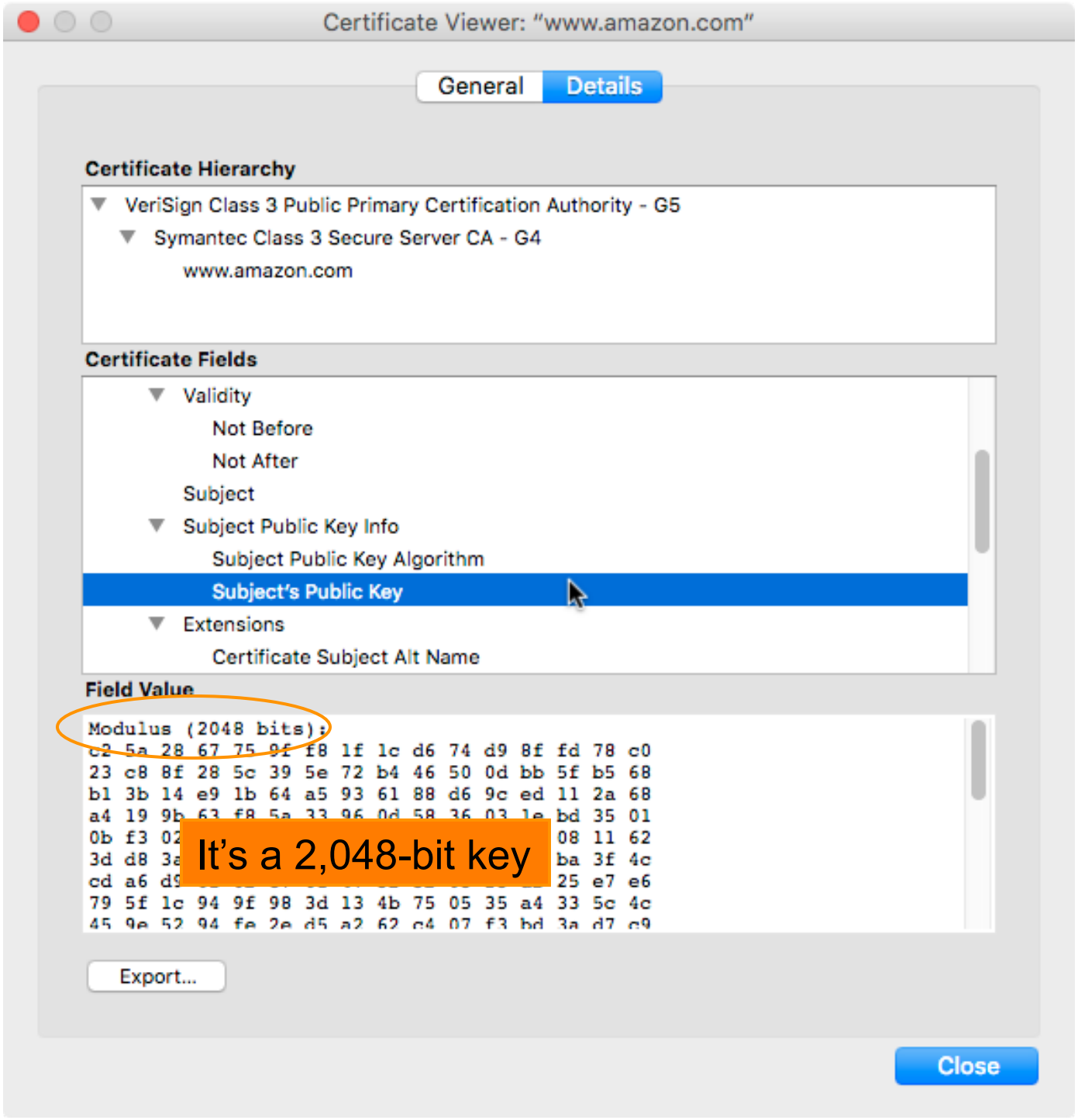

Page Info - https://www.amazon.com/
General
Permissions
Security
Website Identity
Website: www.amazon.com
Owner: This website does not supply ownership information.
Verified by: Symantec Corporation
View Certificate
Privacy & History
Have I visited this website prior to today? Yes, 29 times
Is this website storing information (cookies) on my computer? Yes
View Cookies
Have I saved any passwords for this website? No
View Saved Passwords
Here's the cipher suite used for the connection
Technical Details
Connection Encrypted (TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256, 128 bit keys, TLS 1.2)
The page you are viewing was encrypted before being transmitted over the Internet.
Encryption makes it difficult for unauthorized people to view information traveling between computers. It is therefore unlikely that anyone read this page as it traveled across the network.
This website supplies publicly auditable Certificate Transparency records.

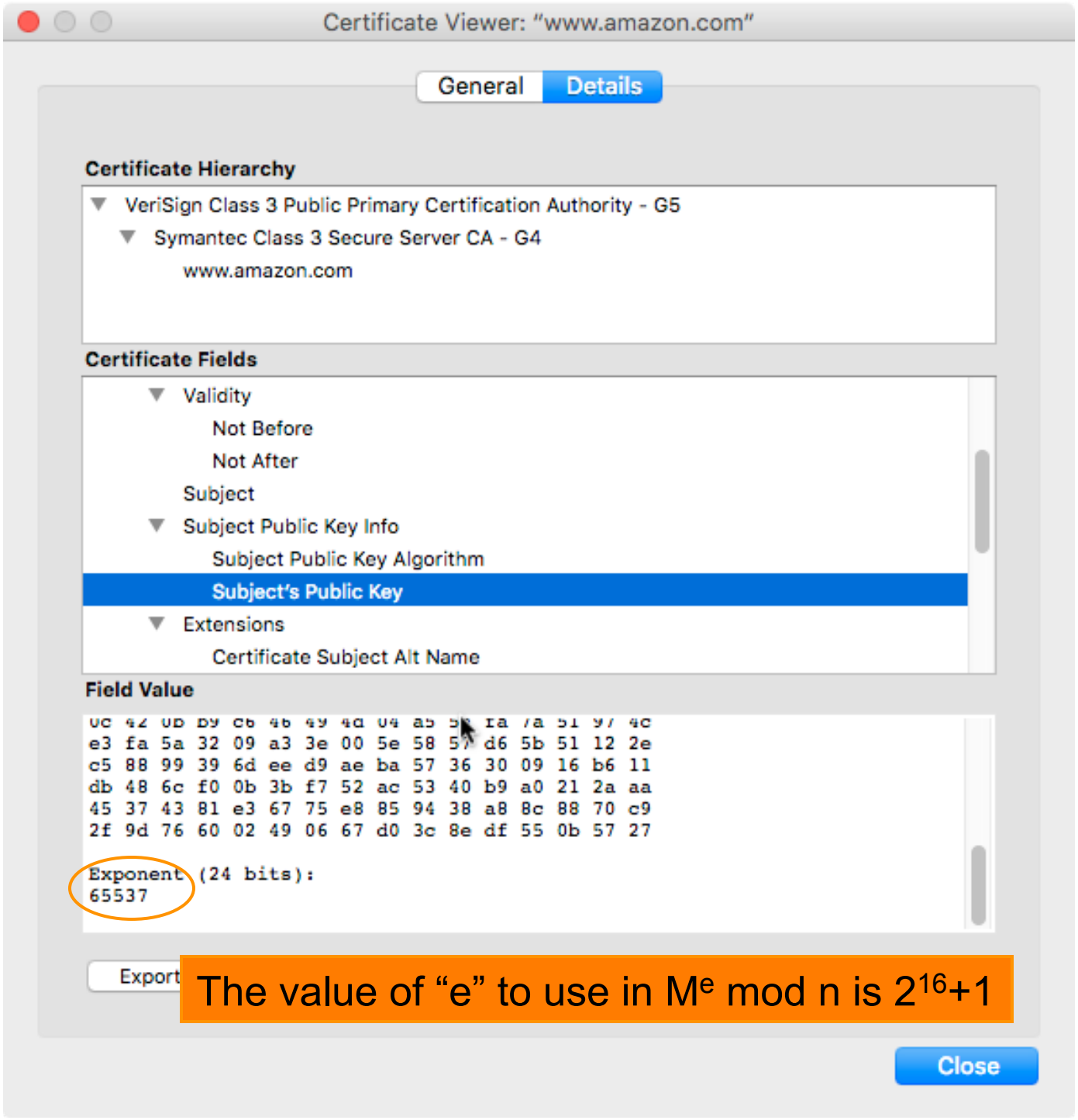

Certificate Viewer: "www.amazon.com"
General
Details
This certificate has been verified for the following uses:
SSL Client Certificate
SSL Server Certificate
Issued To
Common Name (CN) www.amazon.com
Organization (O) Amazon.com, Inc.
Organizational Unit (OU) <Not Part Of Certificate>
Serial Number 1D:4A:BD:AA:78:D0:9A:FE:79:9D:41:BC:EB:7A:76:62
Issued By
Common Name (CN) Symantec Class 3 Secure Server CA - G4
Organization (O) Symantec Corporation
Organizational Unit (OU) Symantec Trust Network
Period of Validity
Begins On October 30, 2016
Expires On December 31, 2017
Fingerprints
SHA-256 Fingerprint 6A:A0:AB:97:D0:F9:F1:50:58:96:31:3B:E2:37:2D:C3:
94:BD:42:77:57:F6:BD:B6:2D:DE:80:ED:54:D4:19:0D
SHA1 Fingerprint EF:14:6C:F1:5C:4A:F8:4D:BA:83:C2:1E:6C:5B:ED:C4:FA:34:1C:3E

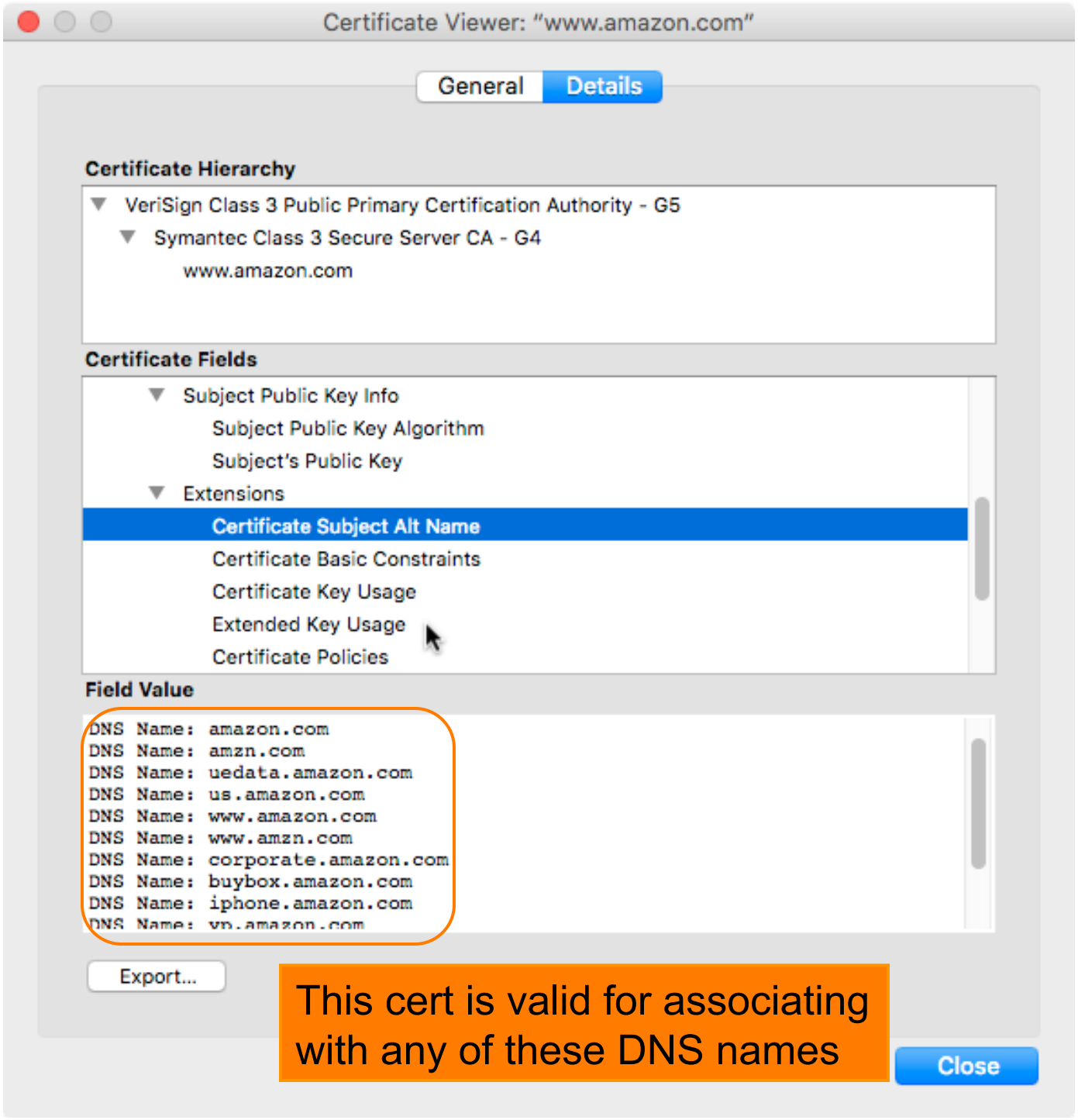

Certificate Viewer: "www.amazon.com"
General
Details
Certificate Hierarchy
VeriSign Class 3 Public Primary Certification Authority - G5
Symantec Class 3 Secure Server CA - G4
www.amazon.com
Certificate Fields
Issuer
Validity
Not Before
Not After
Subject
Subject Public Key Info
Subject Public Key Algorithm
Subject's Public Key
Extensions
Field Value
CN = www.amazon.com
O = "Amazon.com, Inc."
L = Seattle
ST = Washington
C = US
Export...
Close

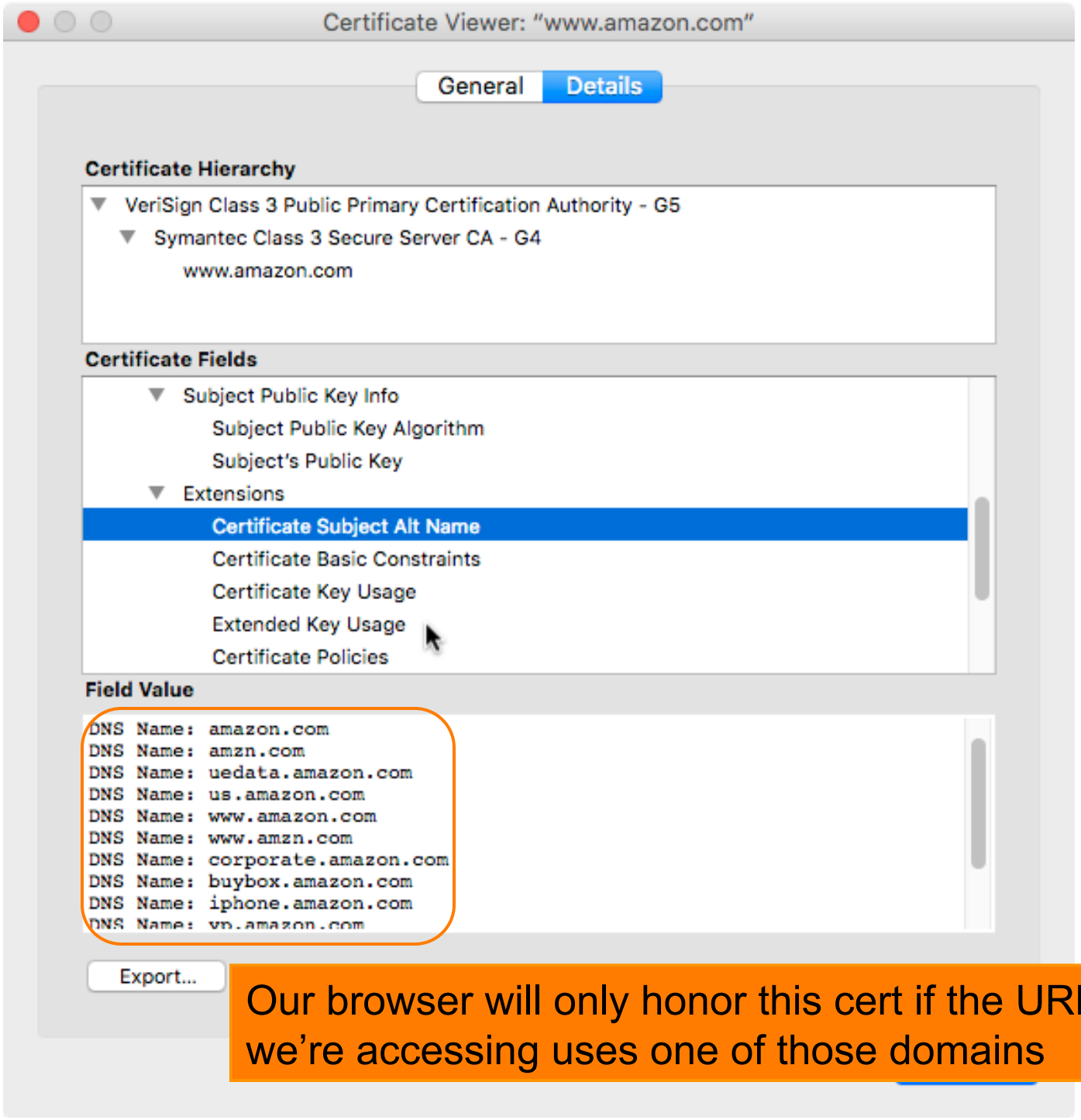


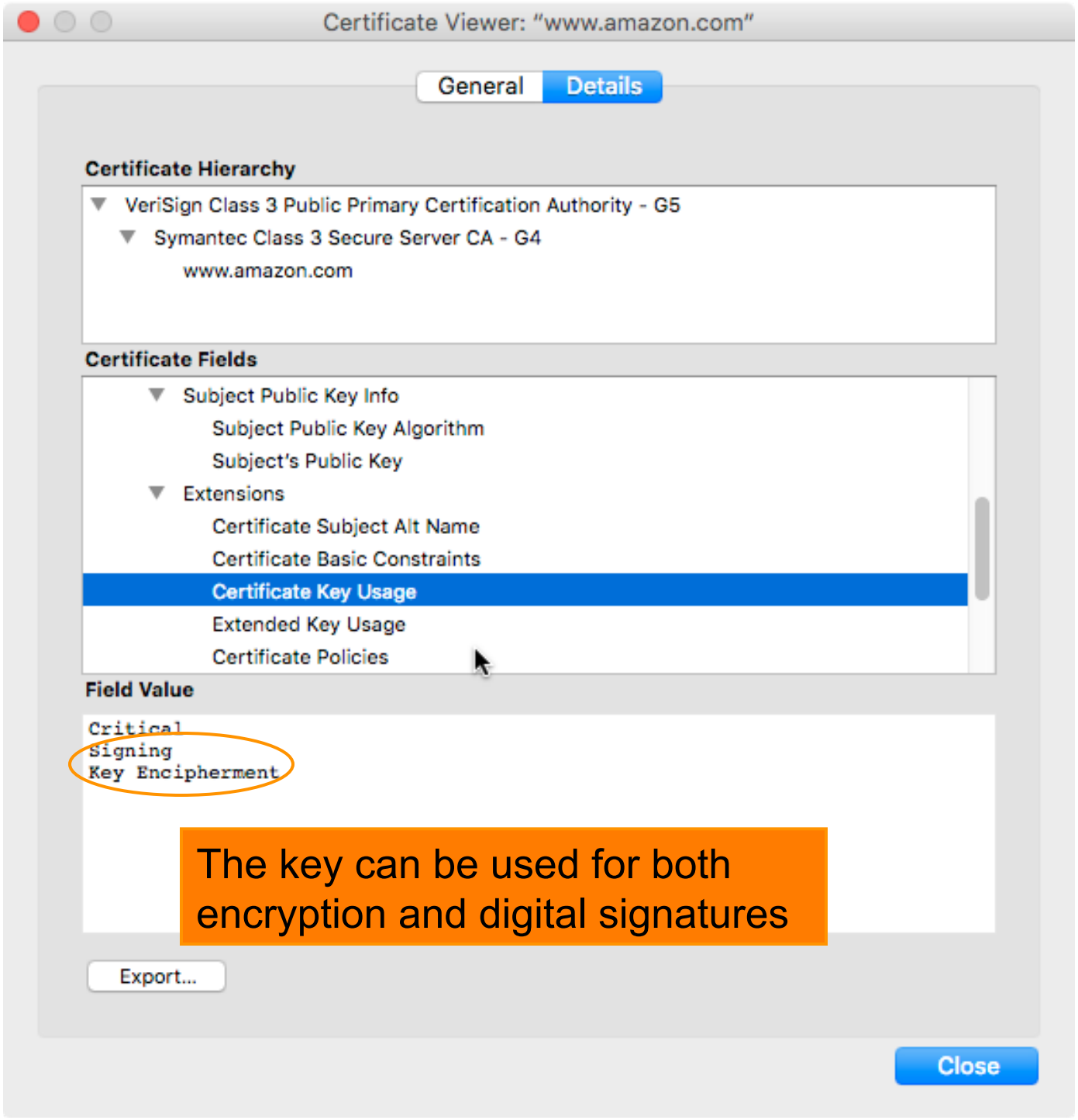
PKCS #1 = "Standard RSA encryption/signing" algorithms

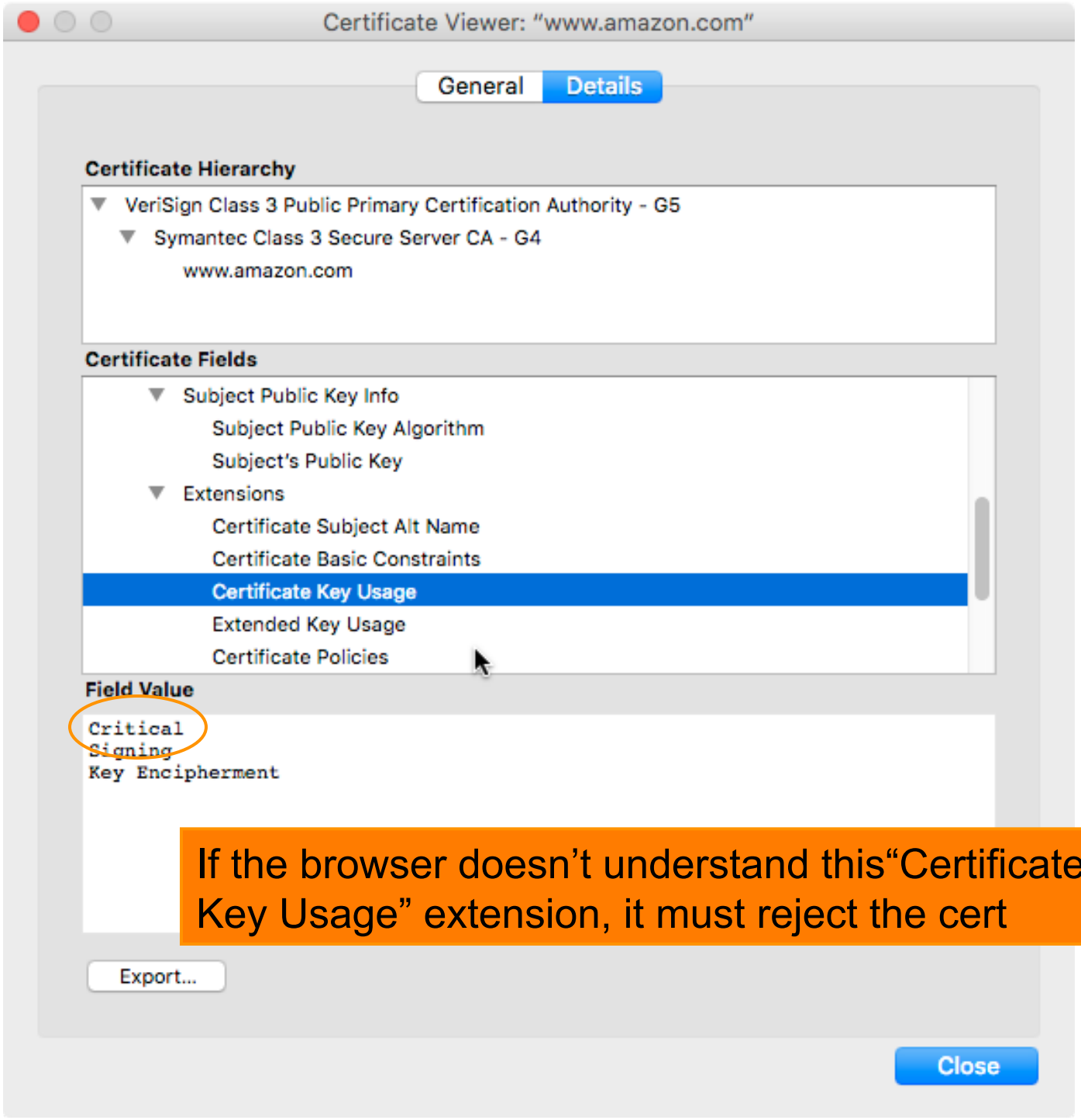

Certificate Viewer: "www.amazon.com"
General
Details
Certificate Hierarchy
VeriSign Class 3 Public Primary Certification Authority - G5
Symantec Class 3 Secure Server CA - G4
www.amazon.com
Certificate Fields
Validity
Not Before
Not After
Subject
Subject Public Key Info
Subject Public Key Algorithm
Subject's Public Key
Extensions
Certificate Subject Alt Name
Field Value
Modulus (2048 bits):
c2 5a 28 67 75 9f f8 1f 1c d6 74 d9 8f fd 78 c0
23 c8 8f 28 5c 39 5e 72 b4 46 50 0d bb 5f b5 68
b1 3b 14 e9 1b 64 a5 93 61 88 d6 9c ed 11 2a 68
a4 19 9b 63 f8 5a 33 96 0d 58 36 03 1e bd 35 01
0b f3 02 08 11 62
3d d8 3a ba 3f 4c
cd a6 d9 25 e7 e6
79 5f 1c 94 9f 98 3d 13 4b 75 05 35 a4 33 5c 4c
45 9e 52 94 fe 2e d5 a2 62 c4 07 f3 bd 3a d7 c9
It's a 2,048-bit key
Export...
Close


Certificate Viewer: "www.amazon.com"
General
Details
Certificate Hierarchy
VeriSign Class 3 Public Primary Certification Authority - G5
Symantec Class 3 Secure Server CA - G4
www.amazon.com
Certificate Fields
Validity
Not Before
Not After
Subject
Subject Public Key Info
Subject Public Key Algorithm
Subject's Public Key
Extensions
Certificate Subject Alt Name
Field Value
e3 fa 5a 32 09 a3 3e 00 5e 58 57 d6 5b 51 12 2e
c5 88 99 39 6d ee d9 ae ba 57 36 30 09 16 b6 11
db 48 6c f0 0b 3b f7 52 ac 53 40 b9 a0 21 2a aa
45 37 43 81 e3 67 75 e8 85 94 38 a8 8c 88 70 c9
2f 9d 76 60 02 49 06 67 d0 3c 8e df 55 0b 57 27
Exponent (24 bits):
65537
Export
The value of "e" to use in Me mod n is 2^16+1
Close

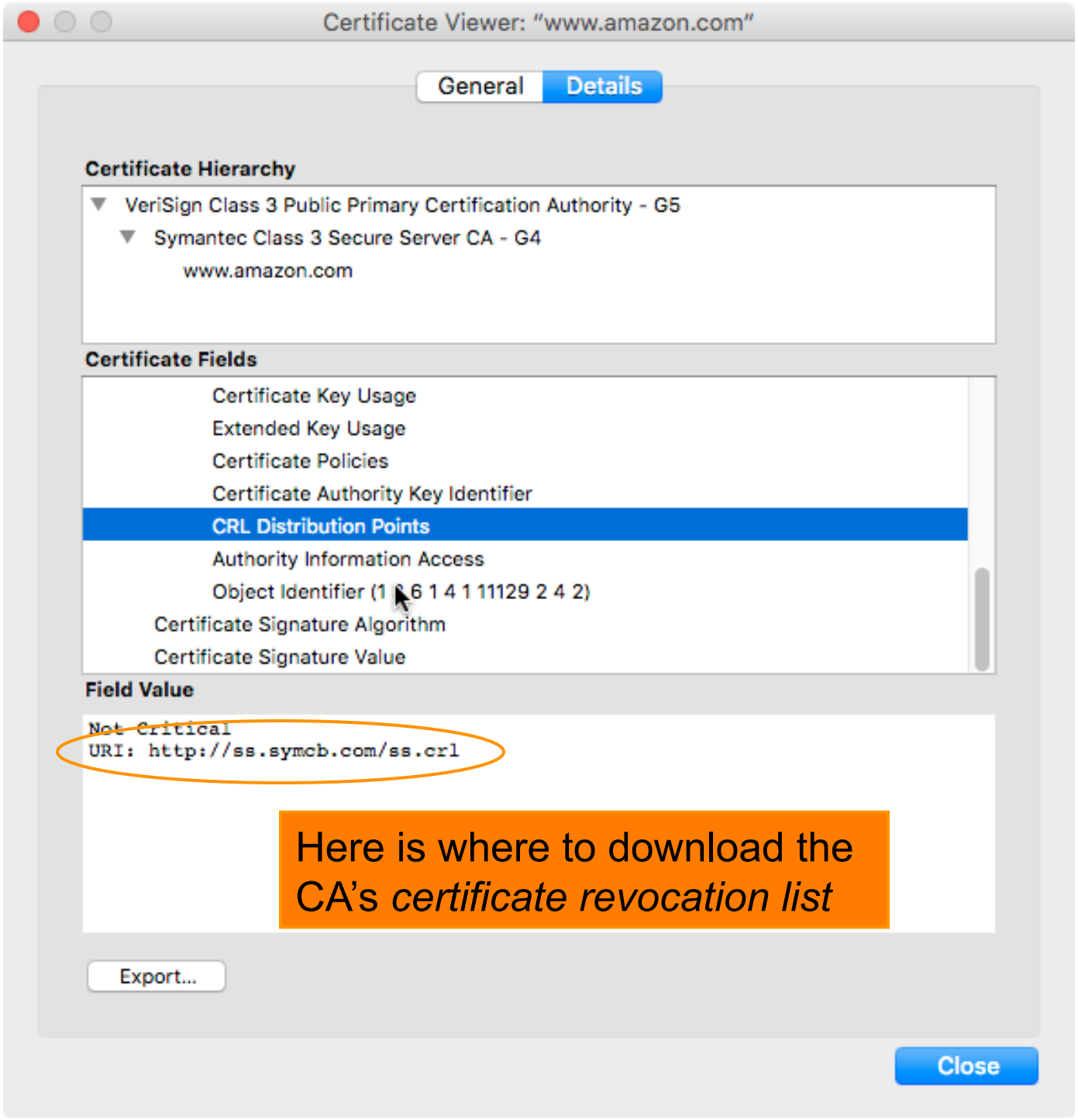

Certificate Viewer: "www.amazon.com"
General
Details
Certificate Hierarchy
VeriSign Class 3 Public Primary Certification Authority - G5
Symantec Class 3 Secure Server CA - G4
www.amazon.com
Certificate Fields
Subject Public Key Info
Subject Public Key Algorithm
Subject's Public Key
Extensions
Certificate Subject Alt Name
Certificate Basic Constraints
Certificate Key Usage
Extended Key Usage
Certificate Policies
Field Value
DNS Name: amazon.com
DNS Name: amzn.com
DNS Name: uedata.amazon.com
DNS Name: us.amazon.com
DNS Name: www.amazon.com
DNS Name: www.amzn.com
DNS Name: corporate.amazon.com
DNS Name: buybox.amazon.com
DNS Name: iphone.amazon.com
Export...
This cert is valid for associating with any of these DNS names
Close

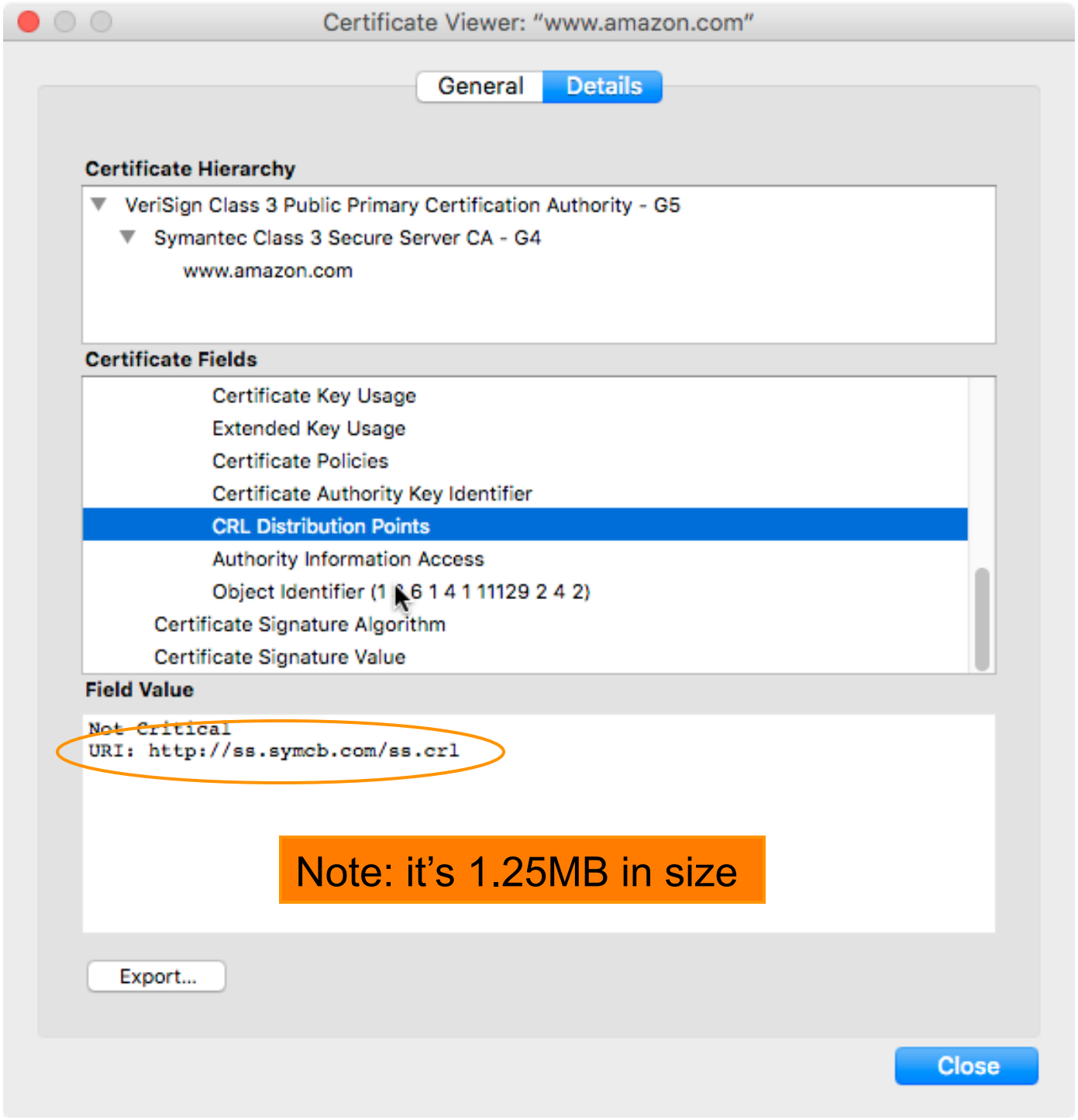

Certificate Viewer: "www.amazon.com"
General
Details
Certificate Hierarchy
VeriSign Class 3 Public Primary Certification Authority - G5
Symantec Class 3 Secure Server CA - G4
www.amazon.com
Certificate Fields
Subject Public Key Info
Subject Public Key Algorithm
Subject's Public Key
Extensions
Certificate Subject Alt Name
Certificate Basic Constraints
Certificate Key Usage
Extended Key Usage
Certificate Policies
Field Value
DNS Name: amazon.com
DNS Name: amzn.com
DNS Name: uedata.amazon.com
DNS Name: us.amazon.com
DNS Name: www.amazon.com
DNS Name: www.amzn.com
DNS Name: corporate.amazon.com
DNS Name: buybox.amazon.com
DNS Name: iphone.amazon.com
Export...
Our browser will only honor this cert if the URL we're accessing uses one of those domains

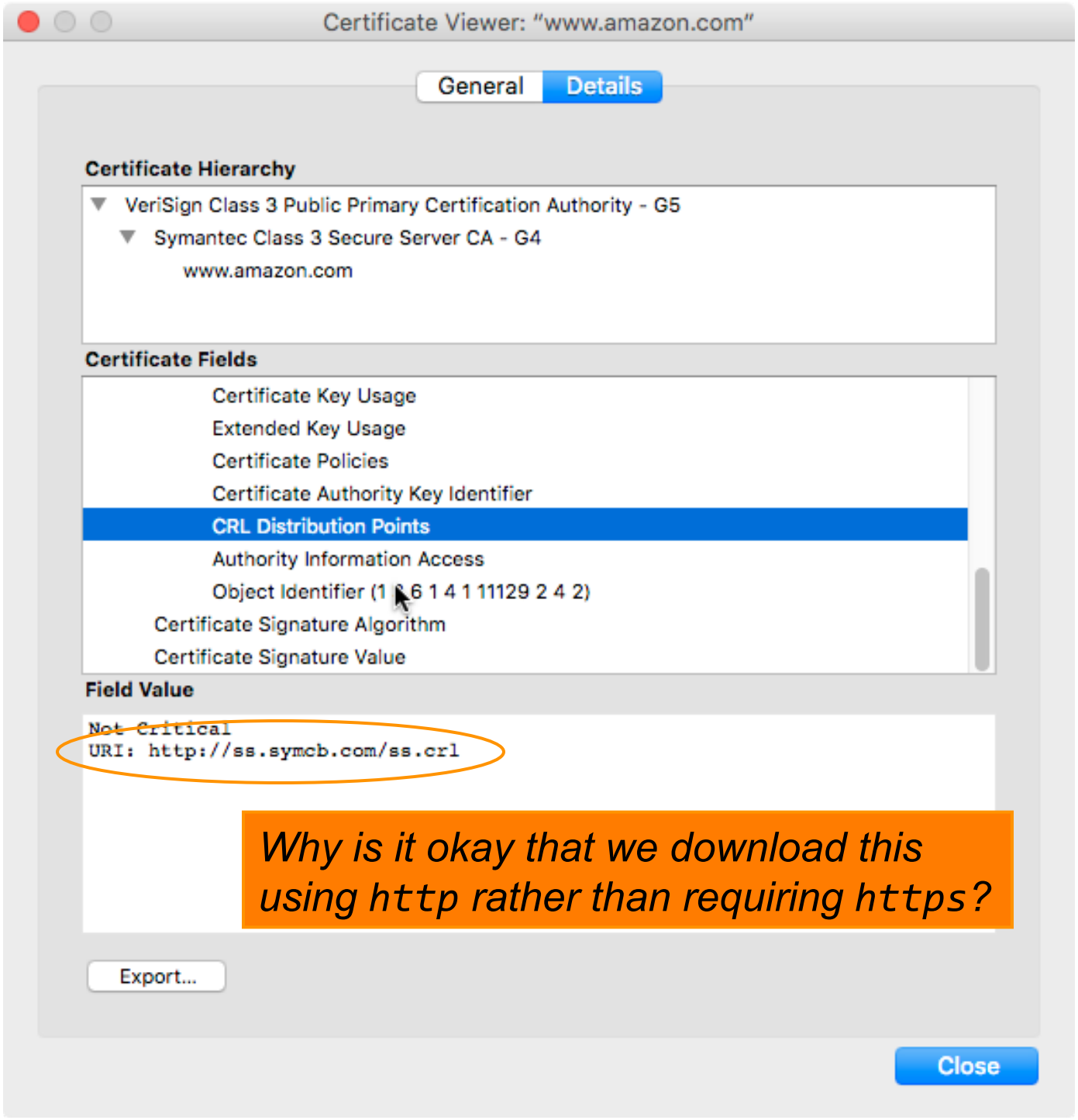

Certificate Viewer: "www.amazon.com"
General
Details
Certificate Hierarchy
VeriSign Class 3 Public Primary Certification Authority - G5
Symantec Class 3 Secure Server CA - G4
www.amazon.com
Certificate Fields
Subject Public Key Info
Subject Public Key Algorithm
Subject's Public Key
Extensions
Certificate Subject Alt Name
Certificate Basic Constraints
Certificate Key Usage
Extended Key Usage
Certificate Policies
Field Value
Critical
Signing
Key Encipherment
The key can be used for both encryption and digital signatures
Export...
Close

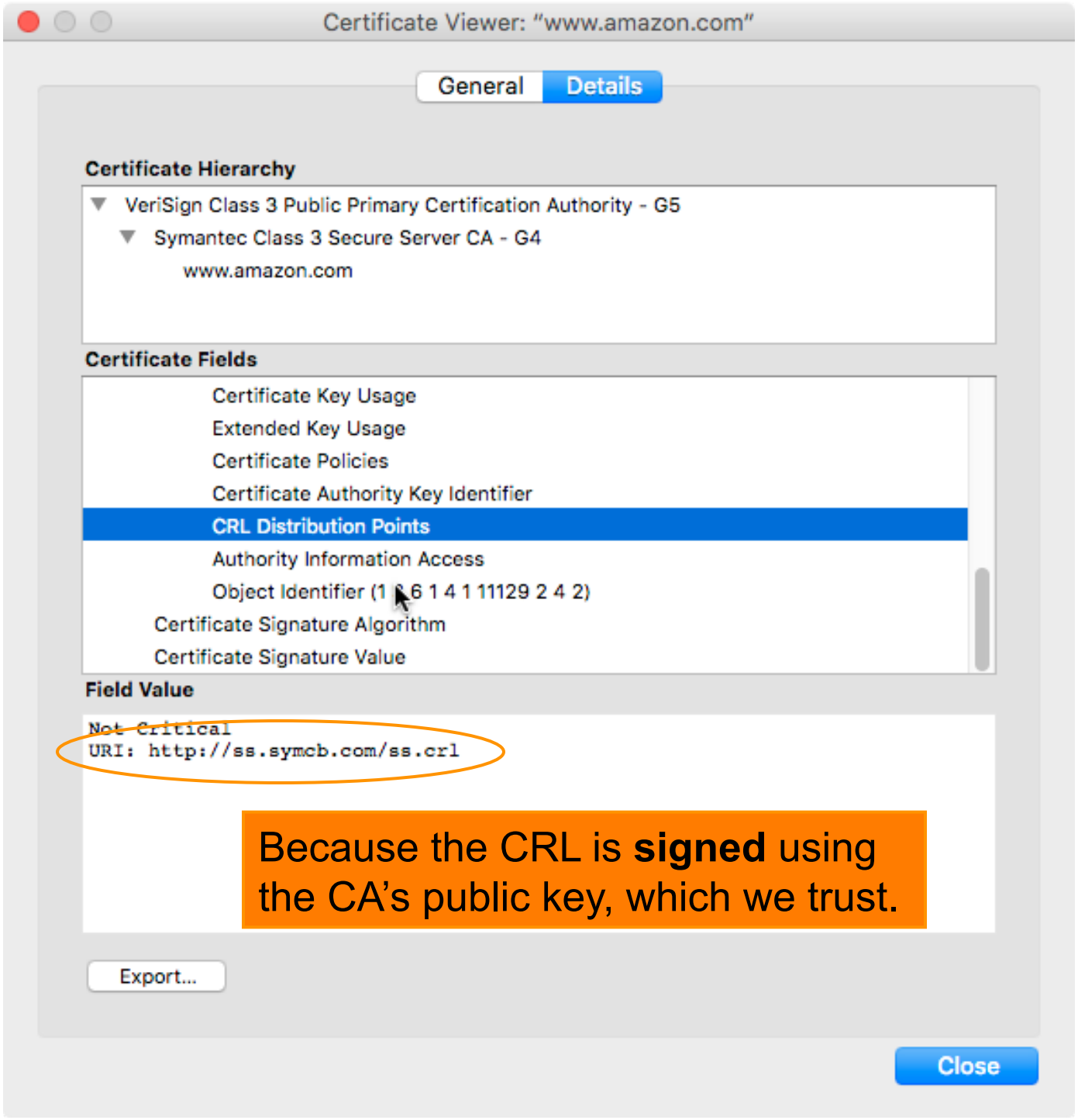


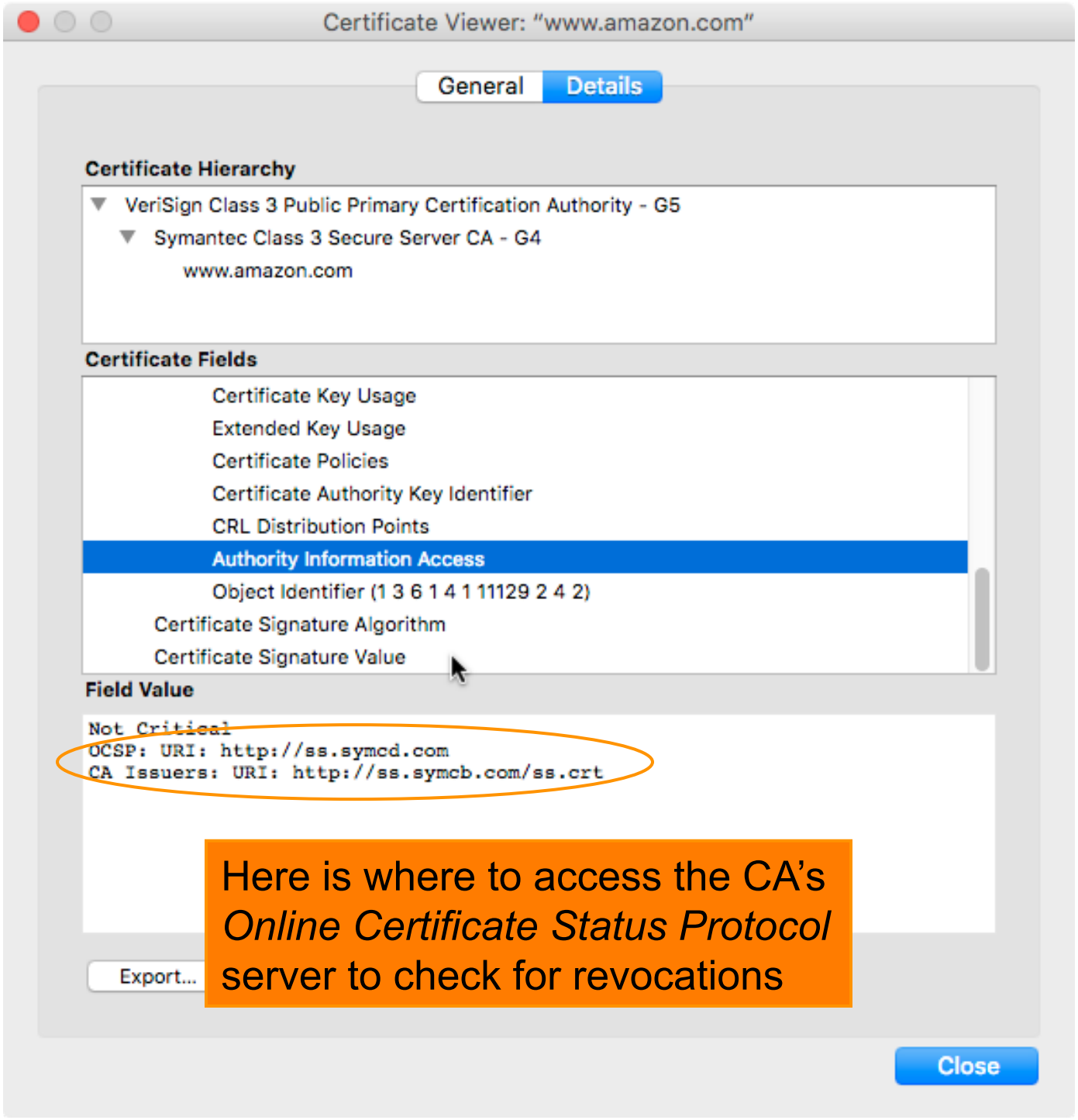
If the browser doesn't understand this"Certificate Key Usage" extension, it must reject the cert

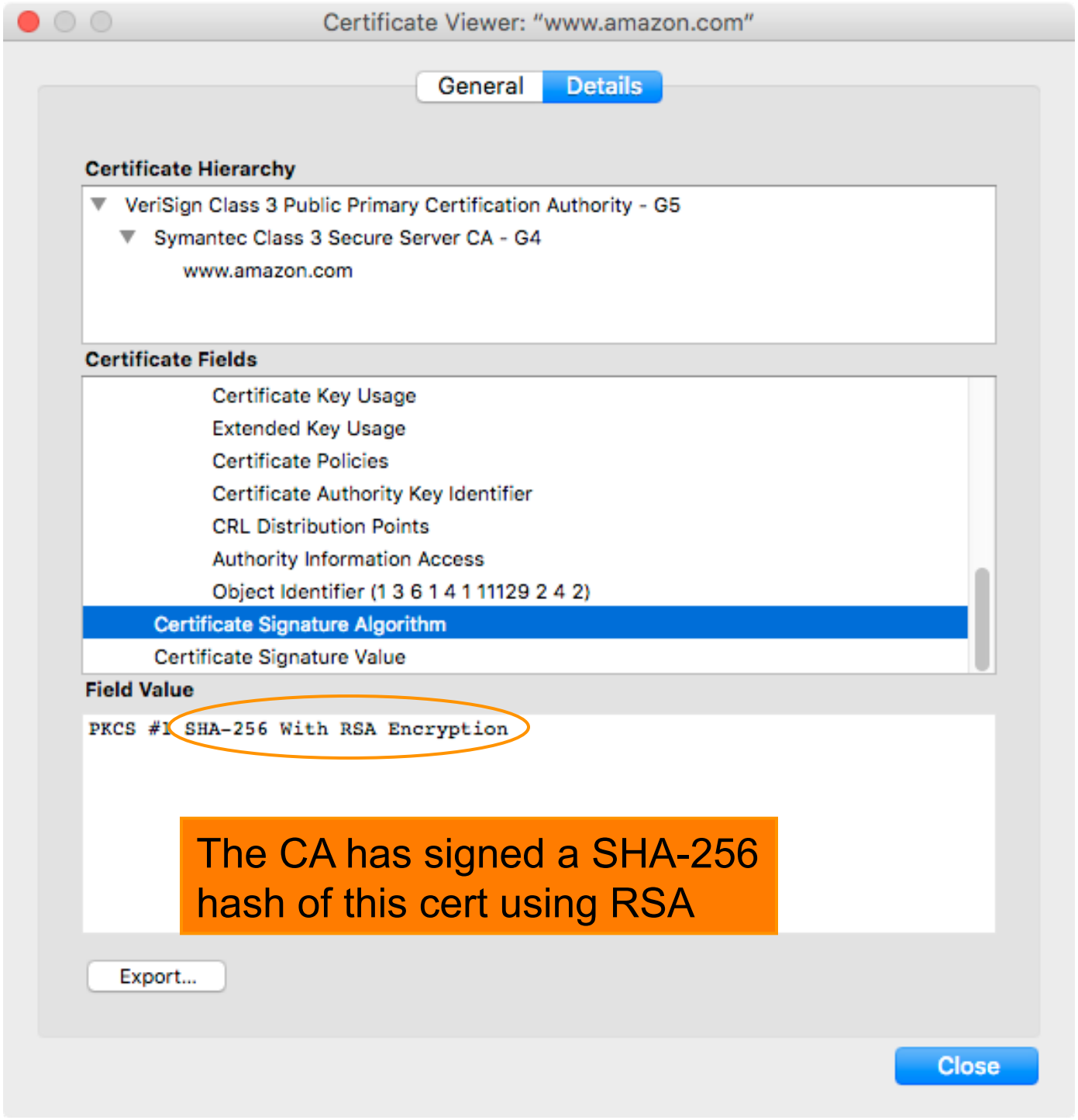

Certificate Viewer: "www.amazon.com"
General
Details
Certificate Hierarchy
VeriSign Class 3 Public Primary Certification Authority - G5
Symantec Class 3 Secure Server CA - G4
www.amazon.com
Certificate Fields
Certificate Key Usage
Extended Key Usage
Certificate Policies
Certificate Authority Key Identifier
CRL Distribution Points
Authority Information Access
Object Identifier (1 3 6 1 4 1 11129 2 4 2)
Certificate Signature Algorithm
Certificate Signature Value
Field Value
Not Critical
URI: http://ss.symcb.com/ss.crl
Here is where to download the CA's certificate revocation list
Export...
Close

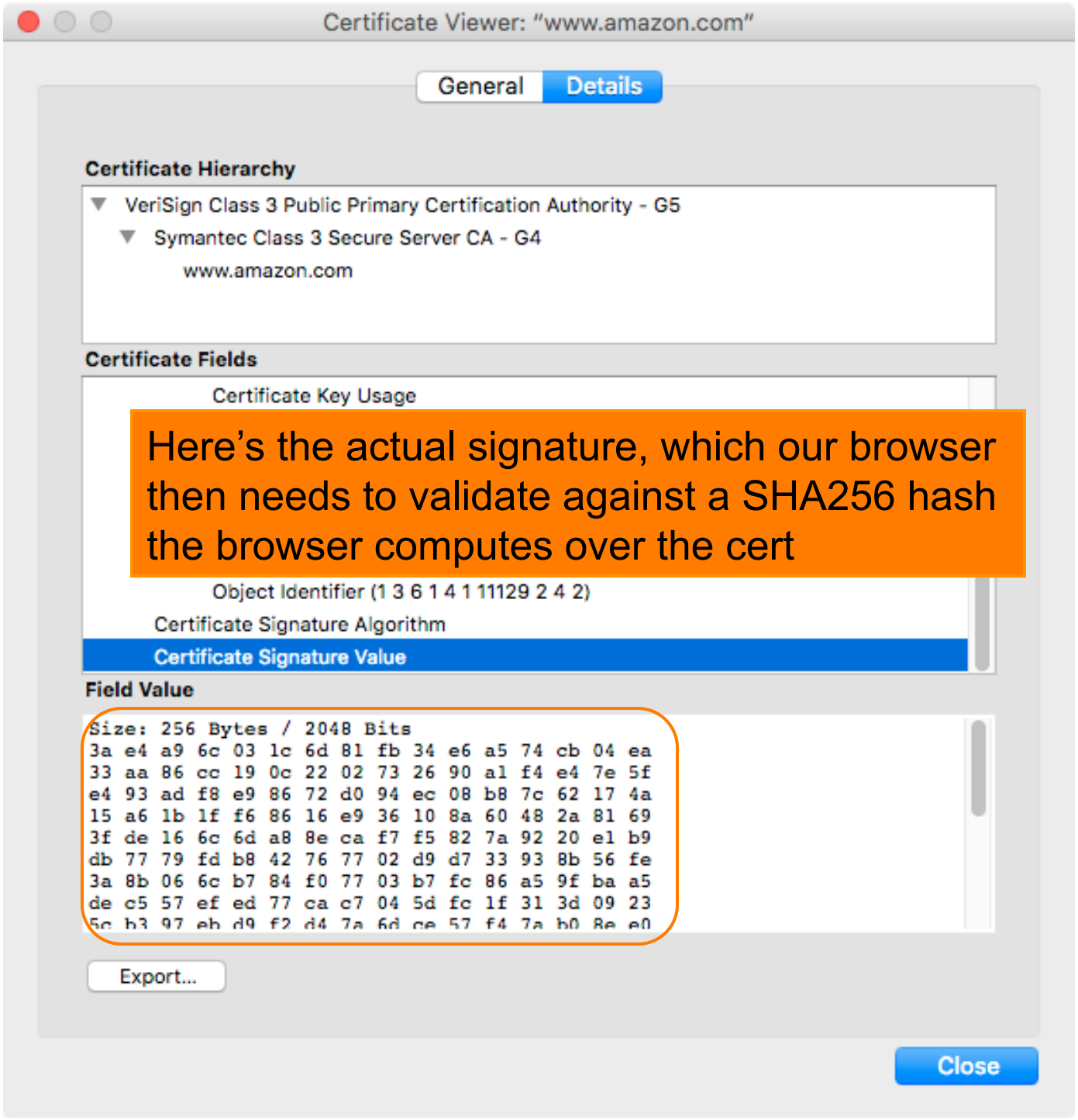

Certificate Viewer: "www.amazon.com"
General
Details
Certificate Hierarchy
VeriSign Class 3 Public Primary Certification Authority - G5
Symantec Class 3 Secure Server CA - G4
www.amazon.com
Certificate Fields
Certificate Key Usage
Extended Key Usage
Certificate Policies
Certificate Authority Key Identifier
CRL Distribution Points
Authority Information Access
Object Identifier (1 3 6 1 4 1 11129 2 4 2)
Certificate Signature Algorithm
Certificate Signature Value
Field Value
Not Critical
URI: http://ss.symcb.com/ss.crl
Note: it's 1.25MB in size
Export...
Close


Certificate Viewer: "www.amazon.com"
General
Details
Certificate Hierarchy
VeriSign Class 3 Public Primary Certification Authority - G5
Symantec Class 3 Secure Server CA - G4
www.amazon.com
Certificate Fields
Certificate Key Usage
Extended Key Usage
Certificate Policies
Certificate Authority Key Identifier
CRL Distribution Points
Authority Information Access
Object Identifier (1 3 6 1 4 1 11129 2 4 2)
Certificate Signature Algorithm
Certificate Signature Value
Field Value
Not Critical
URI: http://ss.symcb.com/ss.crl
Why is it okay that we download this using http rather than requiring https?
Export...
Close


Certificate Viewer: "www.amazon.com"
General
Details
Certificate Hierarchy
VeriSign Class 3 Public Primary Certification Authority - G5
Symantec Class 3 Secure Server CA - G4
www.amazon.com
Certificate Fields
Certificate Key Usage
Extended Key Usage
Certificate Policies
Certificate Authority Key Identifier
CRL Distribution Points
Authority Information Access
Object Identifier (1 3 6 1 4 1 11129 2 4 2)
Certificate Signature Algorithm
Certificate Signature Value
Field Value
Not Critical
URI: http://ss.symcb.com/ss.crl
Because the CRL is signed using the CA's public key, which we trust.
Export...
Close


Certificate Viewer: "www.amazon.com"
General
Details
Certificate Hierarchy
VeriSign Class 3 Public Primary Certification Authority - G5
Symantec Class 3 Secure Server CA - G4
www.amazon.com
Certificate Fields
Certificate Key Usage
Extended Key Usage
Certificate Policies
Certificate Authority Key Identifier
CRL Distribution Points
Authority Information Access
Object Identifier (1 3 6 1 4 1 11129 2 4 2)
Certificate Signature Algorithm
Certificate Signature Value
Field Value
Not Critical
OCSP: URI: http://ss.symcd.com
CA Issuers: URI: http://ss.symcb.com/ss.crt
Here is where to access the CA's Online Certificate Status Protocol server to check for revocations
Export...
Close


Certificate Viewer: "www.amazon.com"
General
Details
Certificate Hierarchy
VeriSign Class 3 Public Primary Certification Authority - G5
Symantec Class 3 Secure Server CA - G4
www.amazon.com
Certificate Fields
Certificate Key Usage
Extended Key Usage
Certificate Policies
Certificate Authority Key Identifier
CRL Distribution Points
Authority Information Access
Object Identifier (1 3 6 1 4 1 11129 2 4 2)
Certificate Signature Algorithm
Certificate Signature Value
Field Value
PKCS #1 SHA-256 With RSA Encryption
The CA has signed a SHA-256 hash of this cert using RSA
Export...
Close


Certificate Viewer: "www.amazon.com"
General
Details
Certificate Hierarchy
VeriSign Class 3 Public Primary Certification Authority - G5
Symantec Class 3 Secure Server CA - G4
www.amazon.com
Certificate Fields
Certificate Key Usage
Here's the actual signature, which our browser then needs to validate against a SHA256 hash the browser computes over the cert
Object Identifier (1 3 6 1 4 1 11129 2 4 2)
Certificate Signature Algorithm
Certificate Signature Value
Field Value
Size: 256 Bytes / 2048 Bits
3a e4 a9 6c 03 1c 6d 81 fb 34 e6 a5 74 cb 04 ea
33 aa 86 cc 19 0c 22 02 73 26 90 a1 f4 e4 7e 5f
e4 93 ad f8 e9 86 72 d0 94 ec 08 b8 7c 62 17 4a
15 a6 1b 1f f6 86 16 e9 36 10 8a 60 48 2a 81 69
3f de 16 6c 6d a8 8e ca f7 f5 82 7a 92 20 e1 b9
db 77 79 fd b8 42 76 77 02 d9 d7 33 93 8b 56 fe
3a 8b 06 6c b7 84 f0 77 03 b7 fc 86 a5 9f ba a5
de c5 57 ef ed 77 ca c7 04 5d fc 1f 31 3d 09 23
Export...
Close

Validating Amazon's Identity

- Browser compares domain *name* in cert w/ URL
  - Note: this provides an end-to-end property (as opposed to say a cert associated with an IP address)
- Browser accesses *separate* cert belonging to **issuer**
  - These are hardwired into the browser - **trusted!**
  - There could be a *chain* of these …
- Browser applies issuer's public key to verify signature, obtaining hash of what issuer signed
  - Compares with its own SHA-256 hash of Amazon's cert
- Assuming hashes match, now have high confidence it's indeed Amazon …
  - ***assuming signatory is trustworthy***

= assuming didn't lose private key; assuming didn't sign thoughtlessly

End-to-End ⇒ Powerful Protections

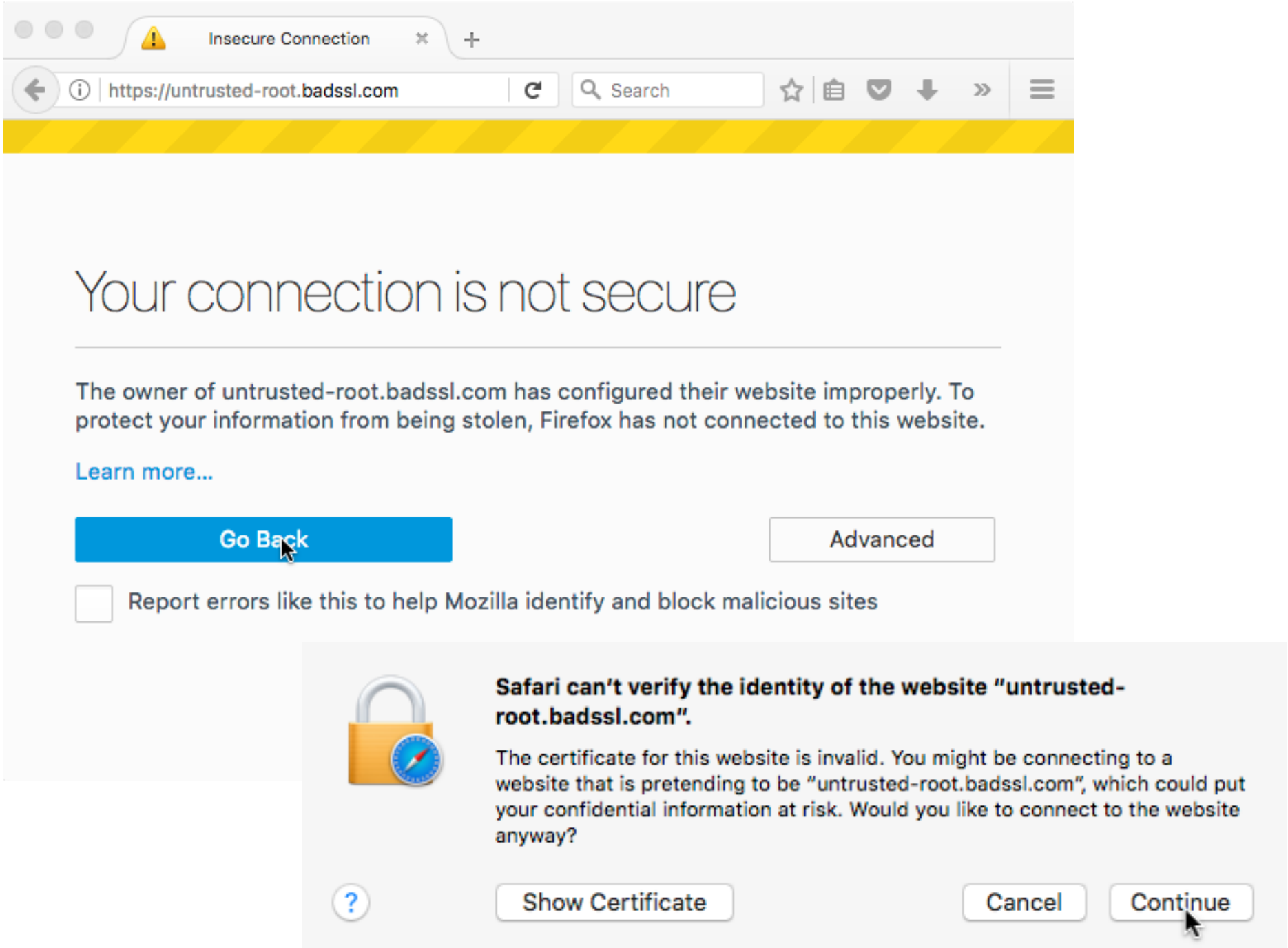
- Attacker runs a sniffer to capture our WiFi session?
  - (maybe by buying a cup of coffee to get the password)
  - But: encrypted communication is unreadable
    - No problem!
- DNS cache poisoning?
  - Client goes to wrong server
  - But: detects impersonation since attacker lacks valid cert
    - No problem!
- Attacker hijacks our connection, injects new traffic
  - But: data receiver rejects it due to failed integrity check
    - No problem!

Powerful Protections, con't

- DHCP spoofing?
  - Client goes to wrong server
  - But: detects impersonation since attacker lacks valid cert
    - No problem!
- Attacker manipulates routing to run us by an eavesdropper or take us to the wrong server?
  - But: they can't read; we detect impersonation
    - No problem!
- Attacker slips in as a Man In The Middle?
  - But: they can't read, they can't inject
  - They can't even replay previous encrypted traffic
  - **No problem!**

Validating Amazon's Identity, con't

- Browser accesses *separate* cert belonging to **issuer**
  - These are hardwired into the browser - **trusted!**
- What if browser can't find a cert for the issuer?


Insecure Connection
https://untrusted-root.badssl.com
Search
Your connection is not secure
The owner of untrusted-root.badssl.com has configured their website improperly. To protect your information from being stolen, Firefox has not connected to this website.
Learn more...
Go Back
Advanced
Report errors like this to help Mozilla identify and block malicious sites
Safari can't verify the identity of the website "untrusted-root.badssl.com".
The certificate for this website is invalid. You might be connecting to a website that is pretending to be "untrusted-root.badssl.com", which could put your confidential information at risk. Would you like to connect to the website anyway?
?
Show Certificate
Cancel
Continue

Validating Amazon's Identity, con't

- Browser accesses *separate* cert belonging to **issuer**
  - These are hardwired into the browser - **trusted!**
- What if browser can't find a cert for the issuer?
- If it can't find the cert, then warns the user that site has not been verified
  - Note, can still proceed, just without authentication
- Q: Which end-to-end security properties do we lose if we incorrectly trust that the site is whom we think?
- A: **All of them!**
  - Goodbye confidentiality, integrity, authentication
  - Attacker can read everything, modify, impersonate

SSL / TLS Limitations

- Properly used, SSL / TLS provides powerful end-to-end protections
- So why not use it for *everything*??
- Issues:
  - Cost of public-key crypto
    - Takes non-trivial CPU processing (but today a minor issue)
    - Note: *symmetric* key crypto on modern hardware is non-issue
  - Hassle of buying/maintaining certs (fairly minor)


https://letsencrypt.org
let's encrypt
LINUX FOUNDATION COLLABORATIVE PROJECTS
Let's Encrypt
Documentation
Get Help
Donate
About Us
Let's Encrypt is a free, automated, and open Certificate Authority.
Get Started
Donate
FROM OUR BLOG
Mar 23, 2017
OVH Renews Platinum Sponsorship of Let's Encrypt
We're pleased to announce that OVH has renewed their support for Let's Encrypt as a Platinum sponsor for the next three years.
Read more
MAJOR SPONSORS
mozilla
Akamai
cisco
EFF
OVH
chrome
IdenTrust
gemalto
FORD FOUNDATION
facebook
AUTOMATTIC
shopify


https://letsencrypt.org
let's encrypt
LINUX FOUNDATION COLLABORATIVE PROJECTS
Let's Encrypt
Documentation
Get Help
Donate
About Us
Let's Encrypt is a free, automated, and open Certificate Authority.
You prove to this CA that you're entitled to a cert for foo.com by demonstrating your control over the domain.
The CA issues a challenge, one of:
1. Add an (invisible) item to the foo.com homepage
2. Add an entry to the foo.com DNS zone
3. Show you can receive email at the registered foo.com email address
support for Let's Encrypt as a Platinum sponsor for the next three years.
Read more
FORD FOUNDATION
facebook
AUTOMATTIC
shopify

SSL / TLS Limitations

- Properly used, SSL / TLS provides powerful end-to-end protections
- So why not use it for *everything*??
- Issues:
  - Cost of public-key crypto
    - Takes non-trivial CPU processing (but today a minor issue)
    - Note: *symmetric* key crypto on modern hardware is non-issue
  - Hassle of buying/maintaining certs (fairly minor)
  - DoS amplification
    - Client can force server to undertake public key operations
    - But: requires established TCP connection, and given that, there are often other juicy targets like back-end databases
  - Integrating with other sites that don't use HTTPS
  - **Latency**: extra round trips ⇒ pages take longer to load