CSE467: Computer Security

18. Protocol Security: SSL/TLS & HTTPS

Seongil Wi

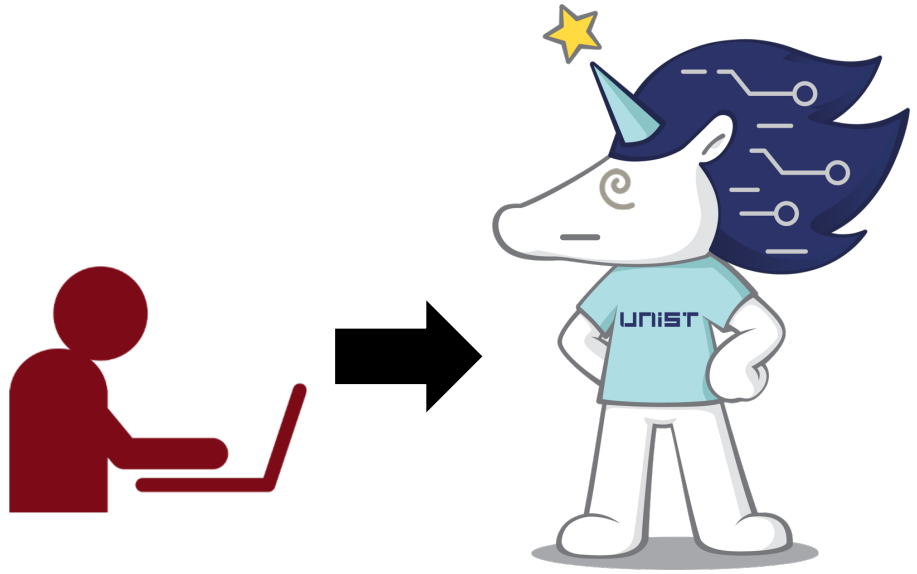
Department of Computer Science and Engineering

Activity #1: SaveUNIST

- **If you find unknown security problems on campus, report them to me!**

- Depending on the severity, bonus points will be given
  - E.g., A+ → 1 letter grade up → 10% score up → ... → 1 drink → ...

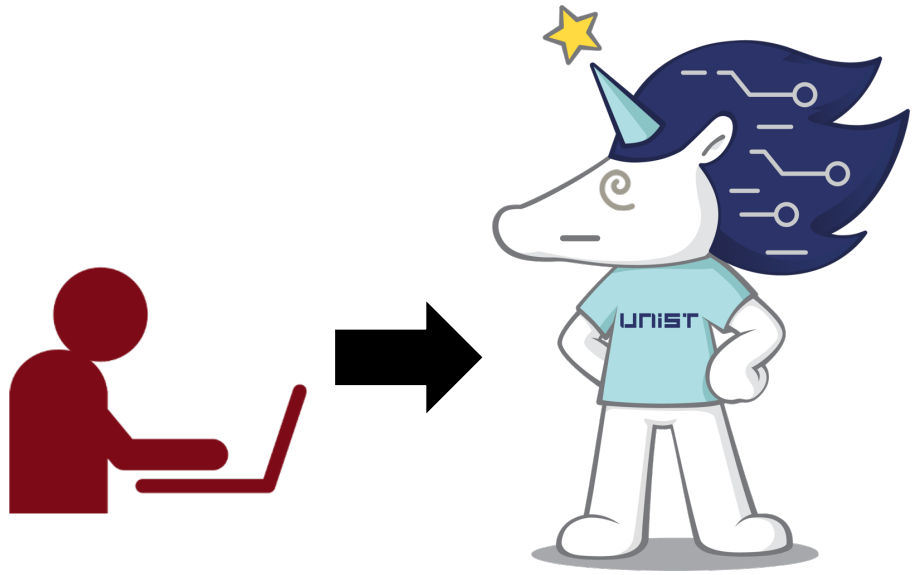
- **(IMPORTANT!) DO NOT** try anything illegal
  - If you cannot decide by yourself, discuss it with us first!



Activity #1: SaveUNIST

- **Period (confirmed)**: **Nov 20 ~ Dec 8**
- **Time: 9:00 ~ 18:00**
- **Target**: UNIST IST hompage (https://ist.unist.ac.kr/), use a VPN to access from outside

Activity should only be done during this period!



Activity #1: SaveUNIST

보 안 서 약 서

본인은 2023 년 11 월 6 일 부터 2023 년 11 월 17 일 까지 울산과학기술원 컴퓨터 공학과의 컴퓨터보안 과목에서 과제를 수행함에 있어, 다음과 같이 1)울산과기원의 정보보안 규정과 통제절차 및 2)상급기관 및 유관기관 규정의 보안사항을 준수할 것을 엄숙히 서약 합니다.

1. **나는 상기한 과제를 수행하며, 습득한 IT 정보와 보안관련 사항 등의 기관정보와 관련된 어떠한 정보도 외부 및 타인에게 일체 발설하거나 노출 또는 반출하지 않을 것을 서약한다.**
   가. 네트워크/시스템/보안 등 IT인프라 관련 구성 일체
   나. 시스템 접근권한 정보 일체
   다. 기관 내에 있는 모든 개인정보 일체
   라. DB정보 일체
   마. 정보시스템 취약점 등의 정보 일체
   바. 기타 기관 정보보안관련 비밀, 대외비, 누출금지 대상 일체

2. **나는 위의 사항을 위반했거나 기밀을 누설한 경우, 아래의 관계 법규 및 규정에 따라 엄중한 처벌을 받을 것을 서약한다.**
   가.「형법」 및「개인정보보호법」
   나.「정보통신망 이용촉진 및 정보
   다.「울산과학기술원 학칙」
   라.「울산과학기술원 학생 징계 지

3. 서약자 연명부

A pledge not to do anything illegal

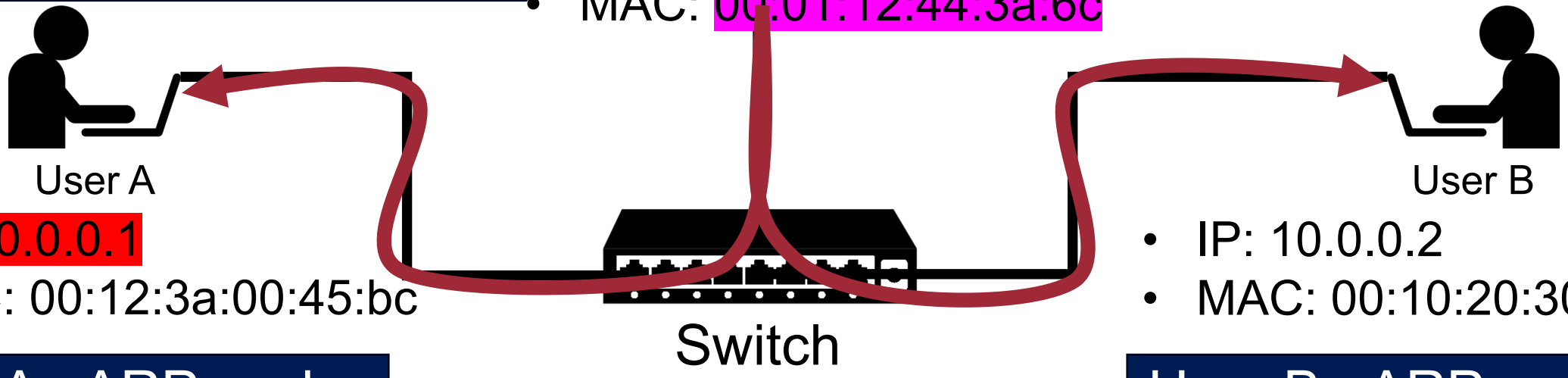
구분 (idx)	소속 (department)	연락처 (email address)	성명 (name)	서명 (sign)	교수 서명 Prof's sign
1					
2					
3					
4					
5					
6					
7					
8					
9					
10					

Recap: ARP Spoofing

ARP response
My IP Addr: 10.0.0.2
My MAC addr: 00:01:12:44:3a:6c
Destination: User A

ARP response
My IP Addr: 10.0.0.1
My MAC addr: 00:01:12:44:3a:6c
Destination: User B

- IP: 10.0.0.3
- MAC: 00:01:12:44:3a:6c



- IP: 10.0.0.1
- MAC: 00:12:3a:00:45:bc

- IP: 10.0.0.2
- MAC: 00:10:20:30:ac:06

User A - ARP cache

IP Addr	Mac Addr
10.0.0.2	~~00:10:20:30:ac:06~~ 00:01:12:44:3a:6c

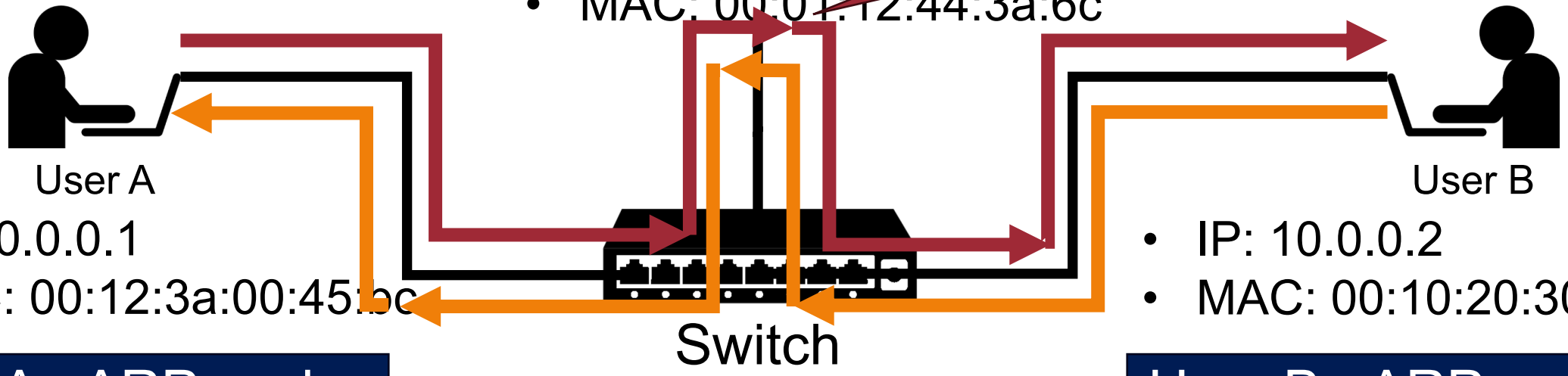
User B - ARP cache

IP Addr	Mac Addr
10.0.0.1	~~00:12:3a:00:45:bc~~ 00:01:12:44:3a:6c

Recap: ARP Spoofing

Man-in-the-Middle (MITM) attack

- IP: 10.0.0.3
- MAC: 00:01:12:44:3a:6c



- IP: 10.0.0.1
- MAC: 00:12:3a:00:45:bc

- IP: 10.0.0.2
- MAC: 00:10:20:30:ac:06

User A - ARP cache

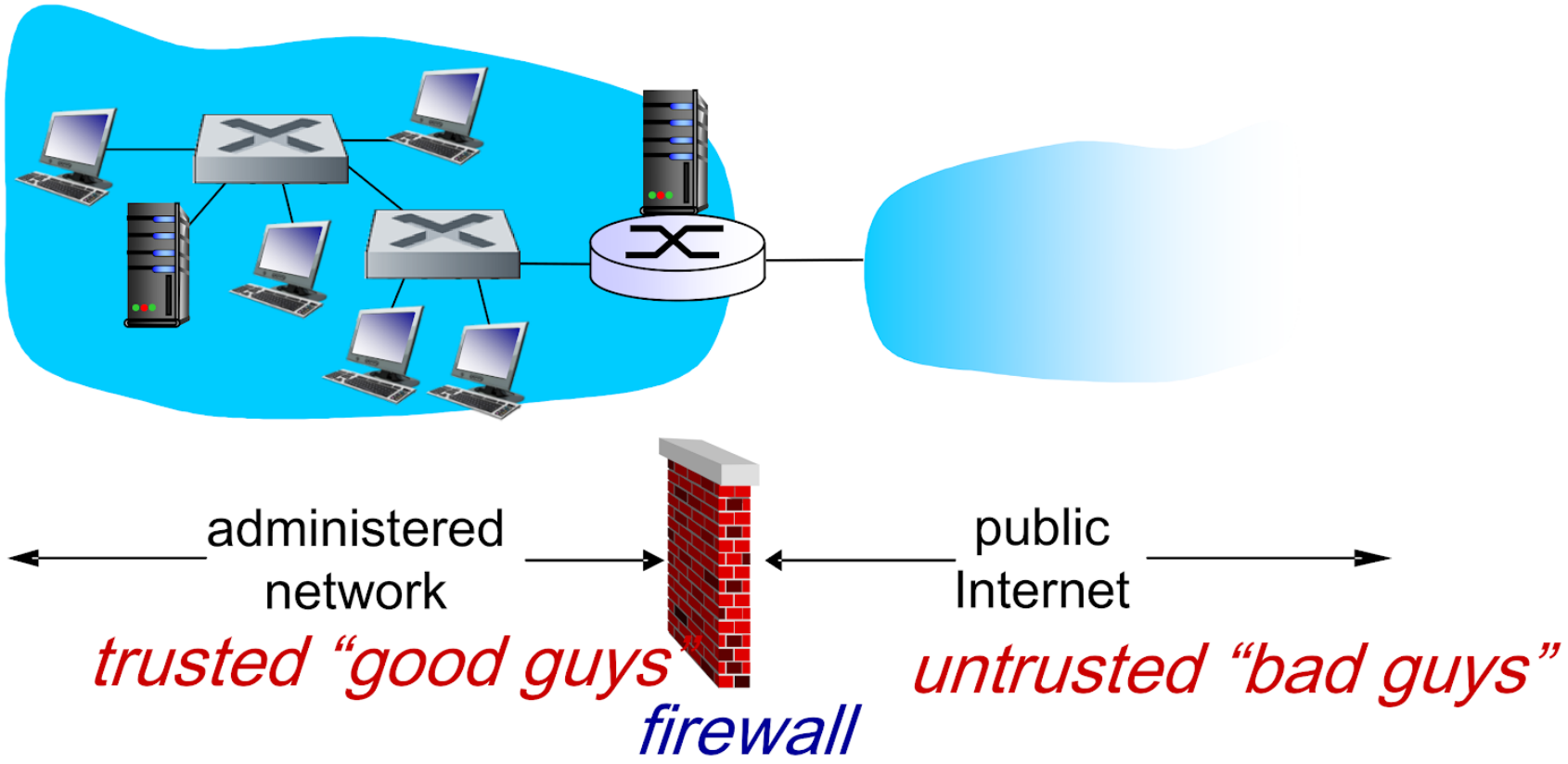
IP Addr	Mac Addr
10.0.0.2	~~00:10:20:30:ac:06~~ 00:01:12:44:3a:6c

User B - ARP cache

IP Addr	Mac Addr
10.0.0.1	~~00:12:3a:00:45:bc~~ 00:01:12:44:3a:6c

Recap: Firewalls

- Isolate organization's internal net from larger Internet, allowing some packets to pass, blocking others



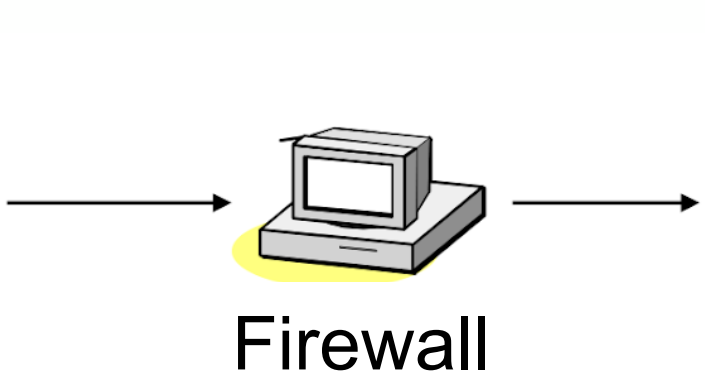
Recap: Intrusion Detection

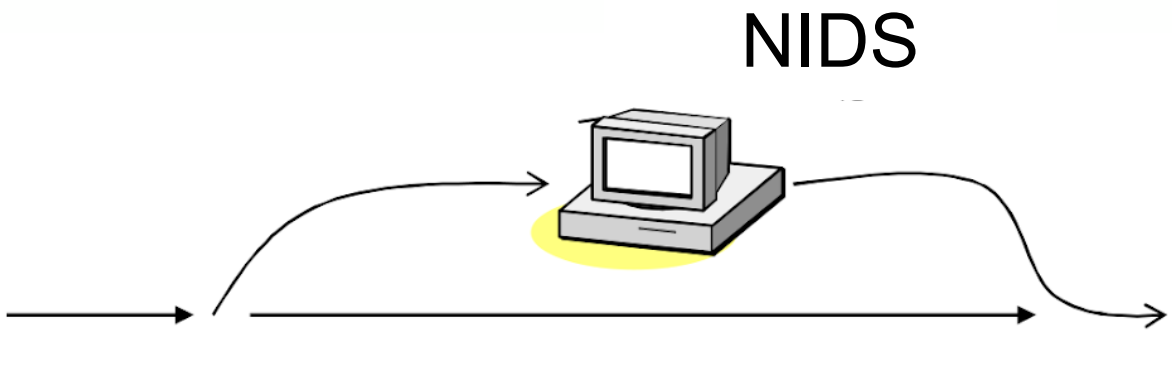
- Intrusion
  - A set of actions aimed to compromise the security goals
- Intrusion detection
  - The process of identifying and responding to intrusion activities

Recap: Firewall vs. IDS

- Firewall
  - Active filtering (prevent intrusion)
  - Location: Between networks (if an attack is from inside the network it doesn't signal)

- IDS
  - Passive monitoring (detect intrusion)
  - Location: Inside the network





Recap: Web Threat Models

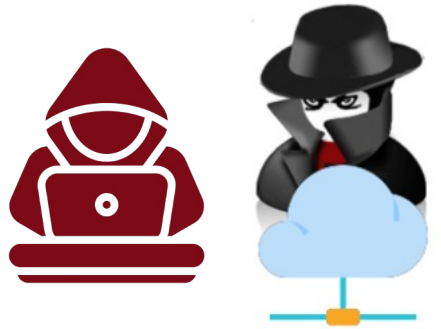
- **Network attacker**: resides somewhere in the communication link between client and server
  - Passive: evasdropping
  - Active: modification of messages, replay…

- **Remote attacker:** can connect to remote system via the network
  - Mostly targets the server

- **Web attacker**: controls attacker.com
  - Can obtain SSL/TLS certificates for attacker.com
  - Users can visit attacker.com

Today's Topic

- **Network attacker**: resides somewhere in the communication link between client and server
  - Passive: evasdropping
  - Active: modification of messages, replay…

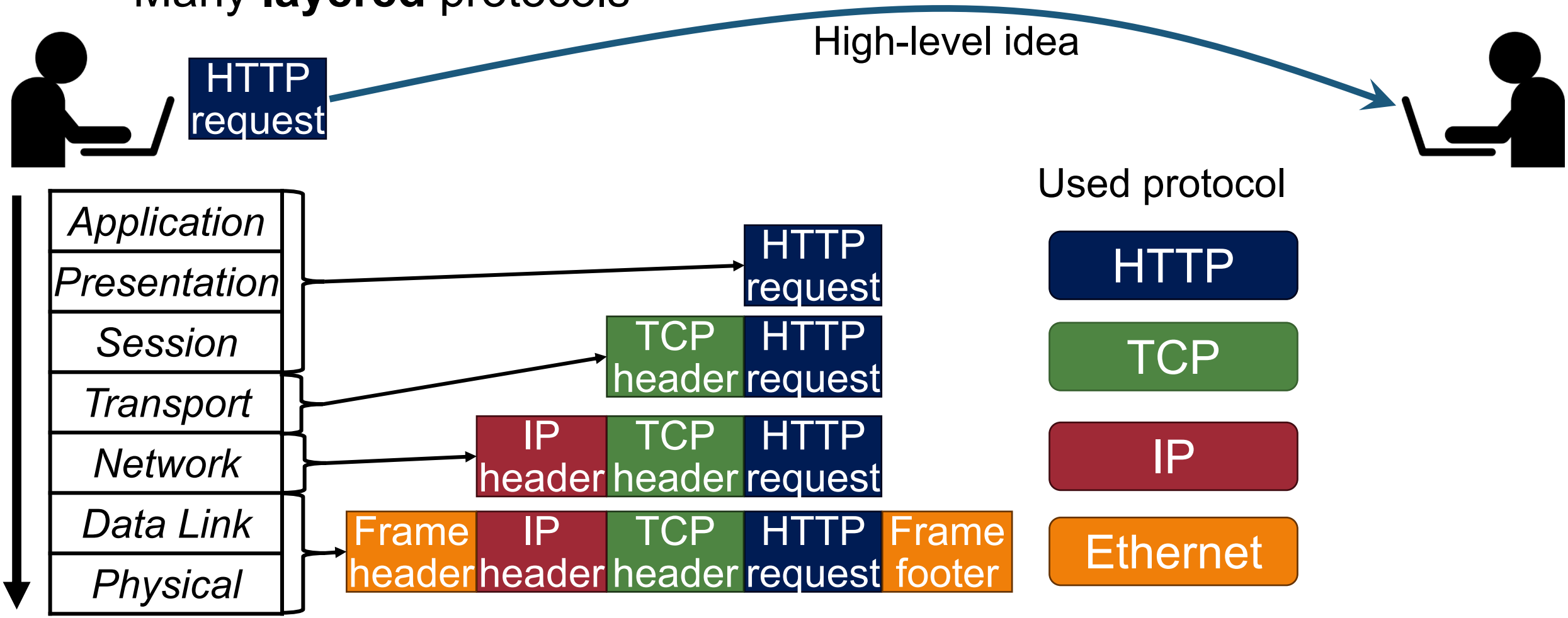


- **Remote attacker:** can connect to remote system via the network
  - Mostly targets the server

- **Web attacker**: controls attacker.com
  - Can obtain SSL/TLS certificates for attacker.com
  - Users can visit attacker.com

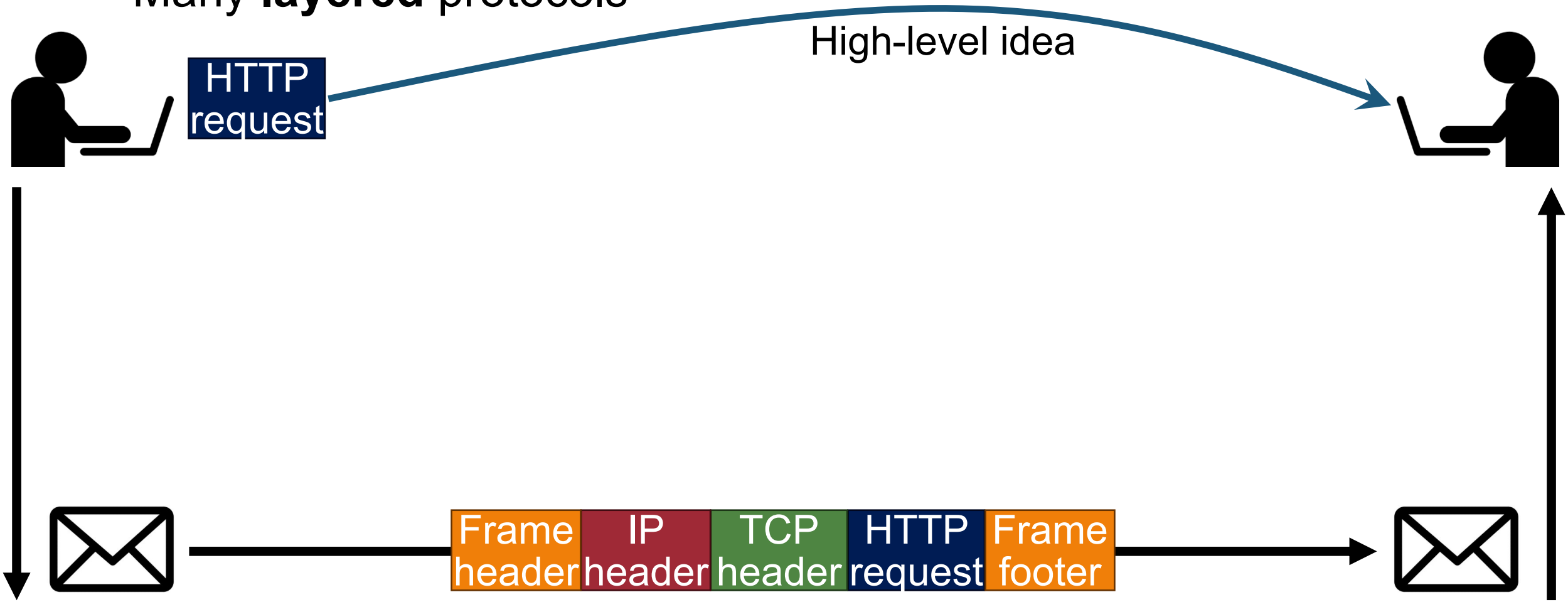
Recap: Protocol

- A system of digital **rules** for data exchange between computers
- Many **layered** protocols



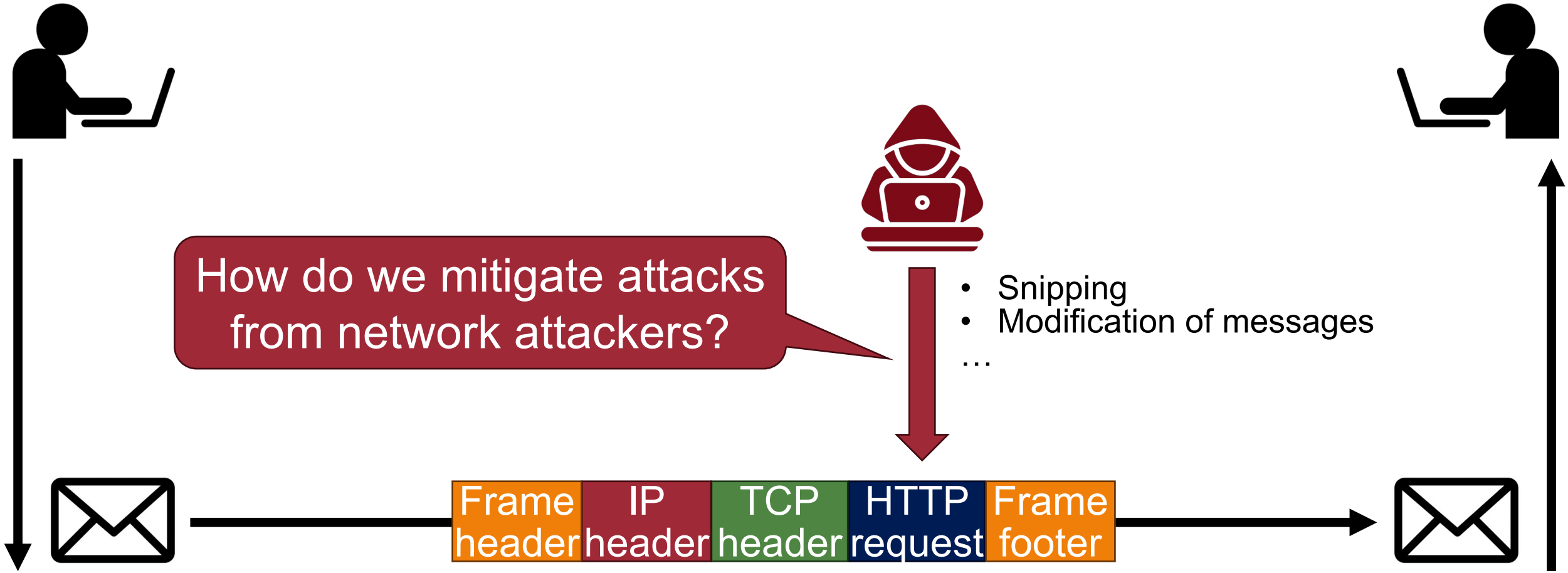
Recap: Protocol

- A system of digital **rules** for data exchange between computers
- Many **layered** protocols



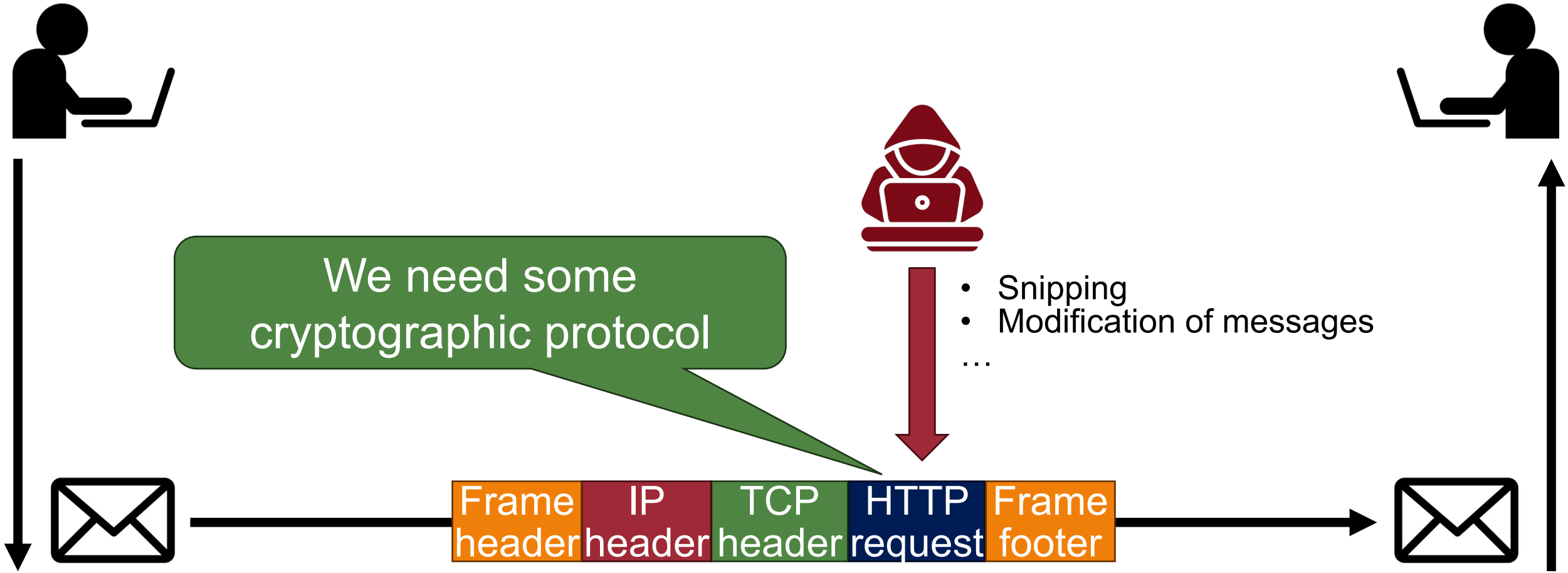
Network Attackers

- A system of digital **rules** for data exchange between computers
- Many **layered** protocols



Motivation: Cryptographical Protocol

- A system of digital **rules** for data exchange between computers
- Many **layered** protocols



SSL/TLS 

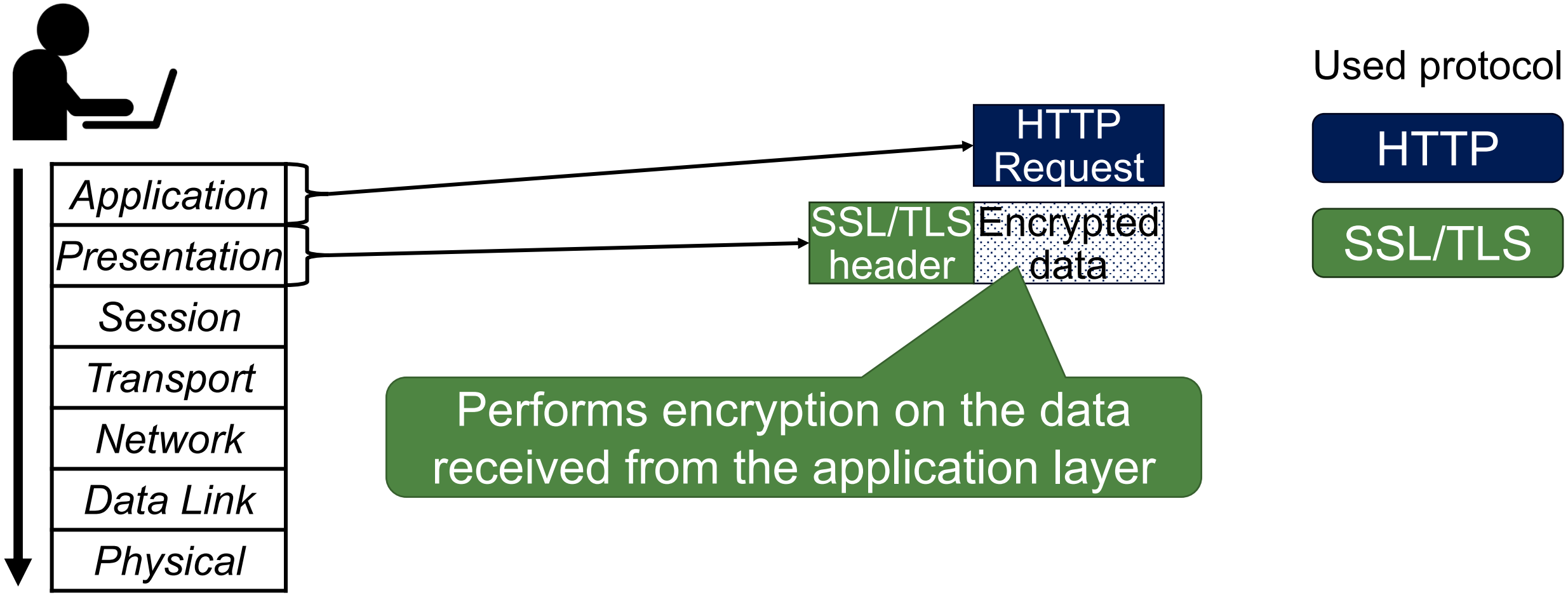
Related to cryptography, network security, web security, and software security!

What is SSL/TLS?

- **Secure Sockets Layer (SSL)** and **Transport Layer Security (TLS)** protocols
  - Same protocol design, different crypto algorithms
  - (Reserved) port number: 443
- Security goals: achieving…
  - Confidentiality
  - Integrity
  - Authentication
- ***De facto* standard for Internet security**

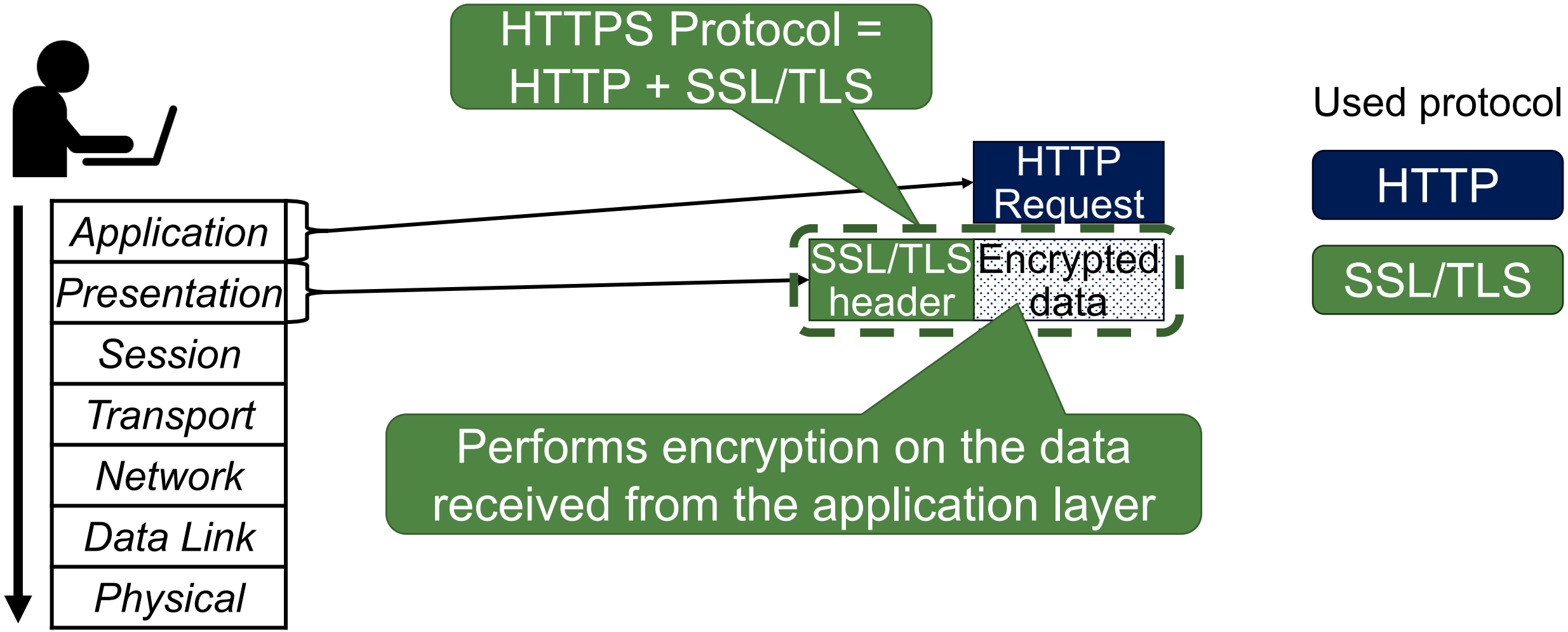
SSL/TLS Basic Idea

- Adding a protocol layer for secure communication!



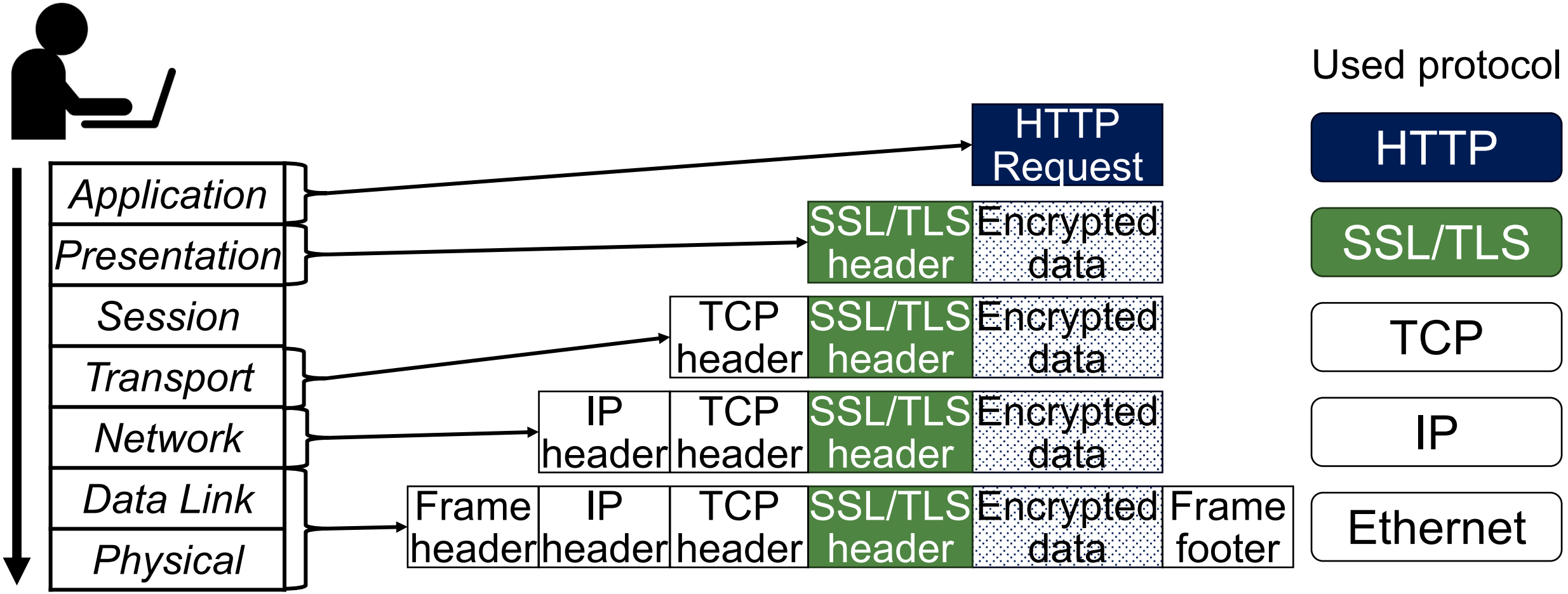
SSL/TLS Basic Idea

- Adding a protocol layer for secure communication!



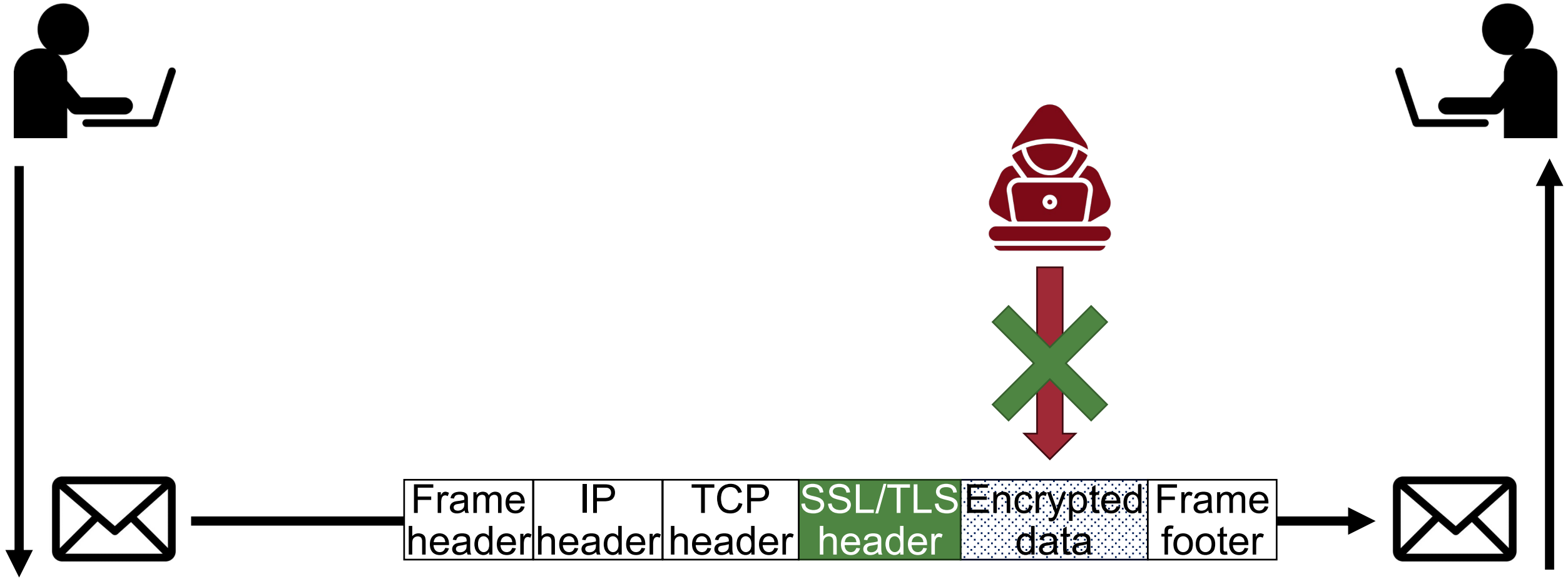
SSL/TLS Basic Idea

- Adding a protocol layer for secure communication!



SSL/TLS Basic Idea

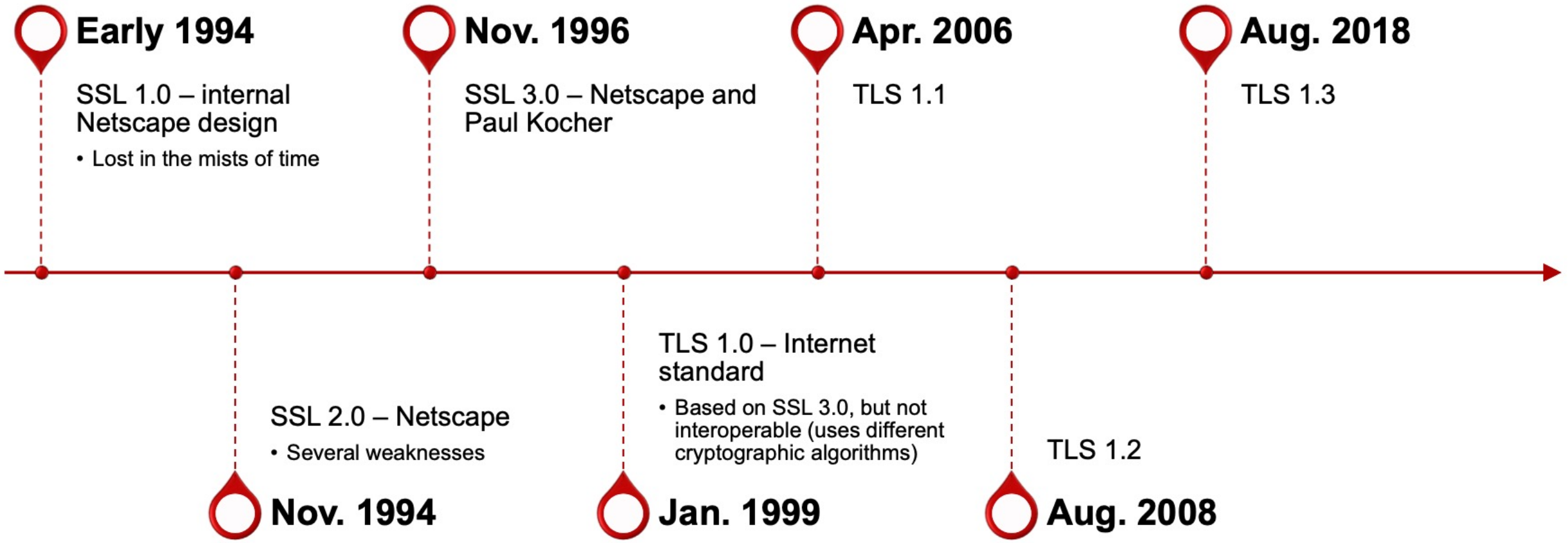
- Adding a protocol layer for secure communication!



Use Cases

- Email
- Vice over IP (VoIP)
- Payment systems (transactions)
- **HTTPS**
  - The most publicly visible use case!

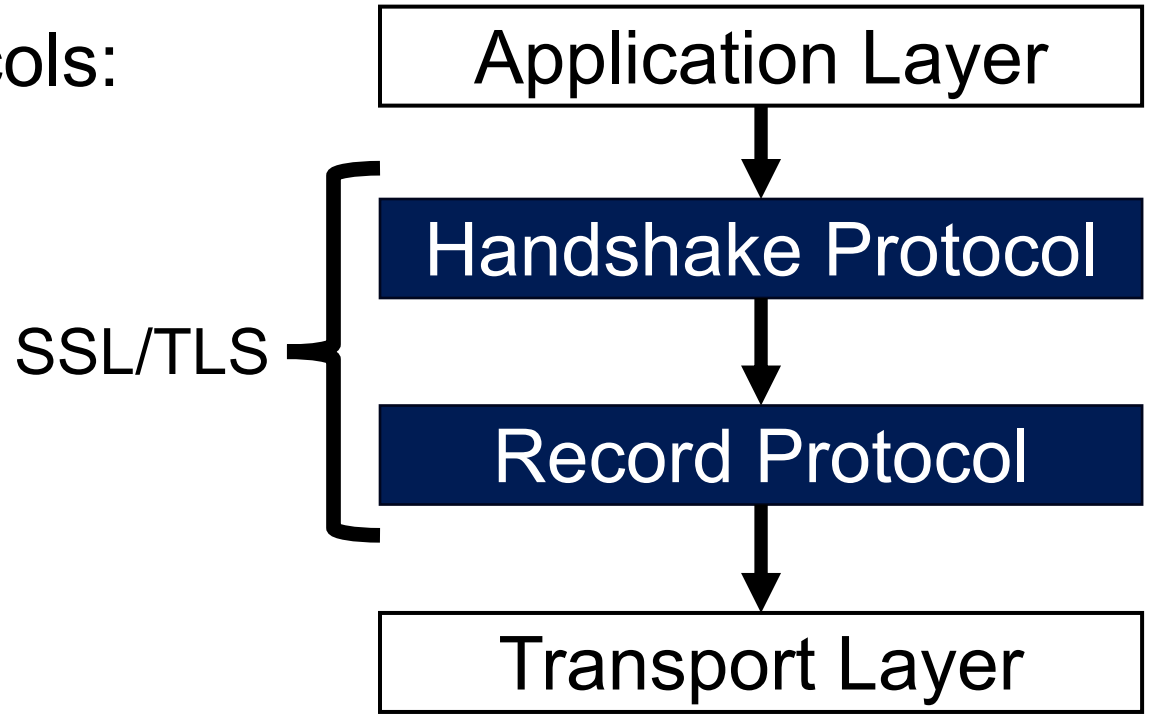
History of the Protocol



SSL/TLS Basics

- Runs in the presentation layer
- Uses symmetric crypto, asymmetric crypto, and digital signatures

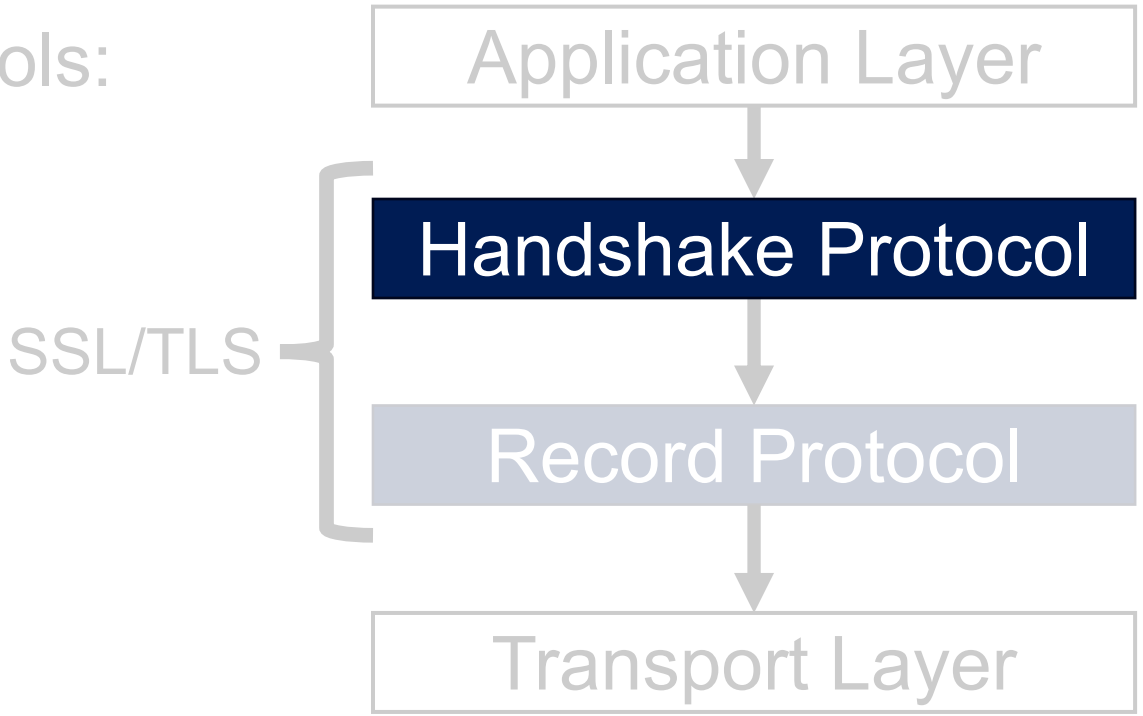
- Composed of two layers of protocols:
  1. Handshake protocol
  2. Record protocol



SSL/TLS Basics

- Runs in the presentation layer
- Uses symmetric crypto, asymmetric crypto, and digital signatures

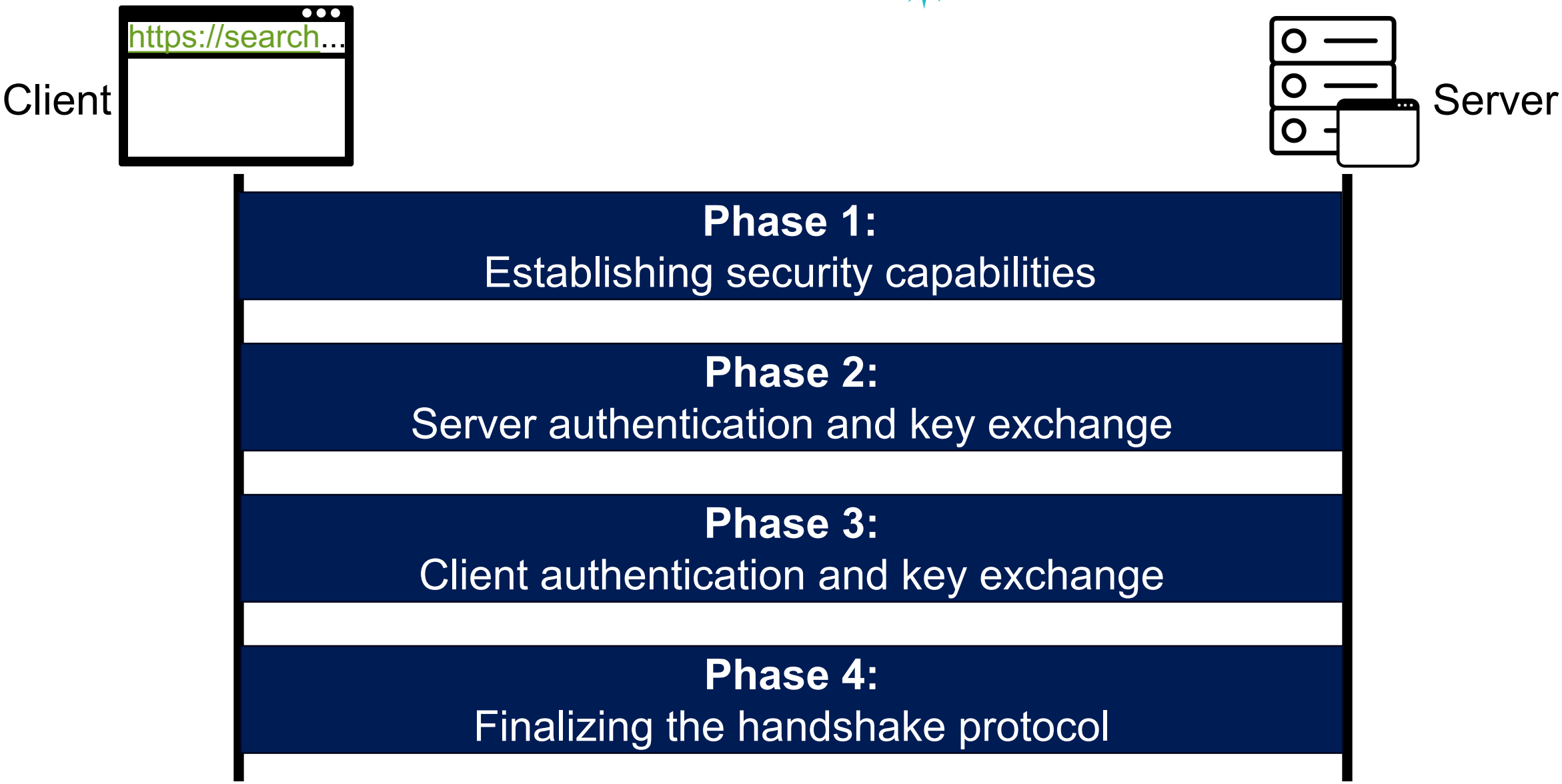
- Composed of two layers of protocols:
  1. Handshake protocol
  2. Record protocol



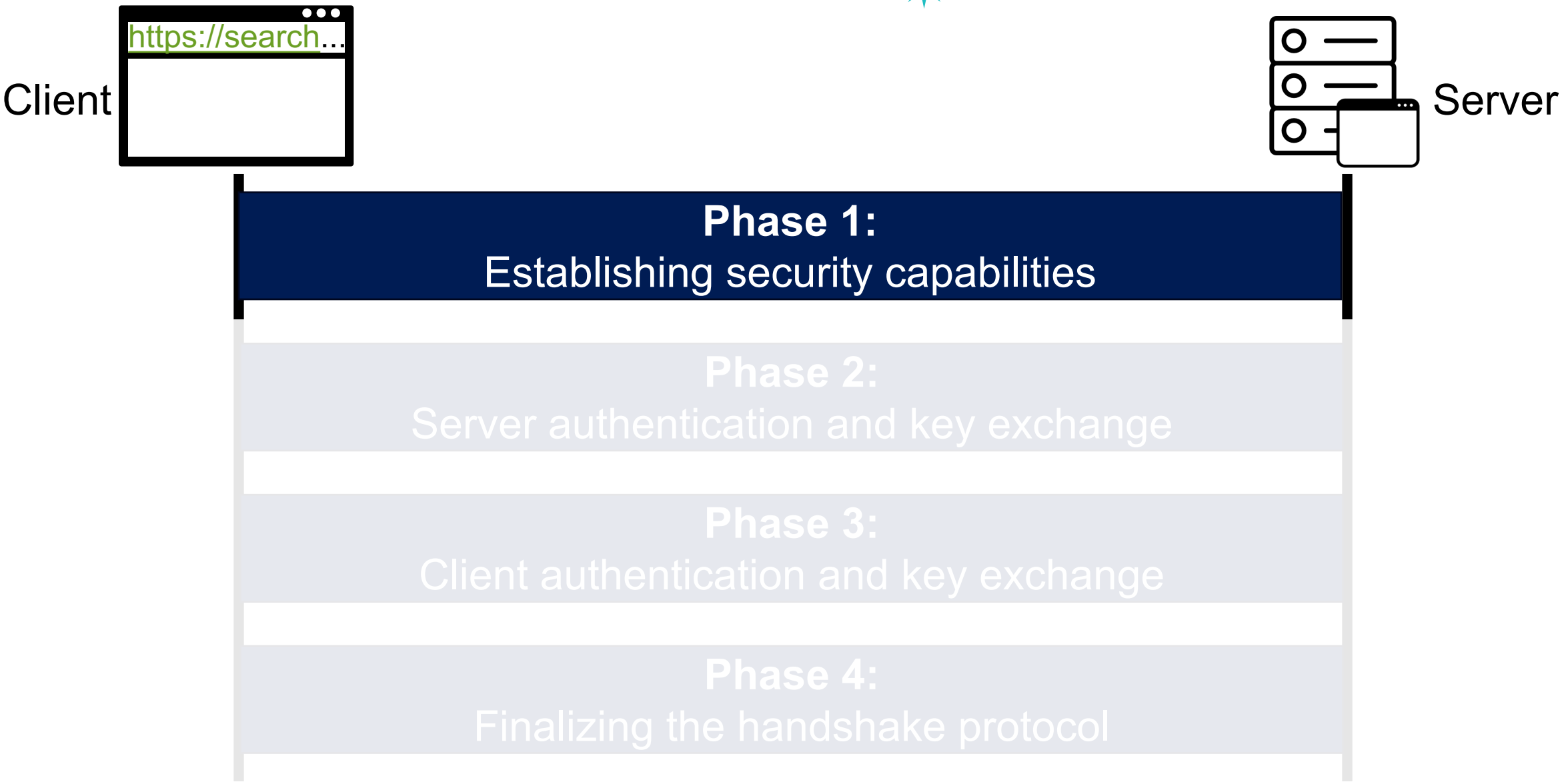
SSL/TLS Handshake Protocol

- The most complex part of SSL
- Uses asymmetric cryptography to establish **several shared secret secret**

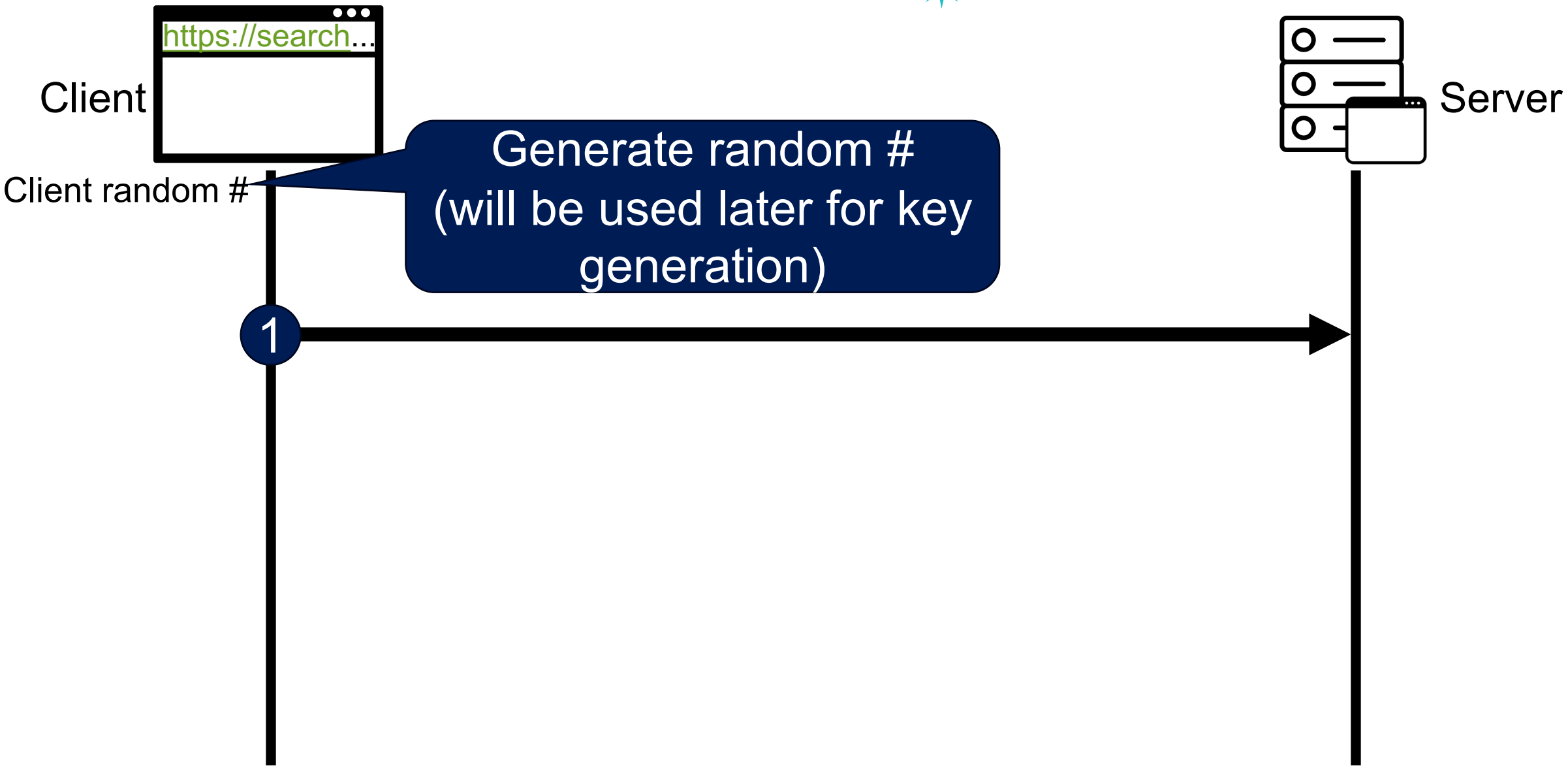
Four Phases of Handshake Protocol



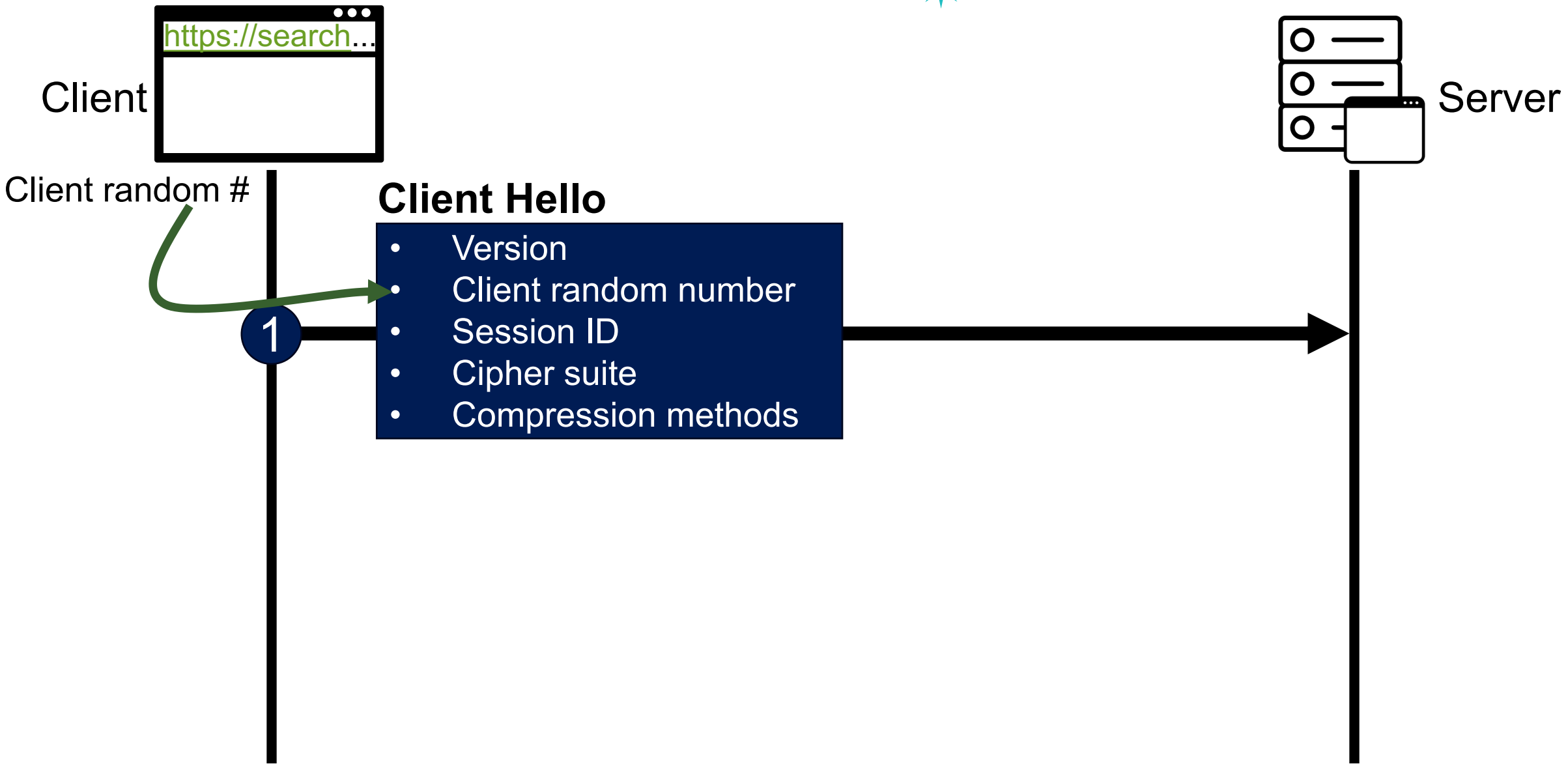
Phase 1: Establishing Security Capabilities



Phase 1: Establishing Security Capabilities



Phase 1: Establishing Security Capabilities



Phase 1 – Client Hello – Details

Client Hello – Details

- **Version**
  - Highest protocol version supported by the client
- **Client random number**
  - Random 32 bit time stamp + 28 random bytes
  - It will be used later for key generation
- **Session ID**
  - 0: establish new connection on new session
  - Non-zero: resume an old session
- **Cipher suite**
  - Set of cryptographic algorithms supported by the client
- **Compression methods**
  - Sequence of compression methods

Cipher Suites

Client Hello – Details

- **Version**
  - Highest protocol version supported by the client
- **Client random number**
  - Random 32 bit time stamp + 28 random bytes
  - It will be used later for key generation
- **Session ID**
  - 0: establish new connection on new session
  - Non-zero: resume an old session
- **Cipher suite**
  - Set of cryptographic algorithms supported by the client
- **Compression methods**
  - Sequence of compression methods

Format:

```
TLS_RSA_WITH_ASE_128_CBC_SHA
```

Cipher Suites

Client Hello – Details

- **Version**
  - Highest protocol version supported by the client
- **Client random number**
  - Random 32 bit time stamp +
  - It will be used later for key generation
- **Session ID**
  - 0: establish
  - Non-zero: re
- **Cipher suite**
  - Set of cryptographic algorithms supported by the client
- **Compression methods**
  - Sequence of compression methods

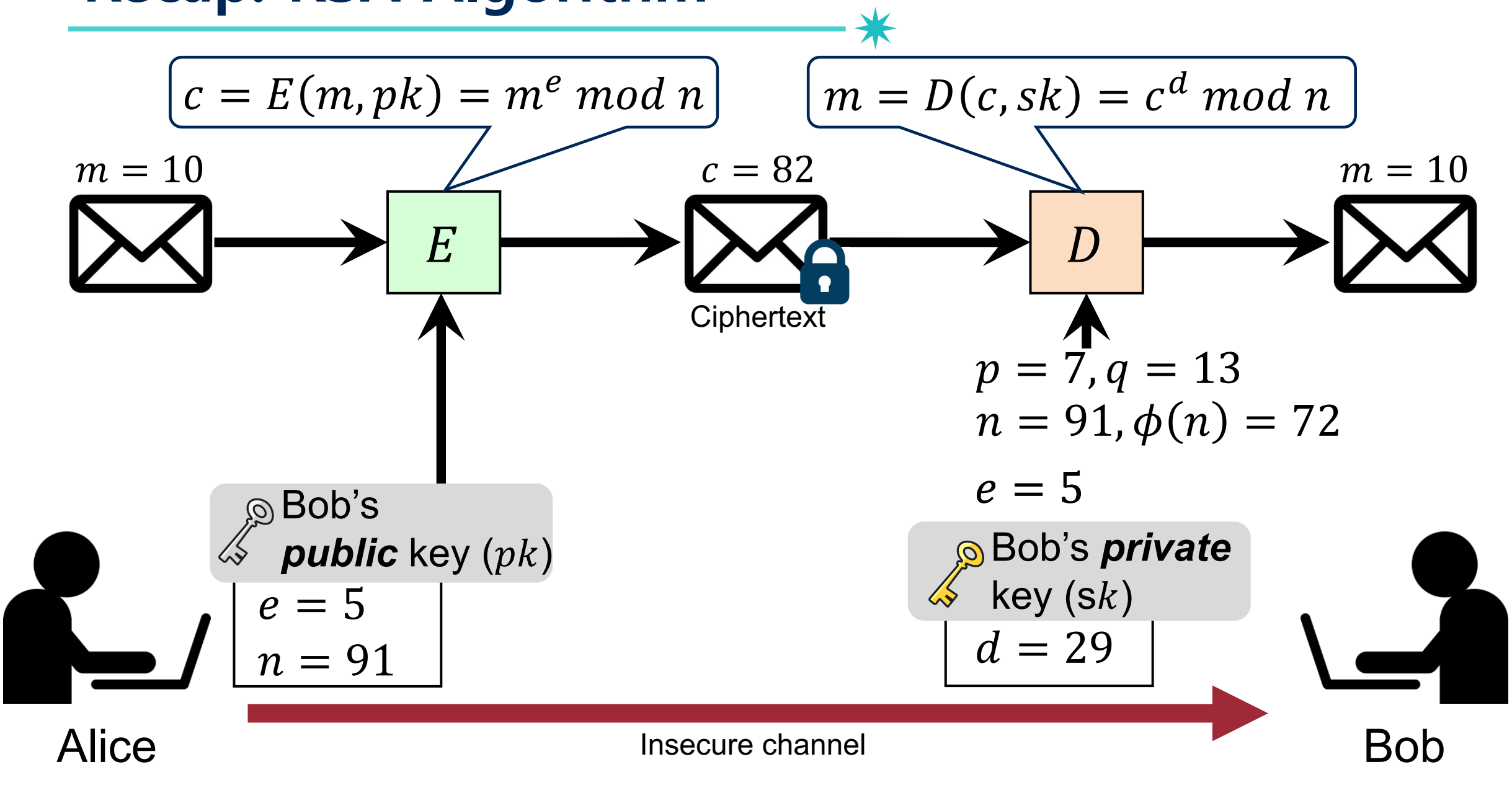
Format:

`TLS_RSA_WITH_ASE_128_CBC_SHA`

Protocol

(Asymmetric) Encryption/decryption algorithm (for key exchange)

Recap: RSA Algorithm



Recap: Diffie-Hellman Key Exchange

Symmetric key:

$K = g^{ab} \bmod p$

$p = 23, g = 9$

$A = (g^a \bmod p) = 6$

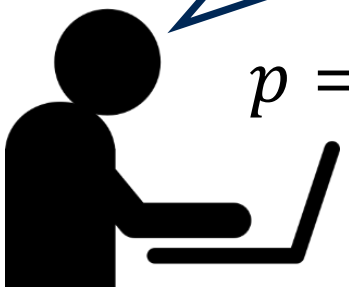
$B = (g^b \bmod p) = 16$

$K = (B^{\boldsymbol{a}} \bmod p) = (g^{ab} \bmod p)$

$= (16^4 \bmod 23) = 9$

$K = (A^{\boldsymbol{b}} \bmod p) = (g^{ab} \bmod p)$

$= (6^3 \bmod 23) = 9$



$\boldsymbol{a = 4}$

$p = 23, g = 9$

$A = (g^a \bmod p) = 6$

$B = (g^b \bmod p) = 16$

$\boldsymbol{b = 3}$

$p = 23, g = 9$

$A = (g^a \bmod p) = 6$

$B = (g^b \bmod p) = 16$

Alice

Insecure channel

Bob

Cipher Suites

Client Hello – Details

- **Version**
  - Highest protocol version supported by the client
- **Client random number**
  - Random 32 bit time stamp +
  - It will be used later for key generation
- **Session ID**
  - 0: establish
  - Non-zero: re
- **Cipher suite**
  - Set of cryptographic algorithm
    the client
- **Compression methods**
  - Sequence of compression me

Format:

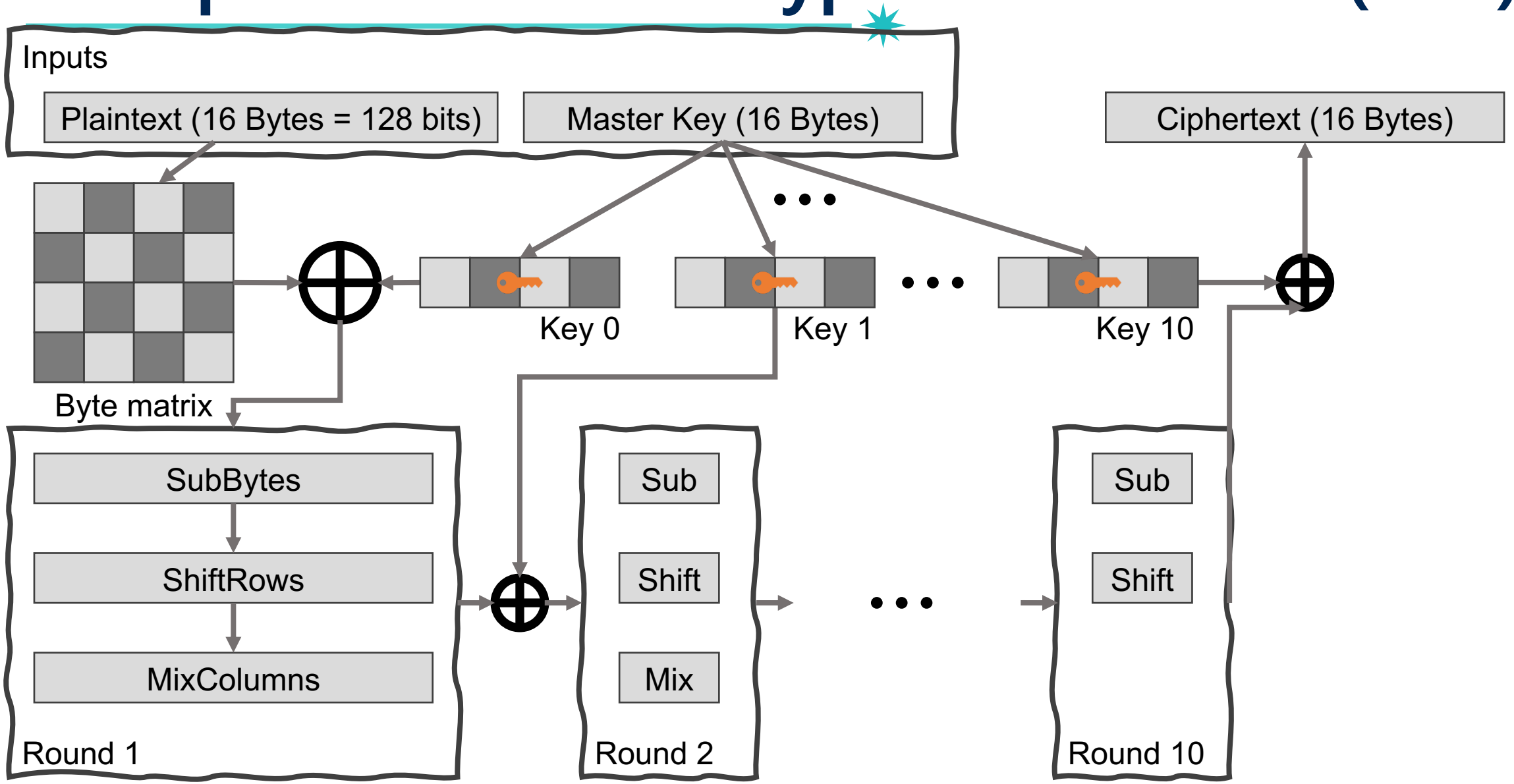
`TLS_RSA_WITH_ASE_128_CBC_SHA`

Protocol

(Asymmetric) Encryption/decryption algorithm (for key exchange)

(Symmetric) Encryption/decryption algorithm (for data exchange)

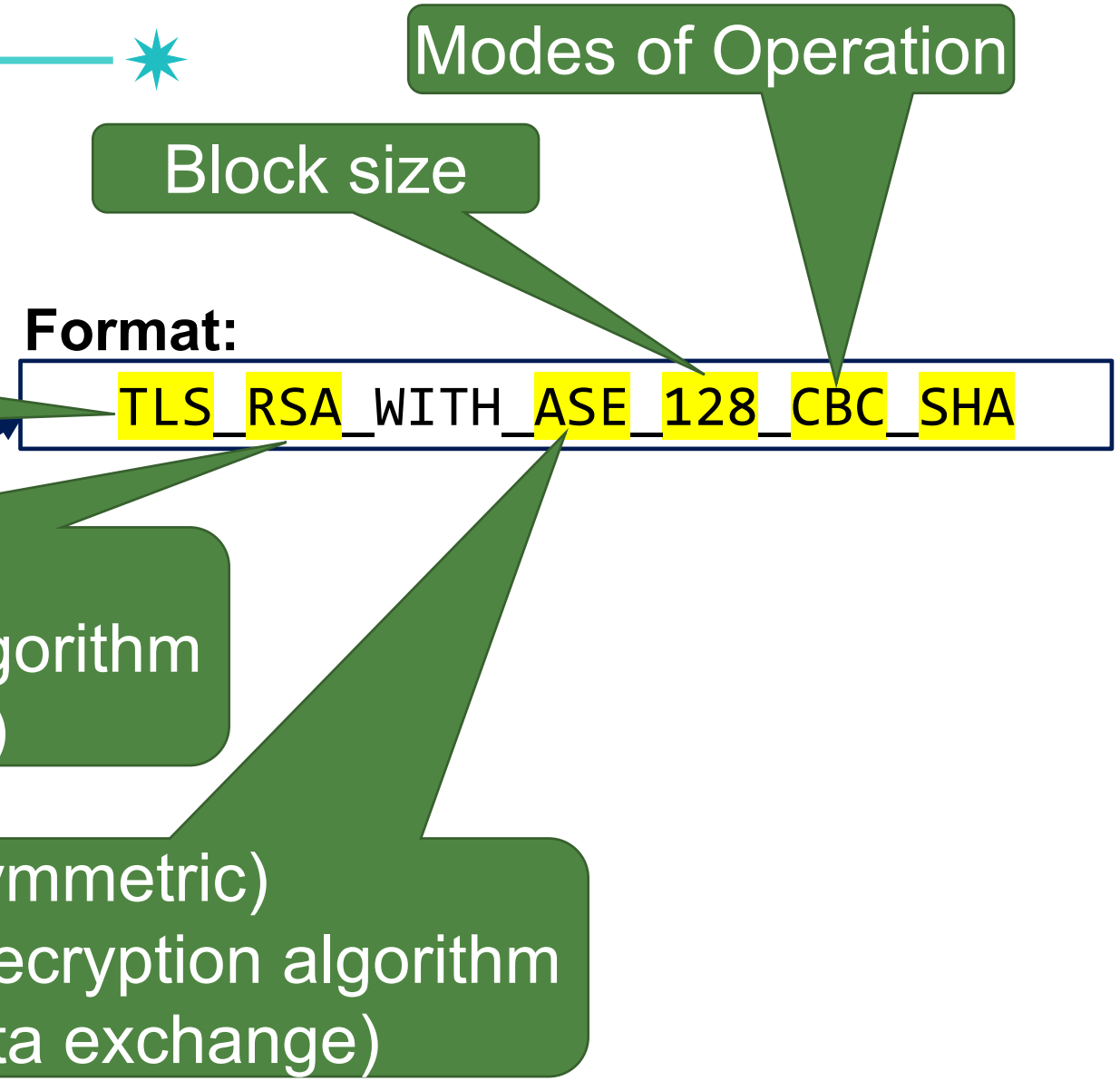
Recap: Advanced Encryption Standard (AES)



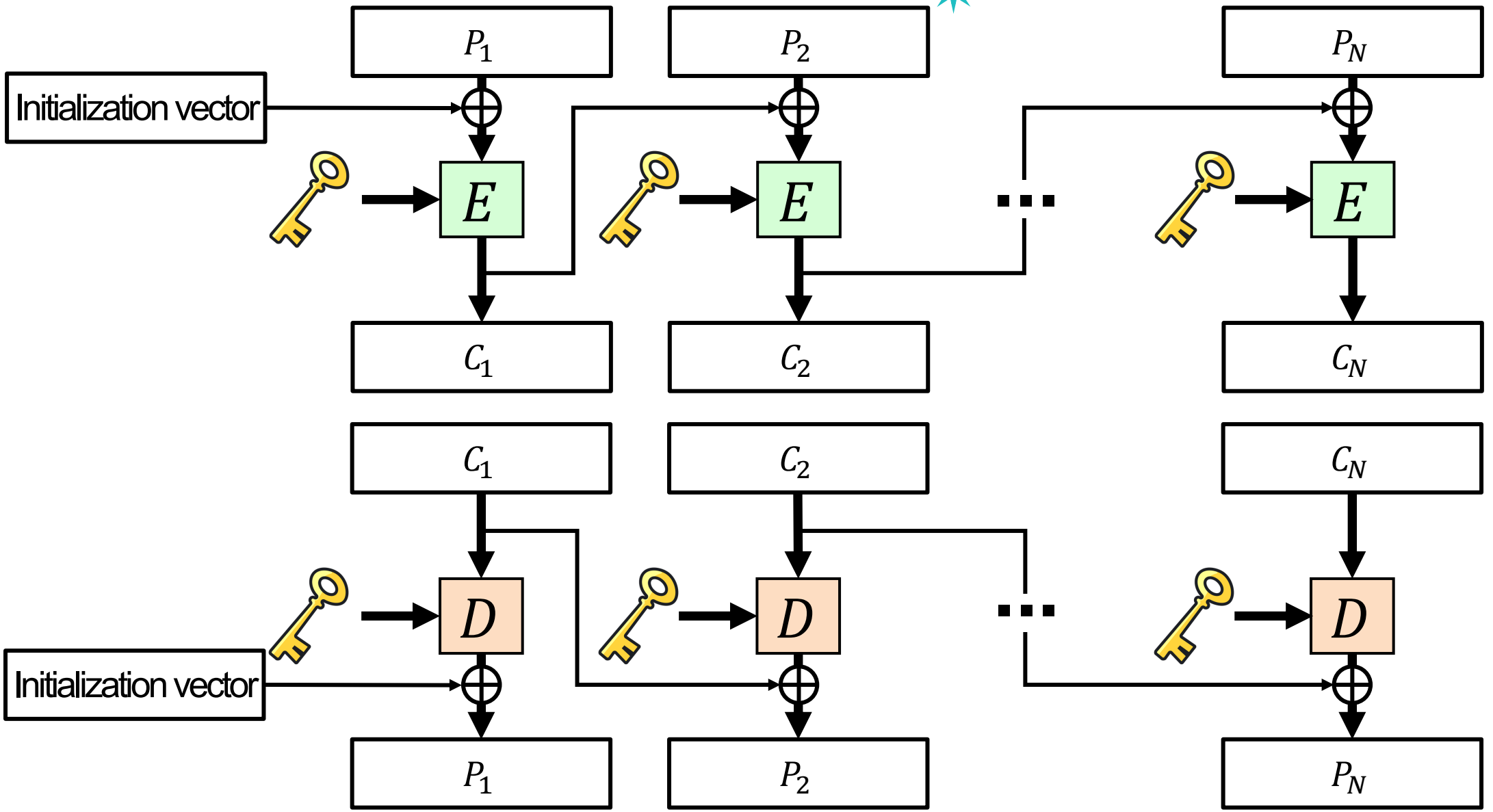
Cipher Suites

Client Hello – Details

- **Version**
  - Highest protocol version supported by the client
- **Client random number**
  - Random 32 bit time stamp +
  - It will be used later for key generation
- **Session ID**
  - 0: establish
  - Non-zero: re
- **Cipher suite**
  - Set of cryptographic algorithm
    the client
- **Compression methods**
  - Sequence of compression me



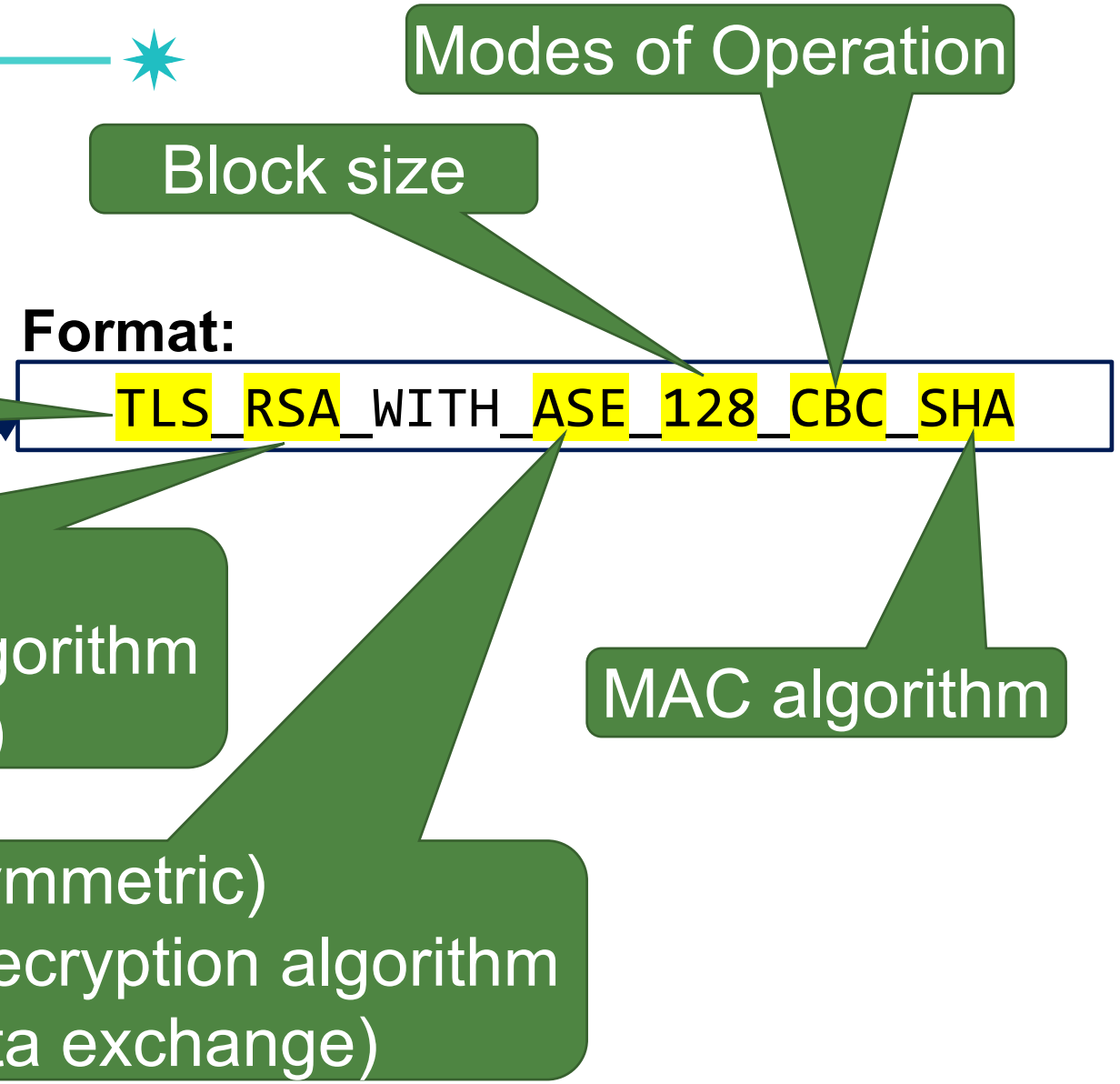
Recap: Cipher Block Chaining (CBC)



Cipher Suites

Client Hello – Details

- **Version**
  - Highest protocol version supported by the client
- **Client random number**
  - Random 32 bit time stamp +
  - It will be used later for key generation
- **Session ID**
  - 0: establish
  - Non-zero: re
- **Cipher suite**
  - Set of cryptographic algorithm
    the client
- **Compression methods**
  - Sequence of compression me



Cipher Suite – Example

Cipher Suite	Key Exchange	Cipher	MAC
TLS_NULL_WITH_NULL_NULL	NULL	NULL	NULL
TLS_RSA_WITH_NULL_MD5	RSA	NULL	MD5
TLS_RSA_WITH_NULL_SHA	RSA	NULL	SHA
TLS_RSA_WITH_NULL_SHA256	RSA	NULL	SHA256
TLS_RSA_WITH_RC4_128_MD5	RSA	RC4_128	MD5
TLS_RSA_WITH_RC4_128_SHA	RSA	RC4_128	SHA
TLS_RSA_WITH_3DES_EDE_CBC_SHA	RSA	3DES_EDE_CBC	SHA
TLS_RSA_WITH_AES_128_CBC_SHA	RSA	AES_128_CBC	SHA
TLS_RSA_WITH_AES_256_CBC_SHA	RSA	AES_256_CBC	SHA
TLS_RSA_WITH_AES_128_CBC_SHA256	RSA	AES_128_CBC	SHA256
TLS_RSA_WITH_AES_256_CBC_SHA256	RSA	AES_256_CBC	SHA256
TLS_DH_anon_WITH_RC4_128_MD5	DH_anon	RC4_128	MD5
TLS_DH_anon_WITH_3DES_EDE_CBC_SHA	DH_anon	3DES_EDE_CBC	SHA
TLS_DH_DSS_WITH_AES_128_CBC_SHA	DH_DSS	AES_128_CBC	SHA
TLS_DH_RSA_WITH_AES_128_CBC_SHA	DH_RSA	AES_128_CBC	SHA
TLS_DHE_DSS_WITH_AES_128_CBC_SHA	DHE_DSS	AES_128_CBC	SHA
TLS_DHE_RSA_WITH_AES_128_CBC_SHA	DHE_RSA	AES_128_CBC	SHA
TLS_DH_anon_WITH_AES_128_CBC_SHA	DH_anon	AES_128_CBC	SHA
TLS_DH_DSS_WITH_AES_256_CBC_SHA	DH_DSS	AES_256_CBC	SHA
TLS_DH_RSA_WITH_AES_256_CBC_SHA	DH_RSA	AES_256_CBC	SHA
TLS_DHE_DSS_WITH_AES_256_CBC_SHA	DHE_DSS	AES_256_CBC	SHA
TLS_DHE_RSA_WITH_AES_256_CBC_SHA	DHE_RSA	AES_256_CBC	SHA
TLS_DH_anon_WITH_AES_256_CBC_SHA	DH_anon	AES_256_CBC	SHA

No protection

Uses RSA (certificate) for key exchange, AES 256 in CBC mode for encryption and SHA256 as MAC

Uses ephemeral Diffie- Hellman with RSA for key exchange, AES 256 CBC for encryption and SHA256 as MAC

Cipher Suites

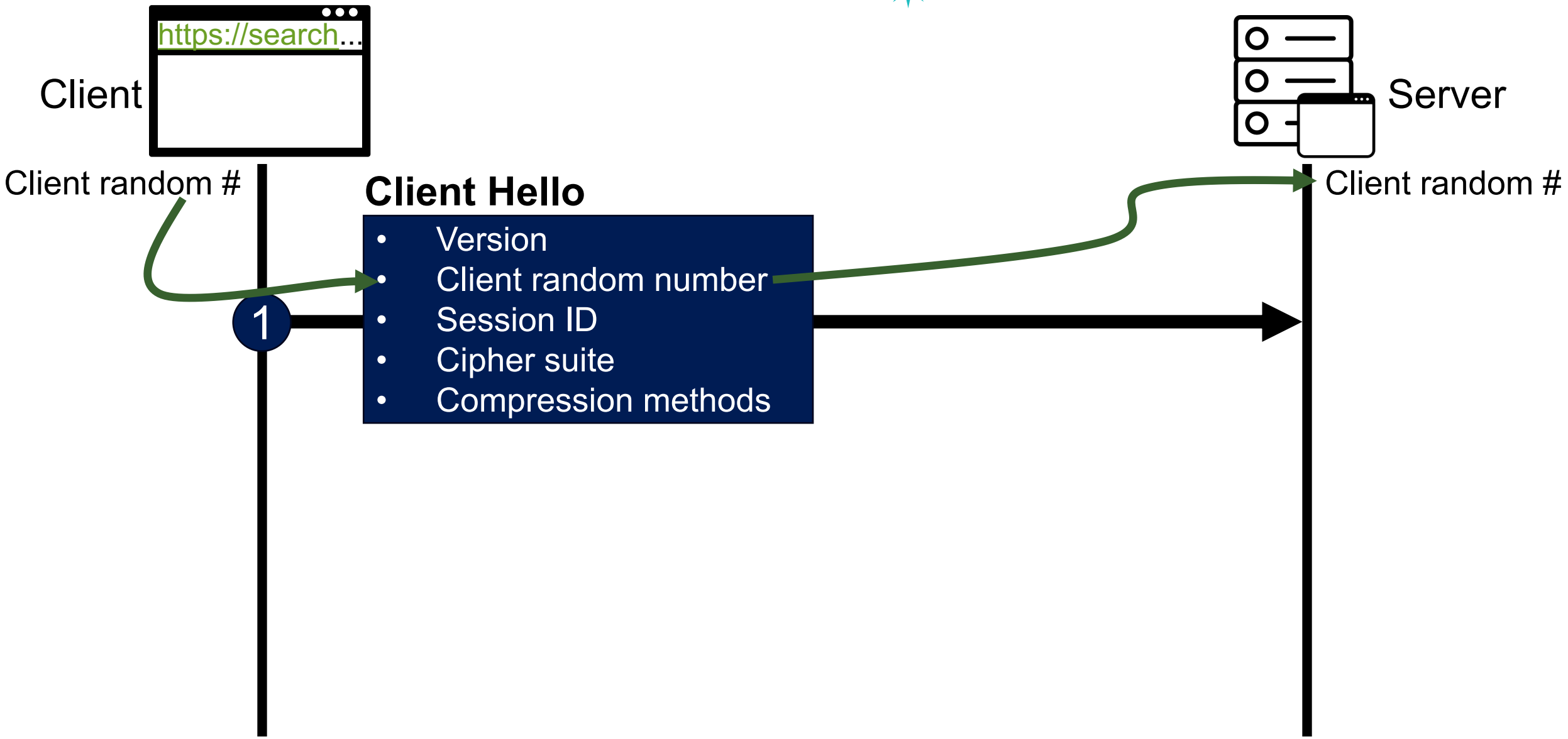
Client Hello –

- **Version**
  - Highest protocol version s
- **Client random number**
  -
  -
- **Se**
  - 0: establish new co
  - Non-zero: resume an old
- **Cipher suite**
  - Set of cryptographic algor the client
- **Compression methods**
  - Sequence of compression

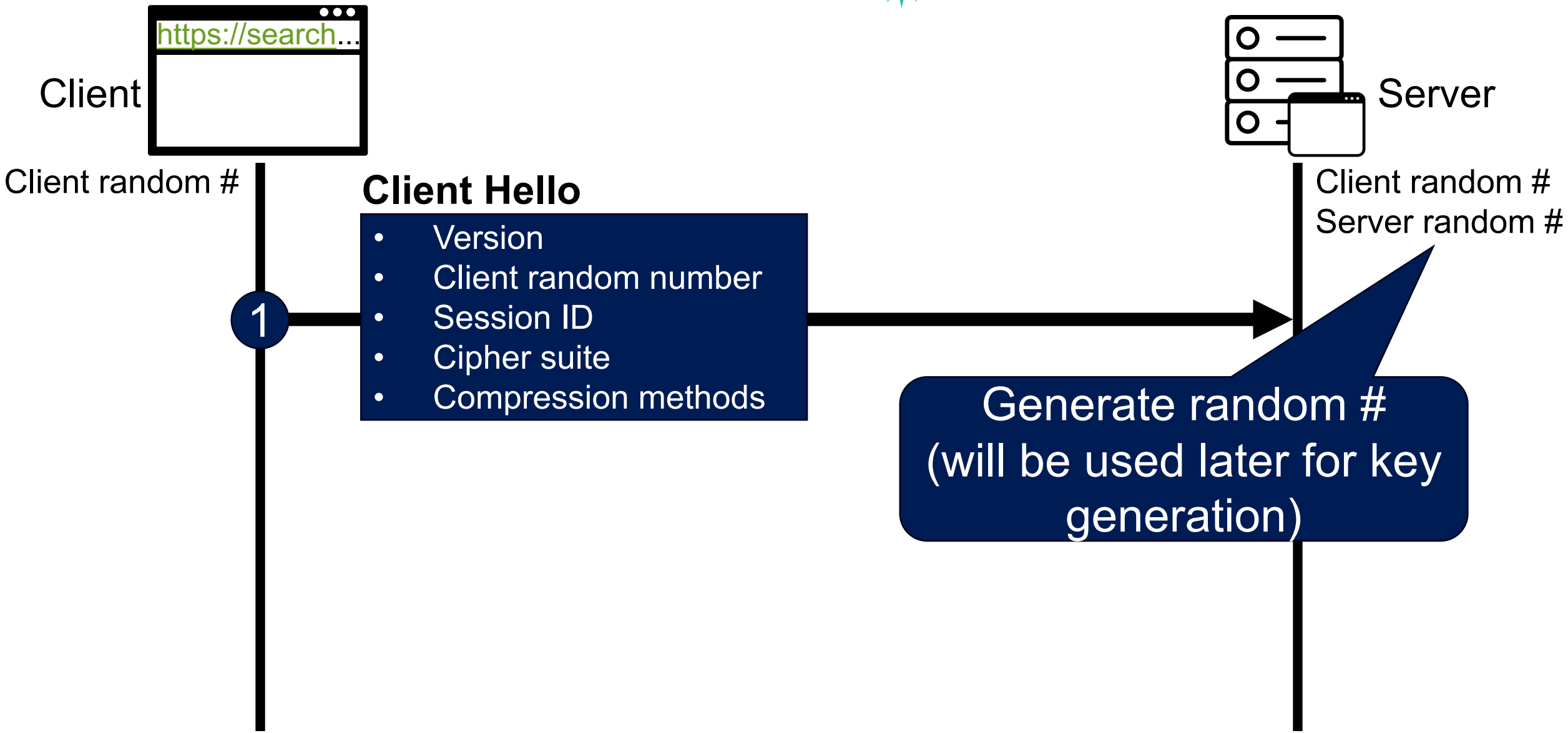
In decreasing order of preference

```
Transport Layer Security
  TLSv1.2 Record Layer: Handshake Protocol: Client Hello
      Content Type: Handshake (22)
      Version: TLS 1.0 (0x0301)
      Length: 512
    Handshake Protocol: Client Hello
      Handshake Type: Client Hello (1)
      Length: 508
      Version: TLS 1.2 (0x0303)
    > Random: 1396873af8d56db07f55a31afba6c98a04e00025005764fe…
      Session ID Length: 32
      Session ID: fe329526917d48c5af72228bdcb801142894fe91f4a548f7…
      Cipher Suites Length: 34
    Cipher Suites (17 suites)
        Cipher Suite: Reserved (GREASE) (0x3a3a)
        Cipher Suite: TLS_AES_128_GCM_SHA256 (0x1301)
        Cipher Suite: TLS_AES_256_GCM_SHA384 (0x1302)
        Cipher Suite: TLS_CHACHA20_POLY1305_SHA256 (0x1303)
        Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 (0xc02b)
        Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f)
        Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 (0xc02c)
        Cipher Suite: TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (0xc030)
        Cipher Suite: TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305_SHA256 (0xcca9)
        Cipher Suite: TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256 (0xcca8)
```

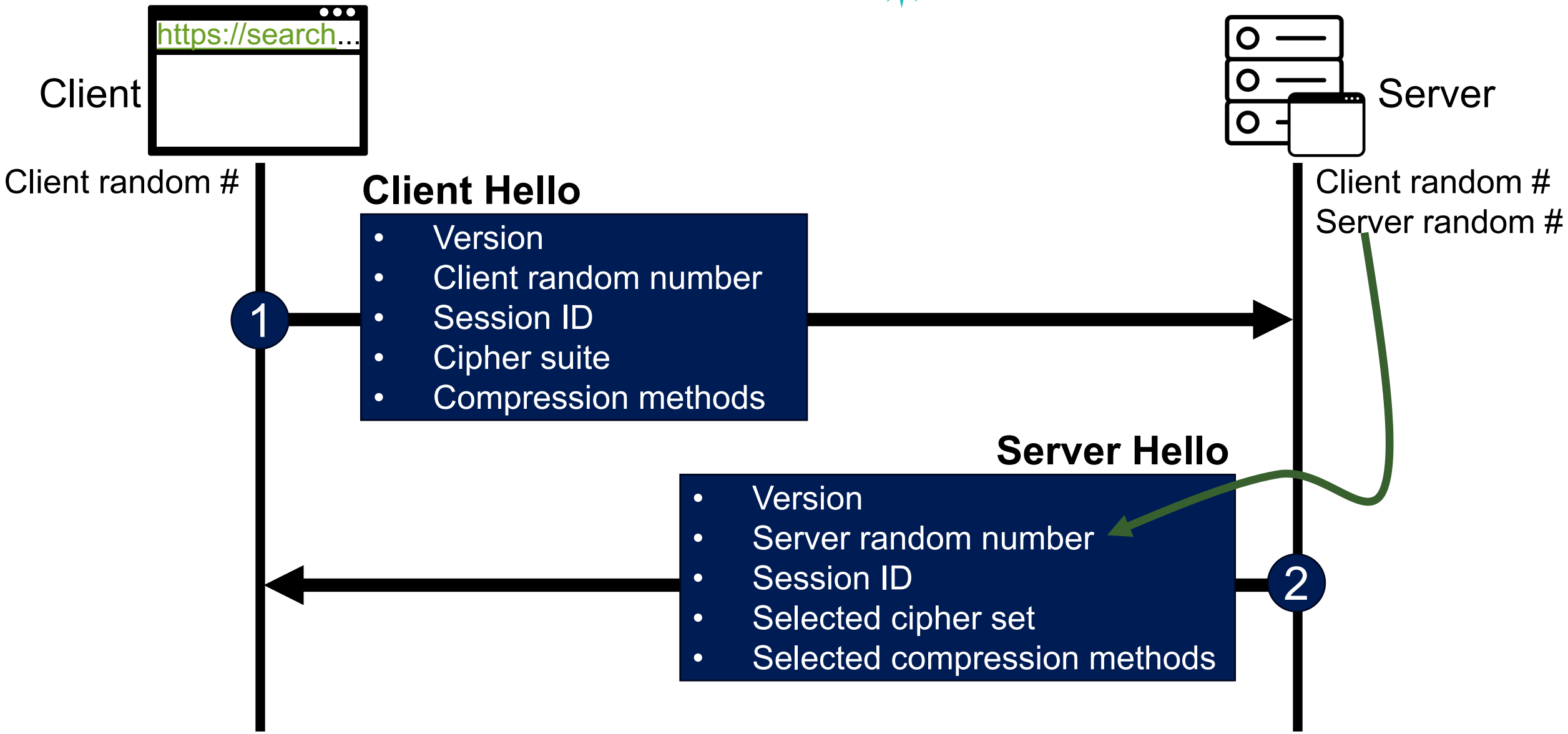
Phase 1: Establishing Security Capabilities



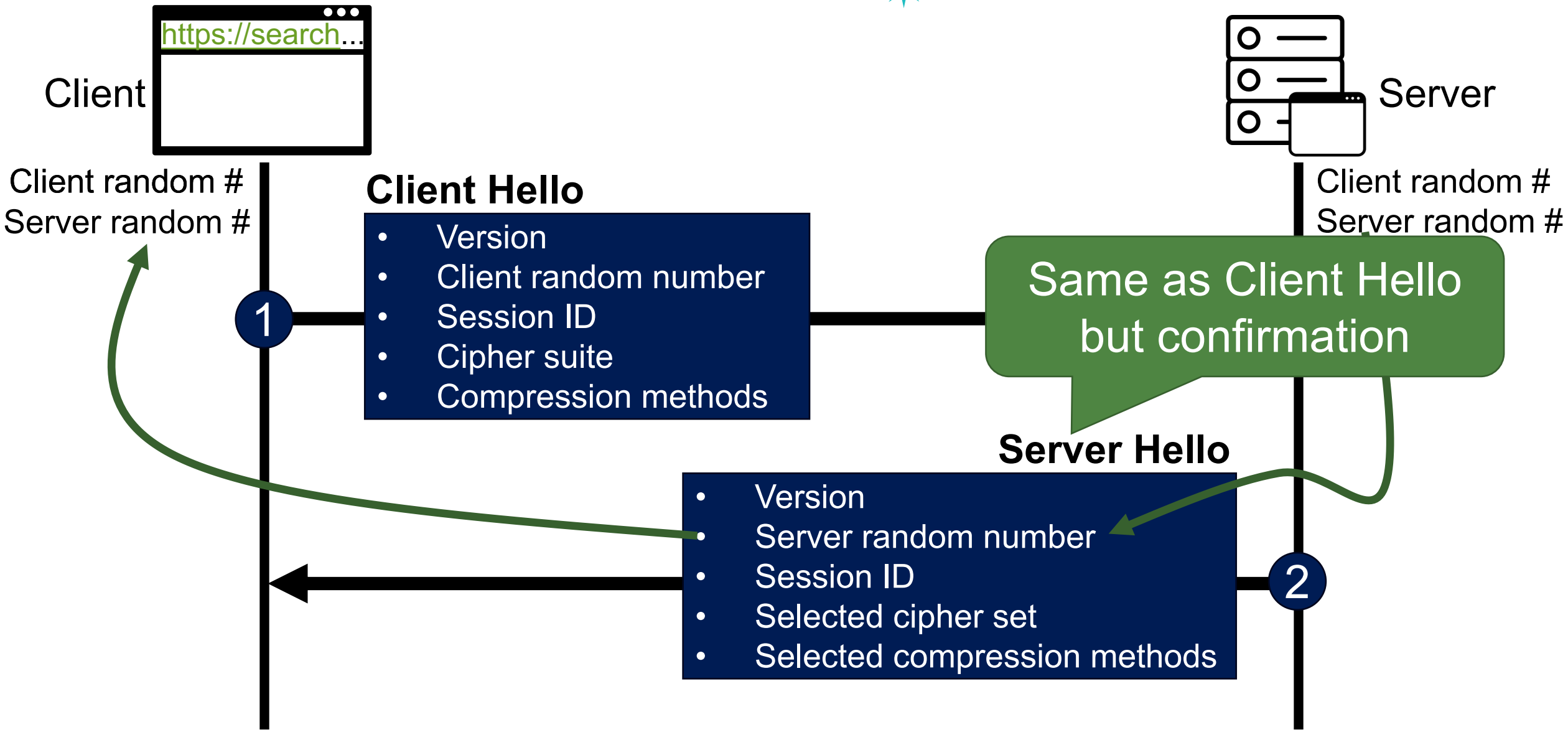
Phase 1: Establishing Security Capabilities



Phase 1: Establishing Security Capabilities



Phase 1: Establishing Security Capabilities



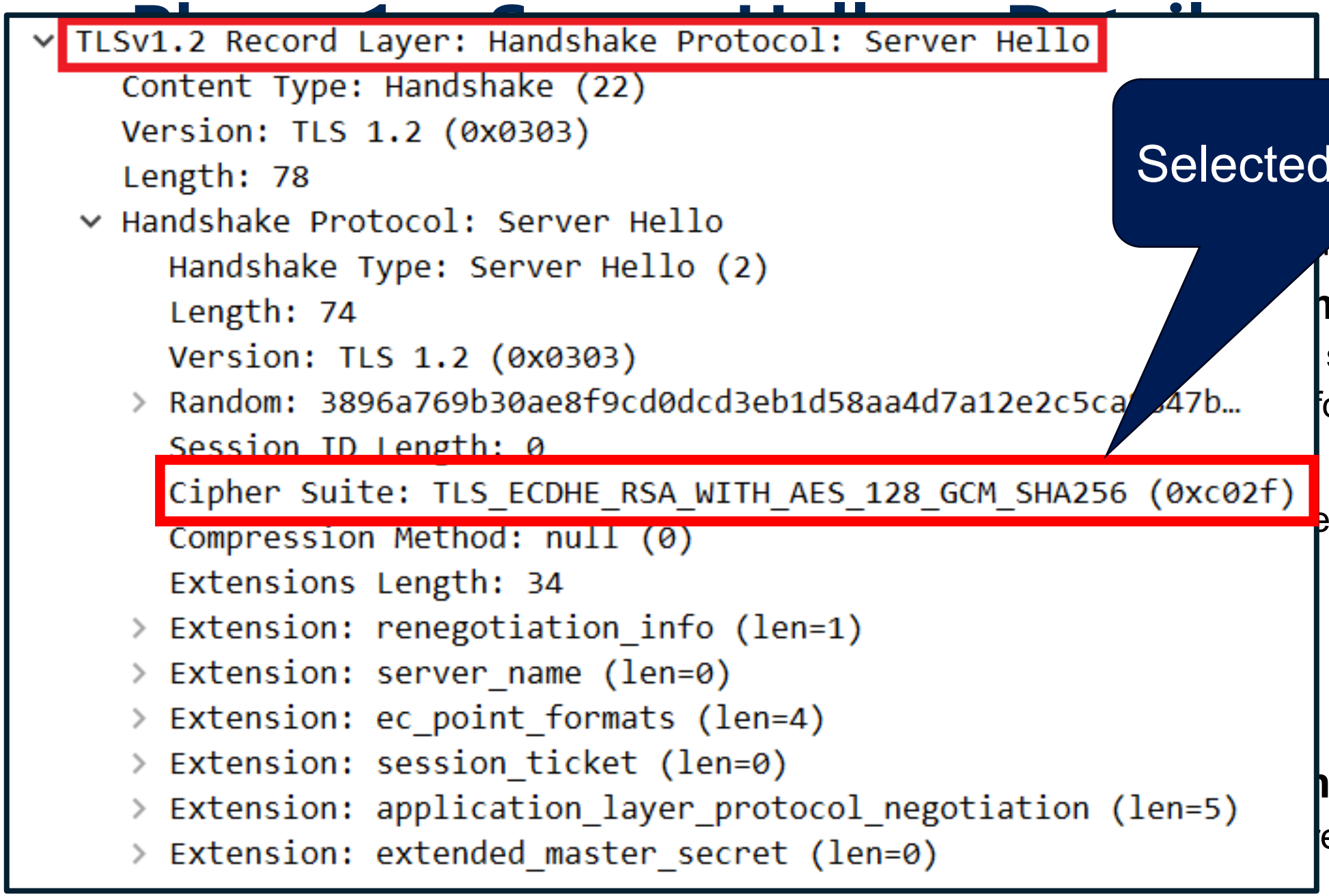
Phase 1 – Server Hello – Details

Client Hello – Details

- **Version**
  - Highest protocol version supported by the client
- **Client random number**
  - Random 32 bit time stamp + 28 random bytes
  - It will be used later for key generation
- **Session ID**
  - 0: establish new connection on new session
  - Non-zero: resume an old session
- **Cipher suite**
  - Set of cryptographic algorithms supported by the client
- **Compression methods**
  - Sequence of compression methods

Server Hello – Details

- **Version**
  - Highest common version
- **Server random number**
  - Random 32 bit time stamp + 28 random bytes
  - It will be used later for key generation
- **Session ID**
  - New session ID if zero, new session ID otherwise
- **Cipher suite**
  - The selected cipher suite
- **Compression methods**
  - The selected compression technique



Selected cipher suite

mber

stamp + 28 random bytes

for key generation

ero, new session ID

suite

nods

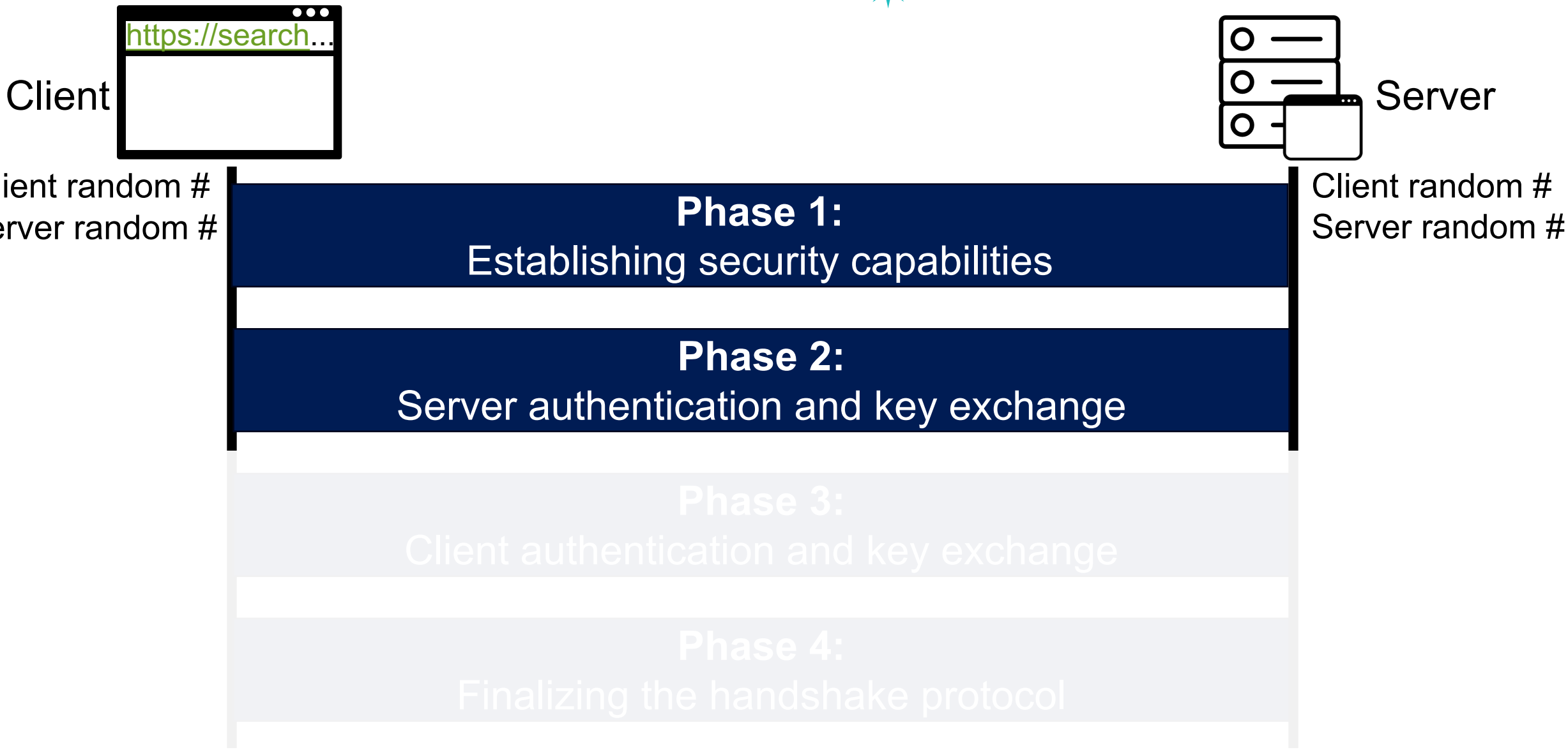
ession technique

Phase 1: Establishing Security Capabilities

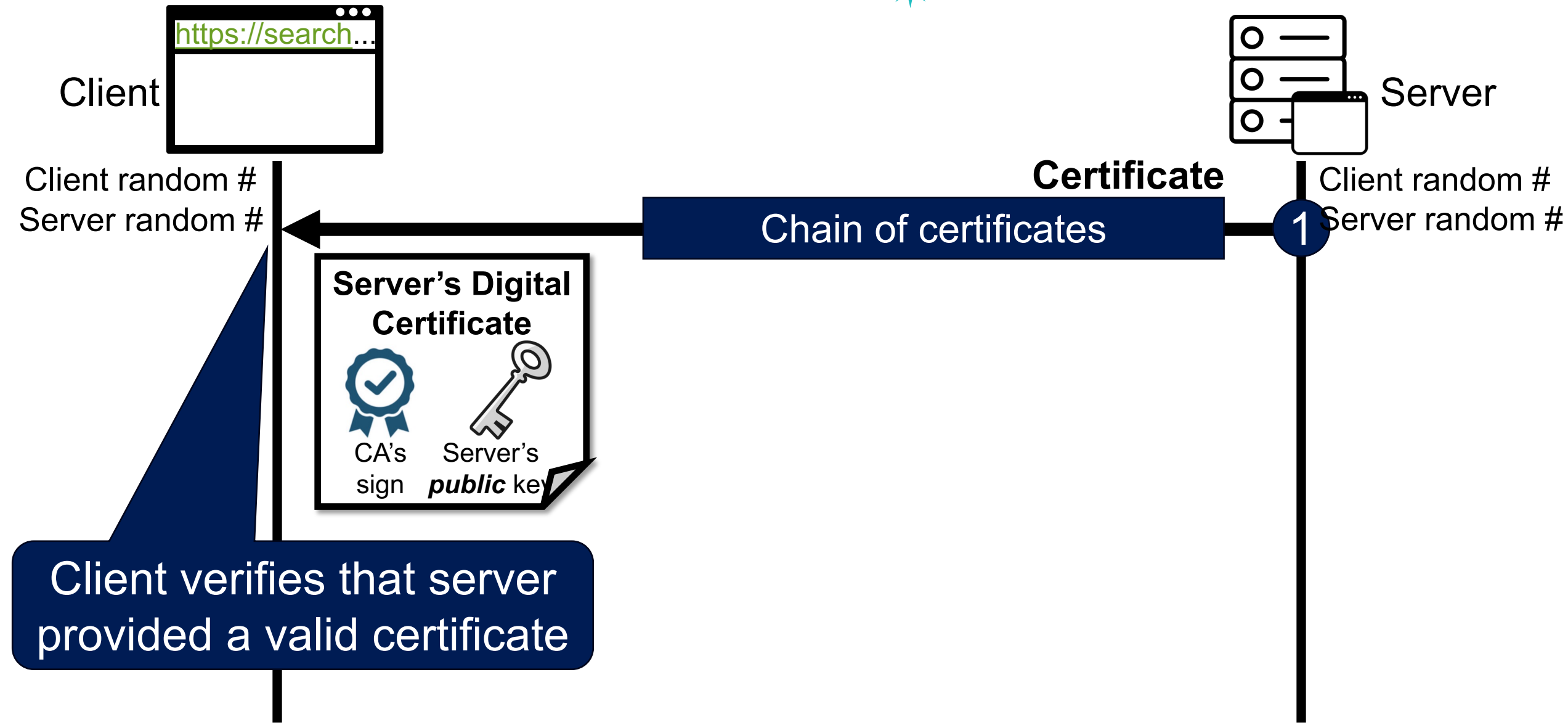
After Phase 1, the client and server know the followings:

- The version of SSL/TLS
- The algorithms for key exchange, hash, and encryption
- The compression method
- The two random numbers for key generation

Phase 2: Server Auth. and Key Exchange

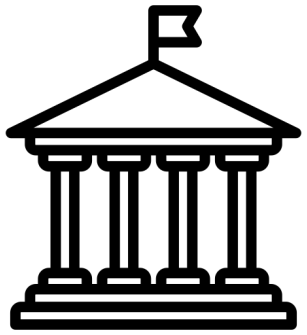


Phase 2: Server Auth. and Key Exchange



Recap: Hash-based Digital Signature in PKI

Signing



Certificate Authority (CA)

Digital Certificate

- ✓ **Subject**: Server
- ✓ **Expires**: 11/25/2034
- ✓ **Bob's public key**: ADFECDBBF…

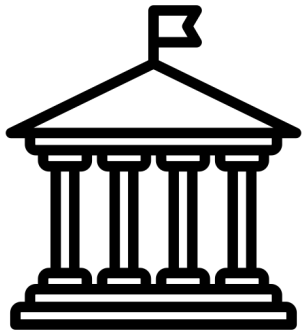
Hash function

0101000010..

Encrypt with CA's ***private key***

Recap: Hash-based Digital Signature in PKI

Signing



Certificate Authority (CA)

Digital Certificate

- ✓ **Subject**: Server
- ✓ **Expires**: 11/25/2024
- ✓ **Bob's public key**: ADFECDBBF…

Hash function

0101000010..

Encrypt with CA's ***private key***

Append

Recap: Hash-based Digital Signature in PKI

Verification

Alice

Digital Certificate

- ✓ **Subject**: Server
- ✓ **Expires**: 11/25/2034
- ✓ **Bob's public key**: ADFECDBBF…

Recap: Hash-based Digital Signature in PKI

Verification

Alice

Digital Certificate

- ✓ **Subject**: Server
- ✓ **Expires**: 11/25/2034
- ✓ **Bob's public key**: ADFECDBBF…

Hash function

CA's sign

Decrypt with CA's ***public key***

0101000010..

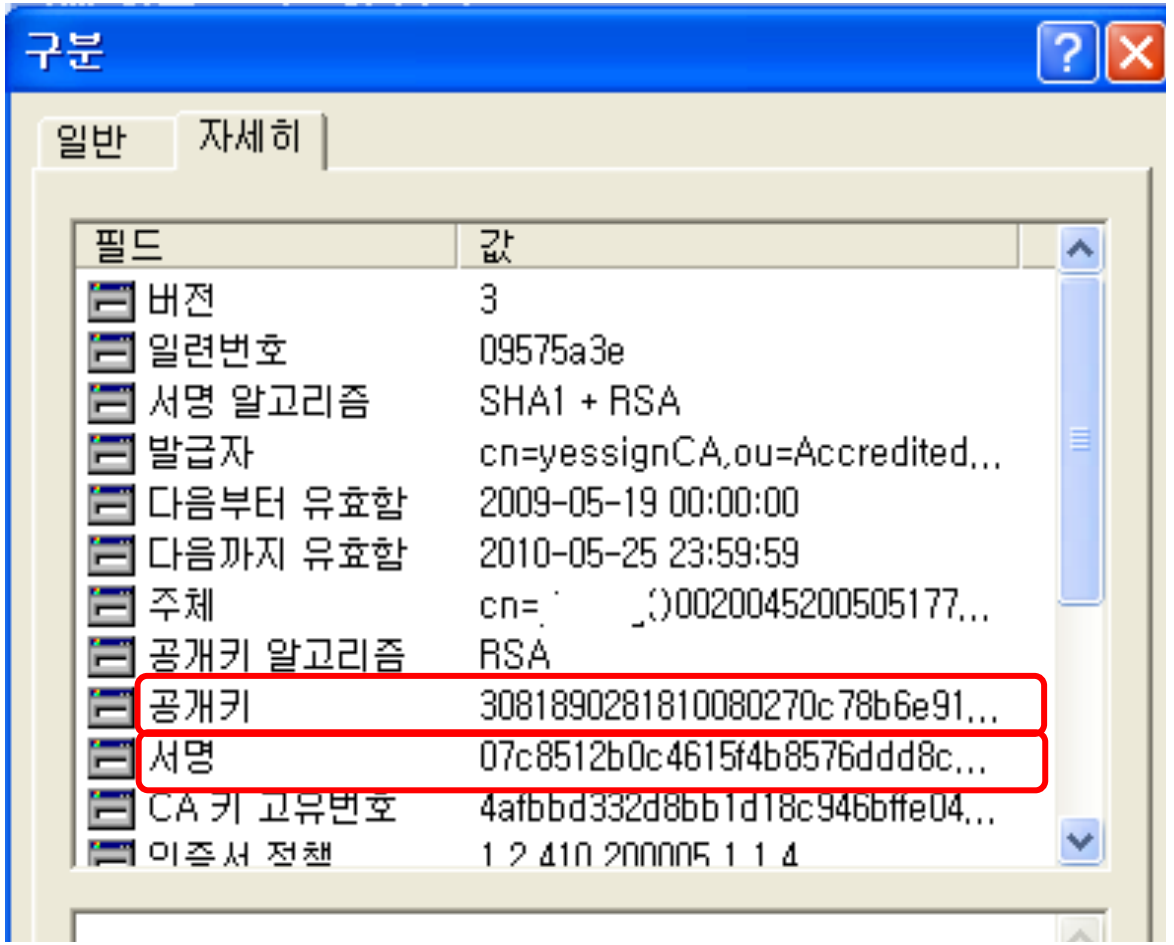
?

0101000010..

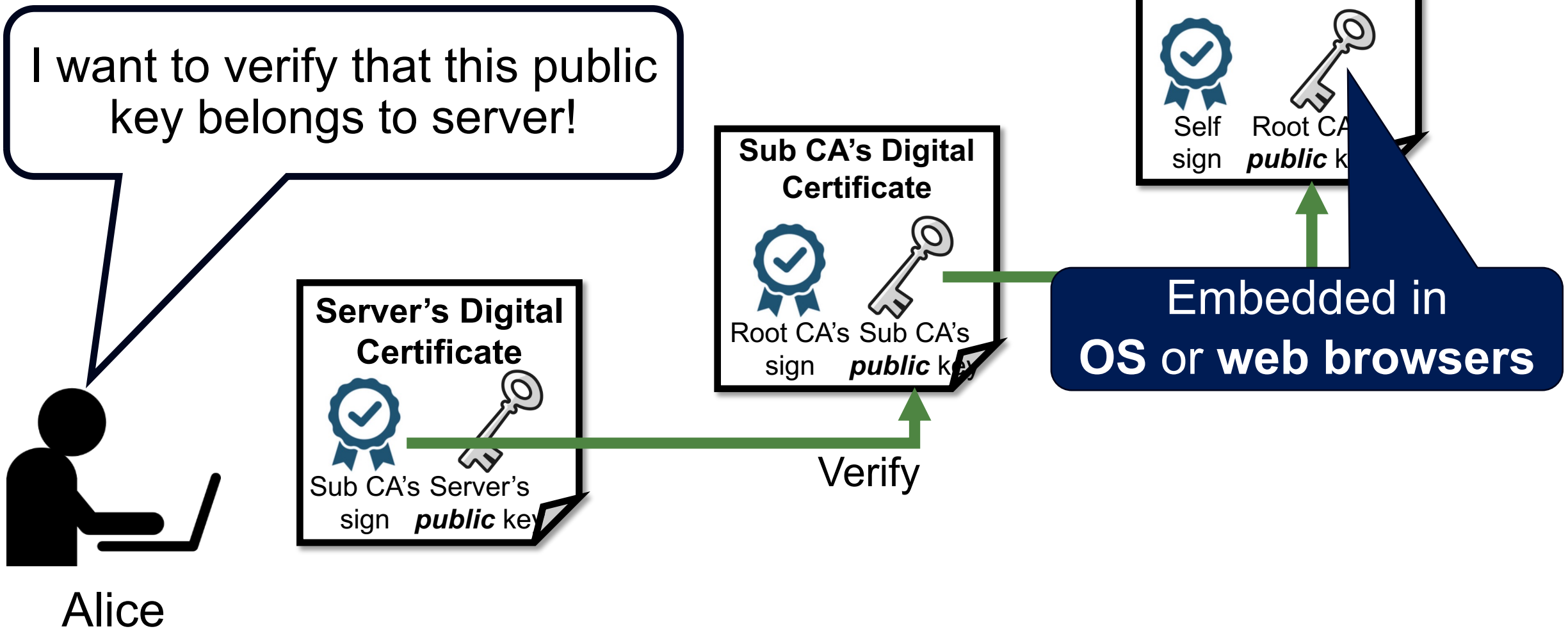
Authentication: Confirm Server's public key

Recap: X.509 Certificate

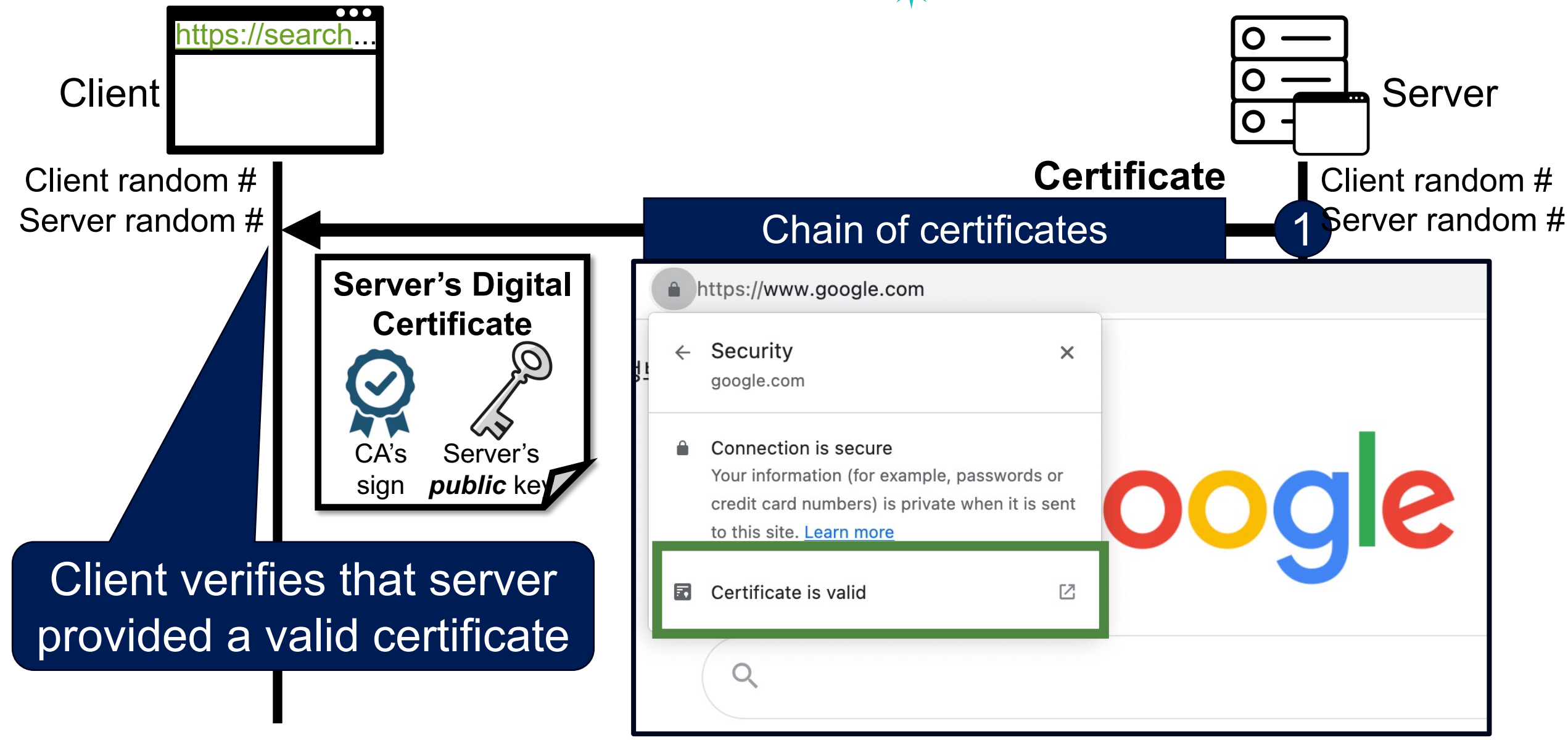
Version
Serial Number
Signature Algorithm Identifier
Issuer Name
Validity Period
Subject Name
Public Key Information
Issuer Unique ID
Subject Unique ID
Extensions



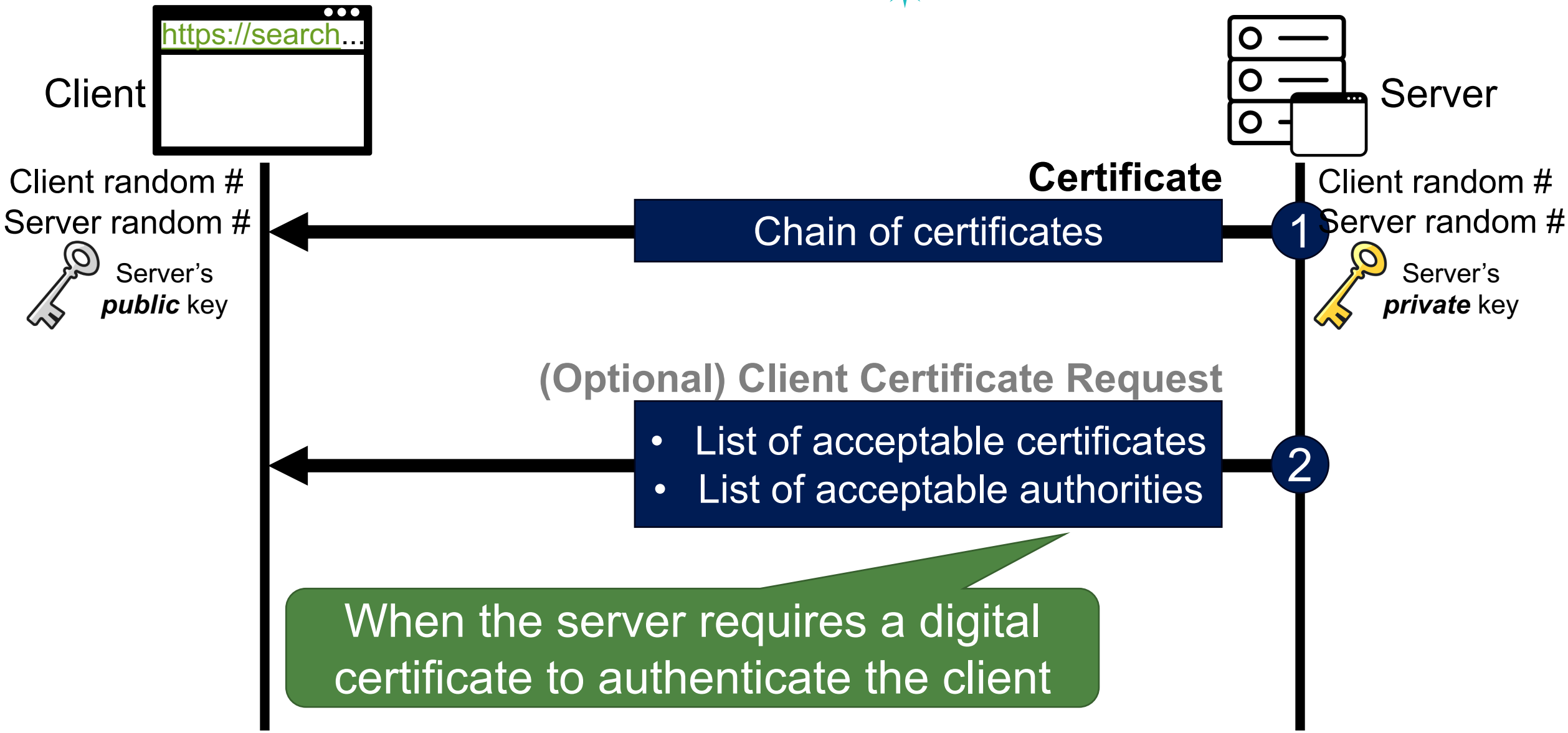
Recap: Chain of Trust



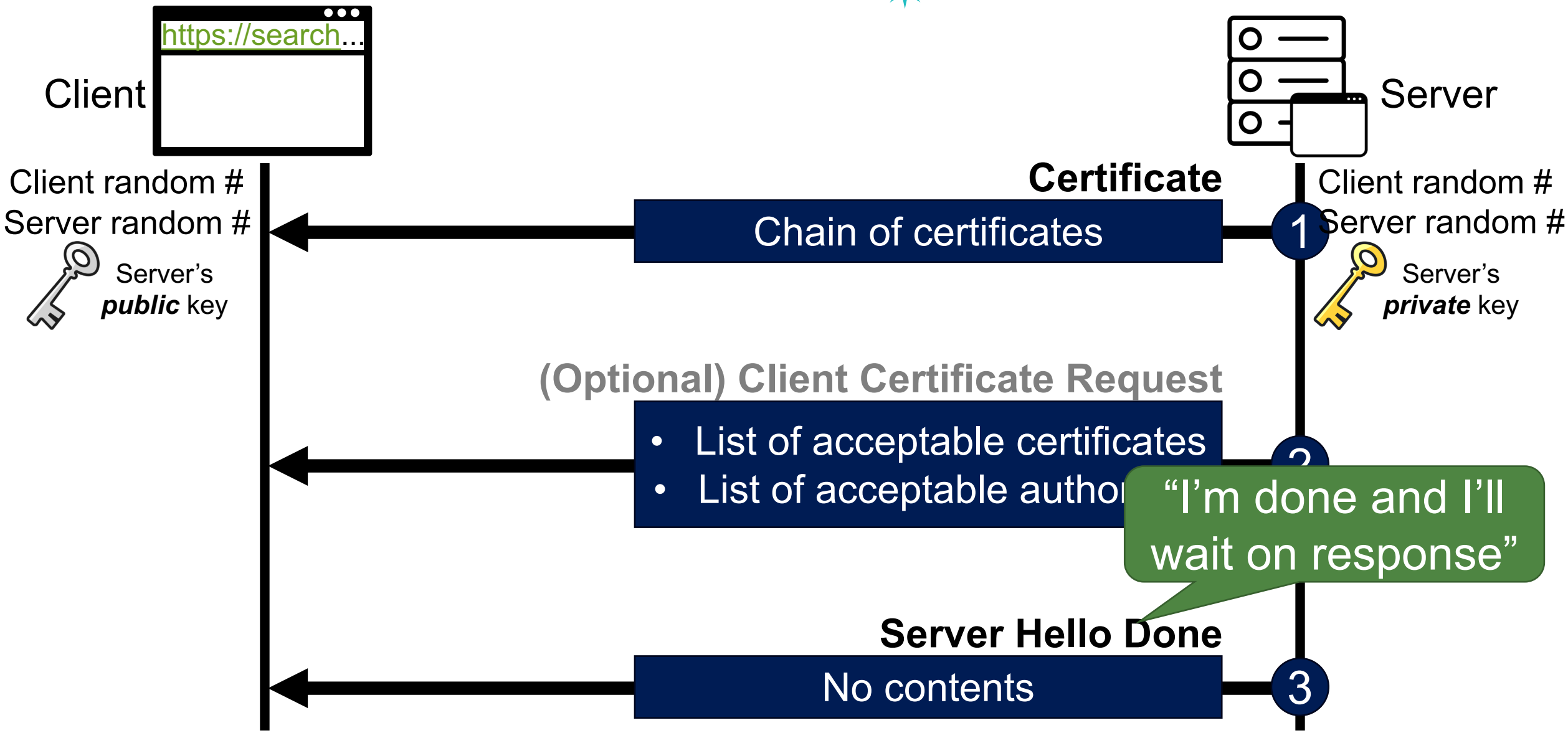
Phase 2: Server Auth. and Key Exchange



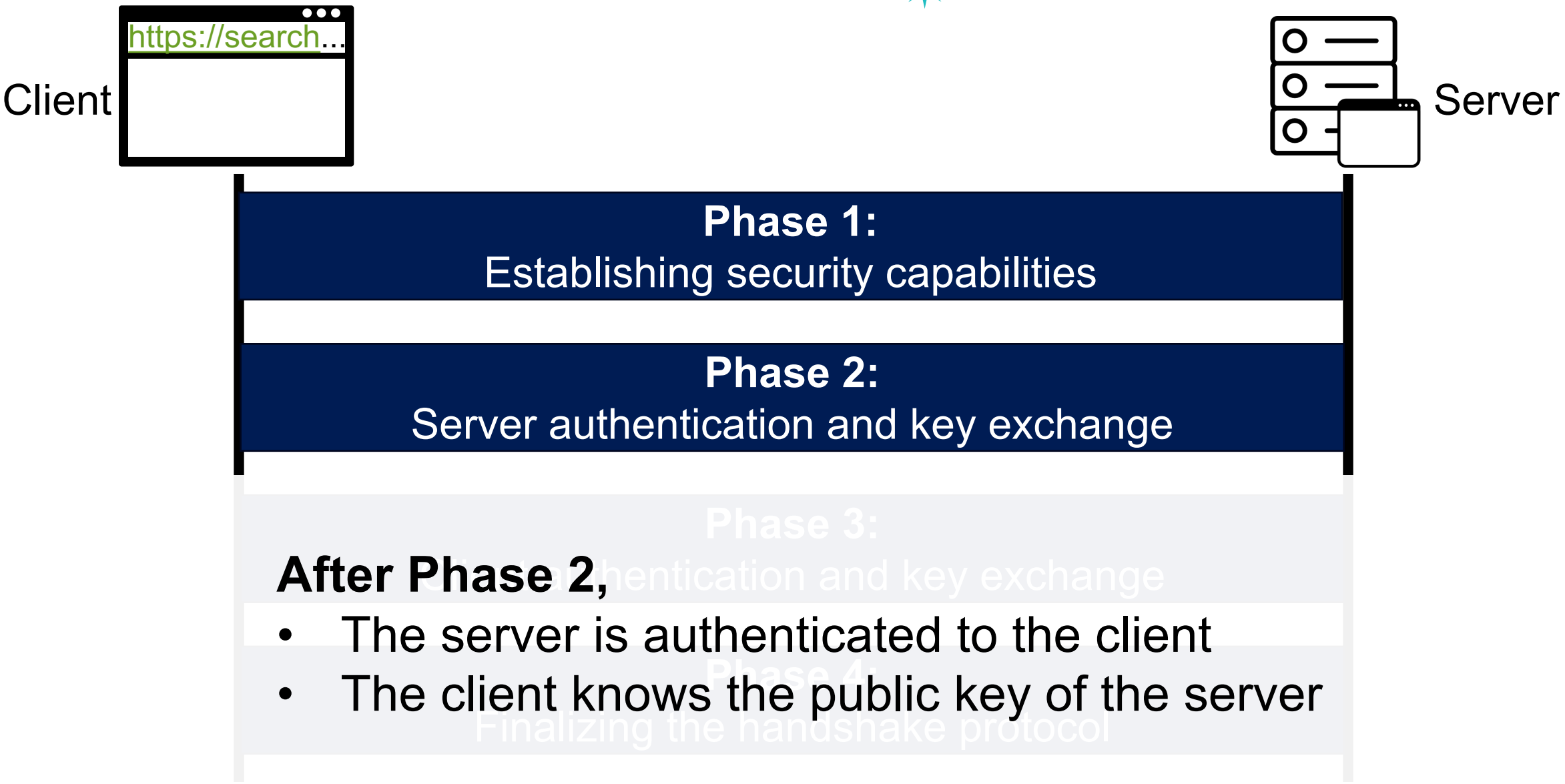
Phase 2: Server Auth. and Key Exchange



Phase 2: Server Auth. and Key Exchange



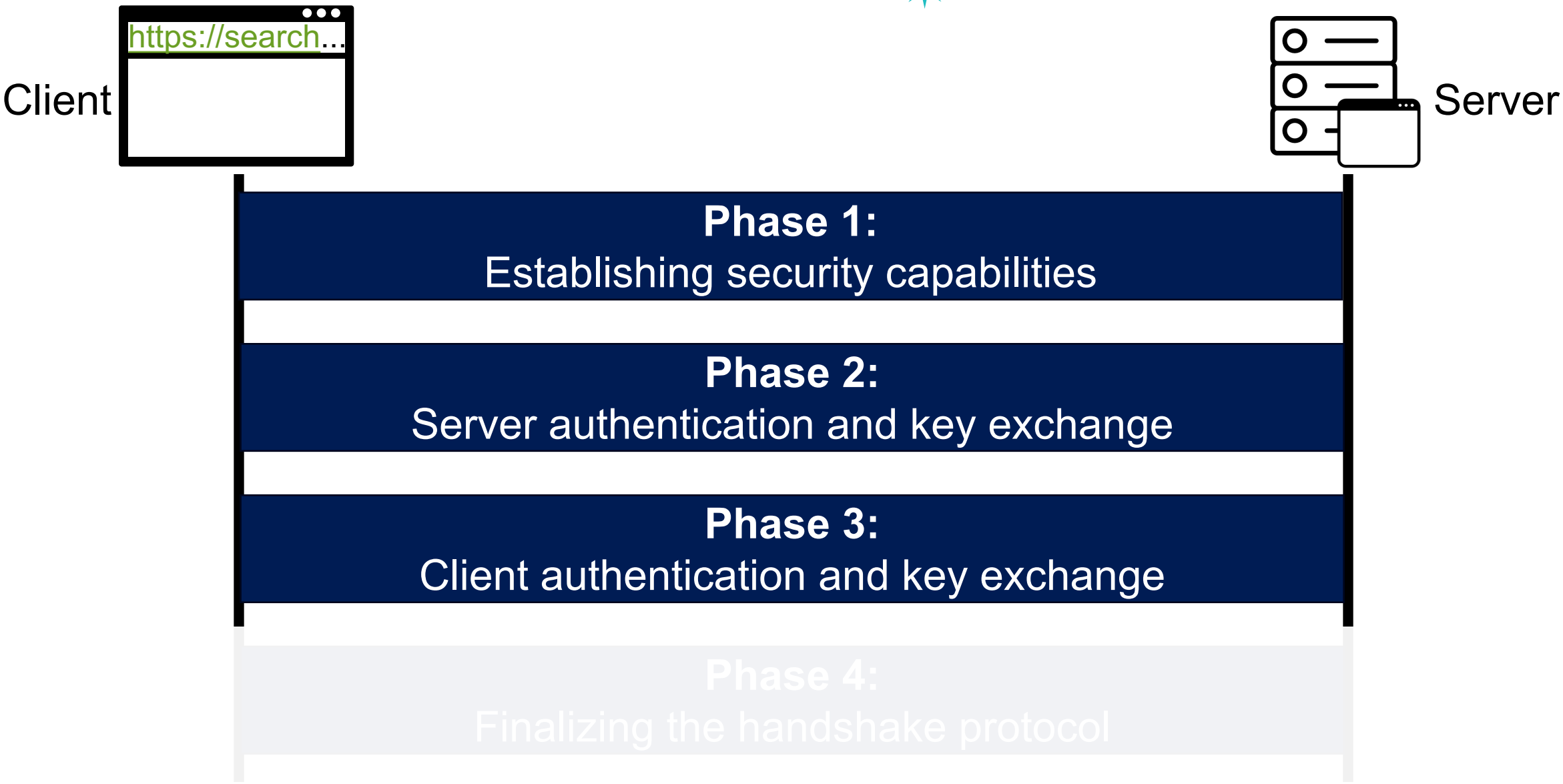
Phase 1: Establishing Security Capabilities



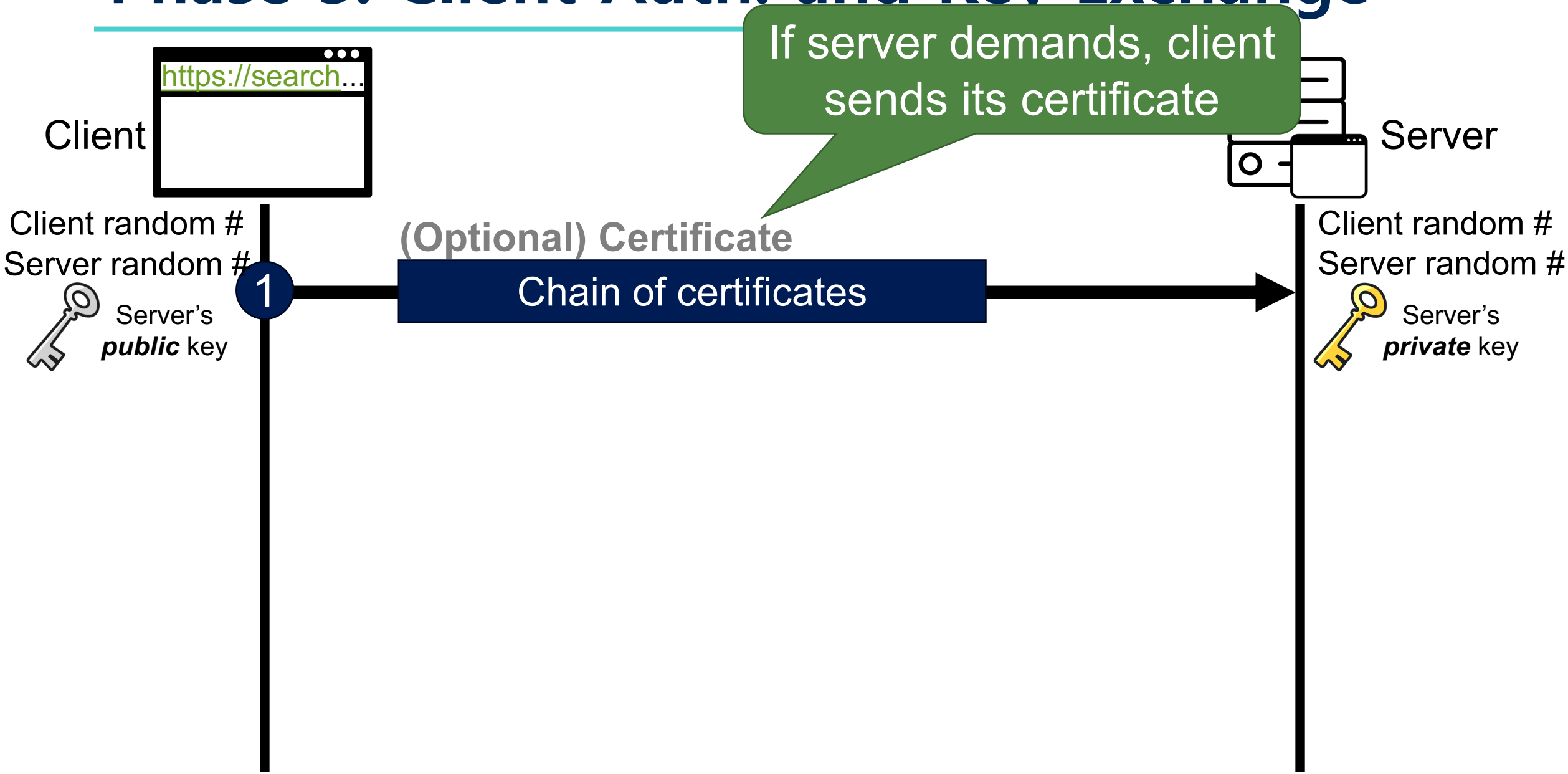
After Phase 2,

- The server is authenticated to the client
- The client knows the public key of the server

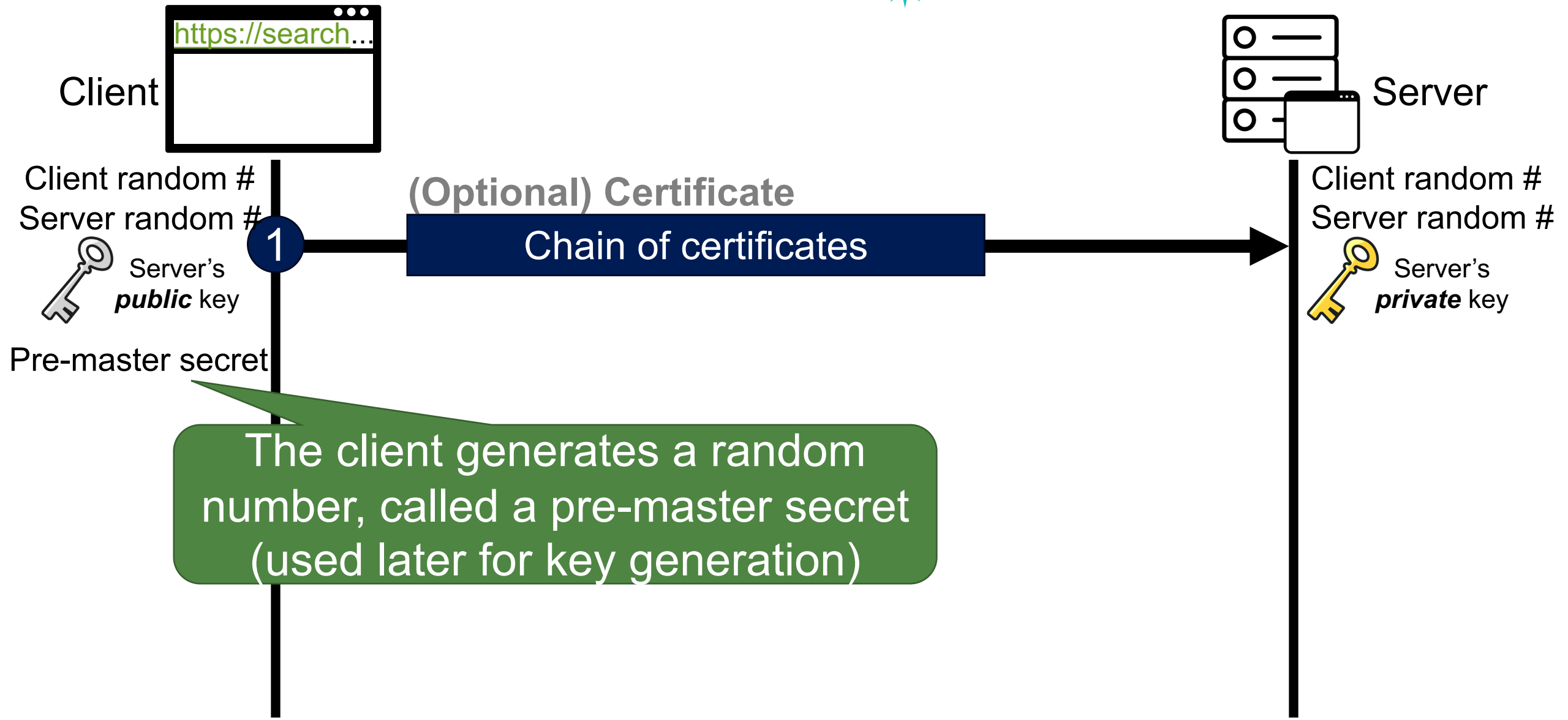
Phase 3: Client Auth. and Key Exchange



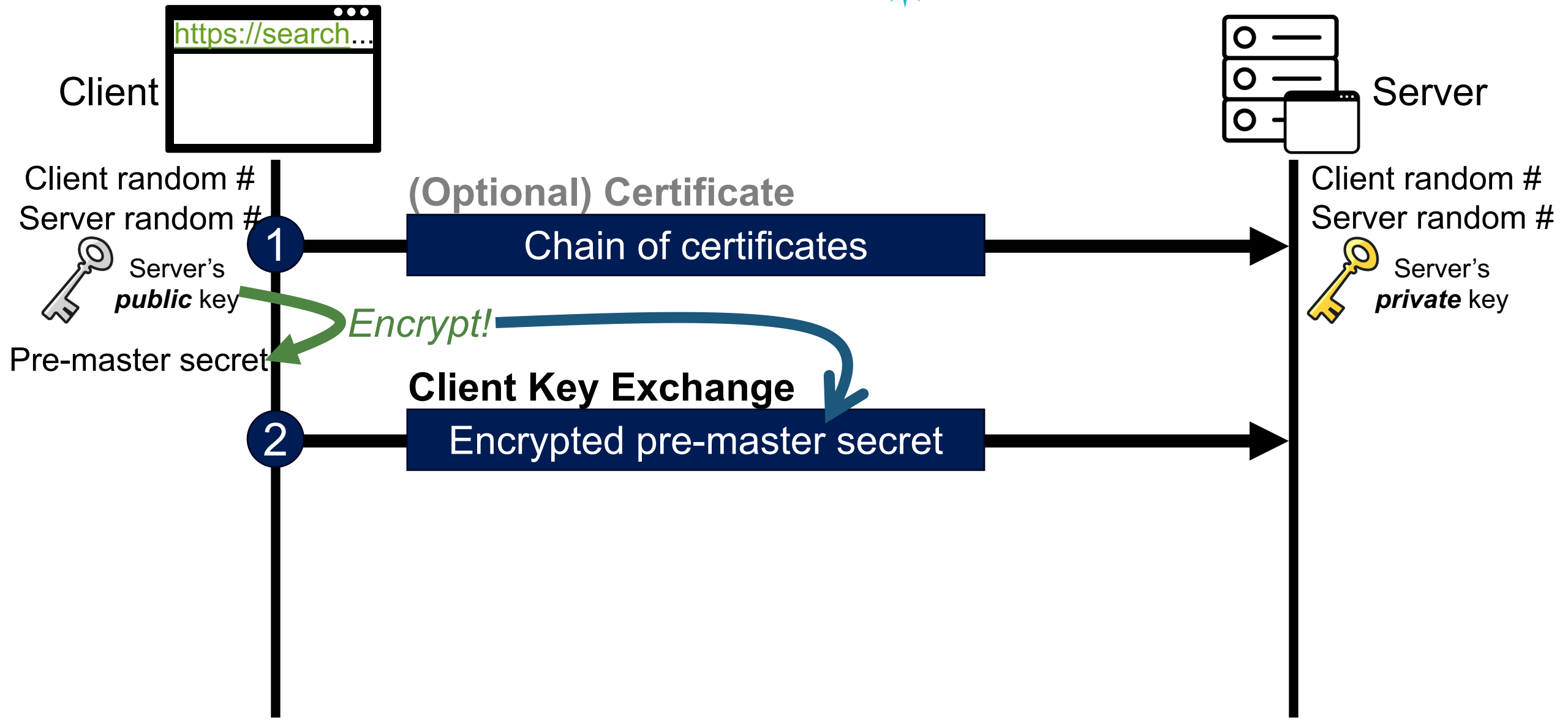
Phase 3: Client Auth. and Key Exchange



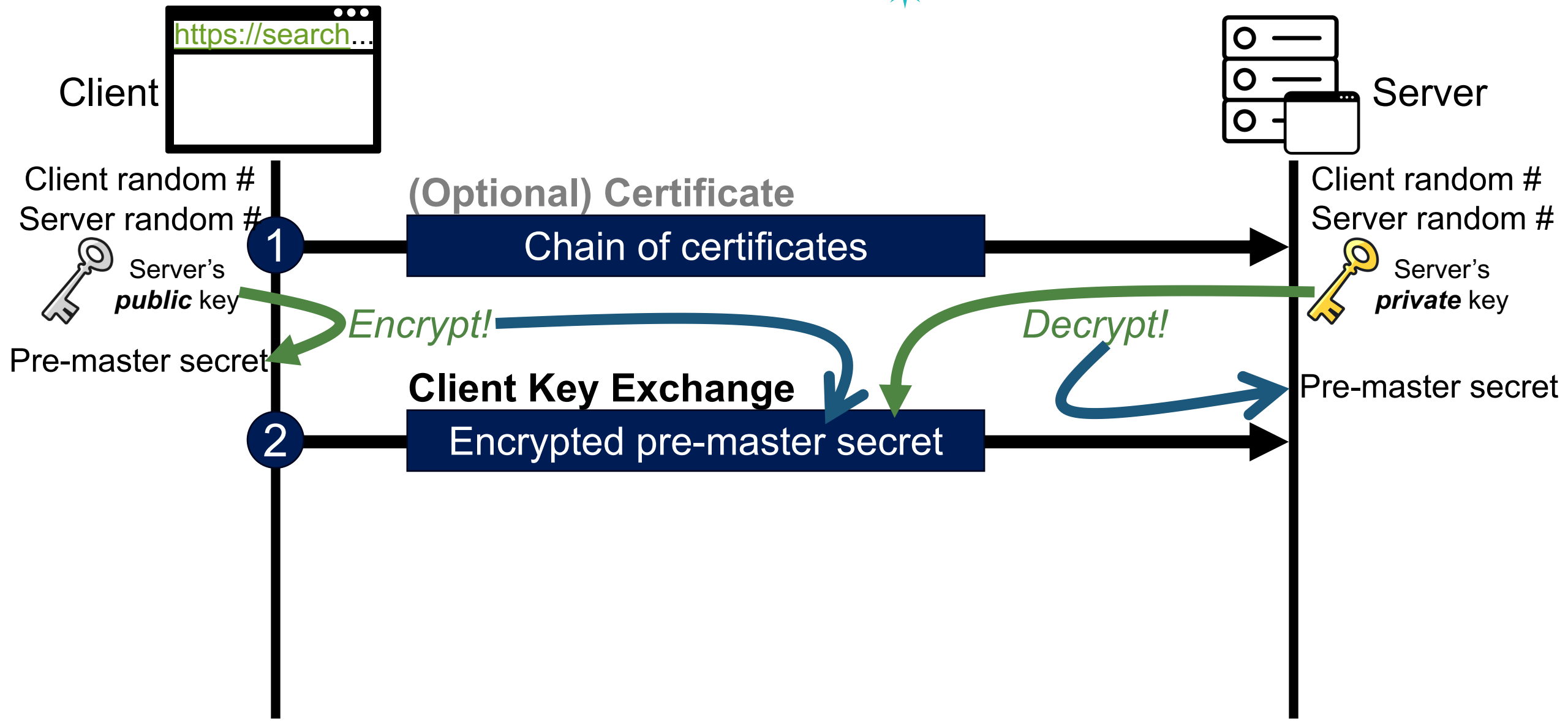
Phase 3: Client Auth. and Key Exchange



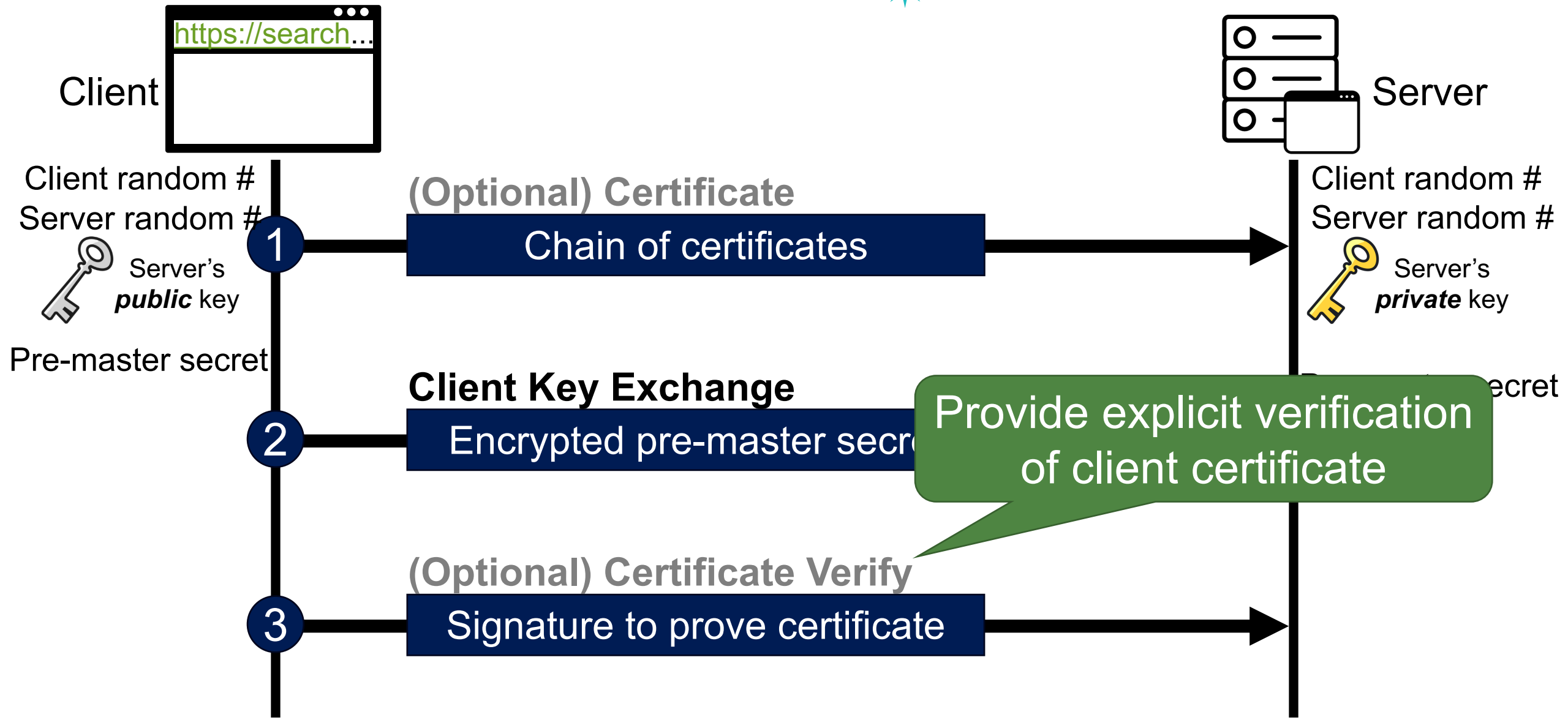
Phase 3: Client Auth. and Key Exchange



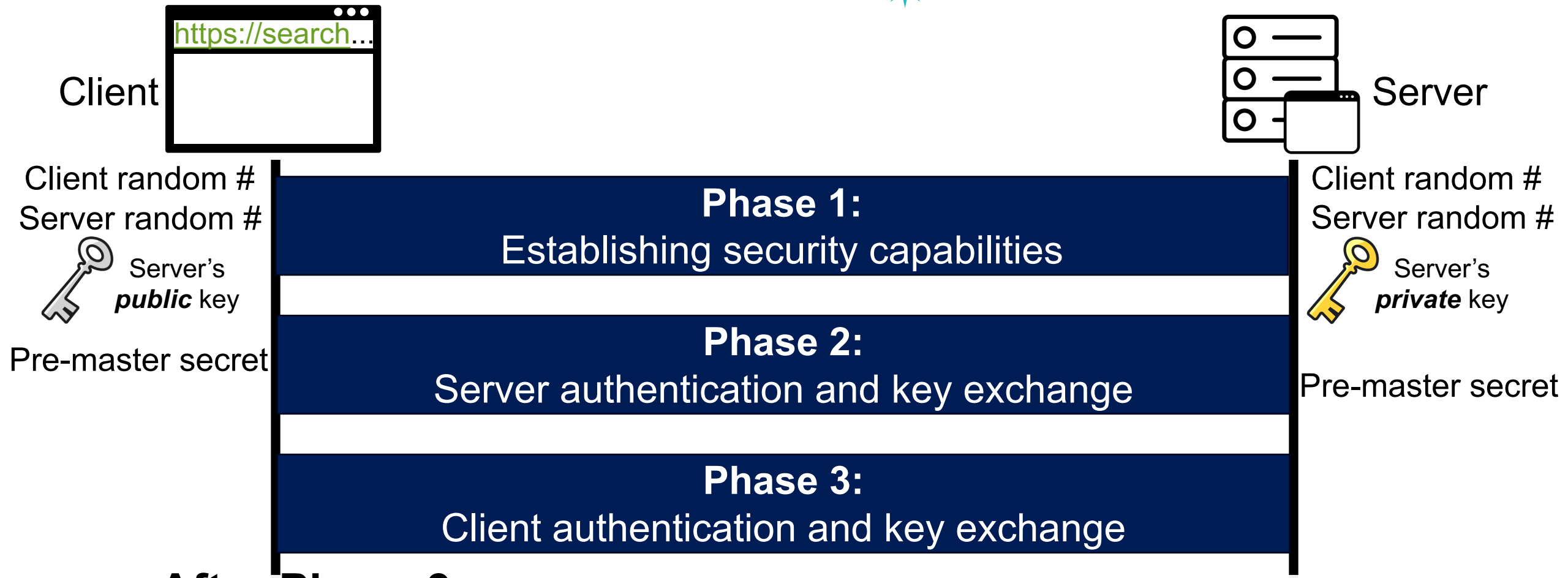
Phase 3: Client Auth. and Key Exchange



Phase 3: Client Auth. and Key Exchange



Phase 3: Client Auth. and Key Exchange

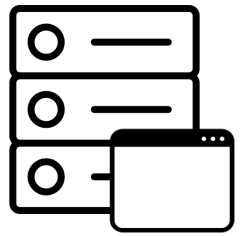


After Phase 3,

- (Optional) The client is authenticated for the server
- Both the client and the server know the pre-master secret

Phase 3: Client Auth. and Key Exchange

Client



Server

Client random #
Server random #

Server's ***public*** key

Pre-master secret

Before move on Phase 4, let's make symmetric key

Why do we need a symmetric key
even though we already have asymmetric key?

Client random #
Server random #

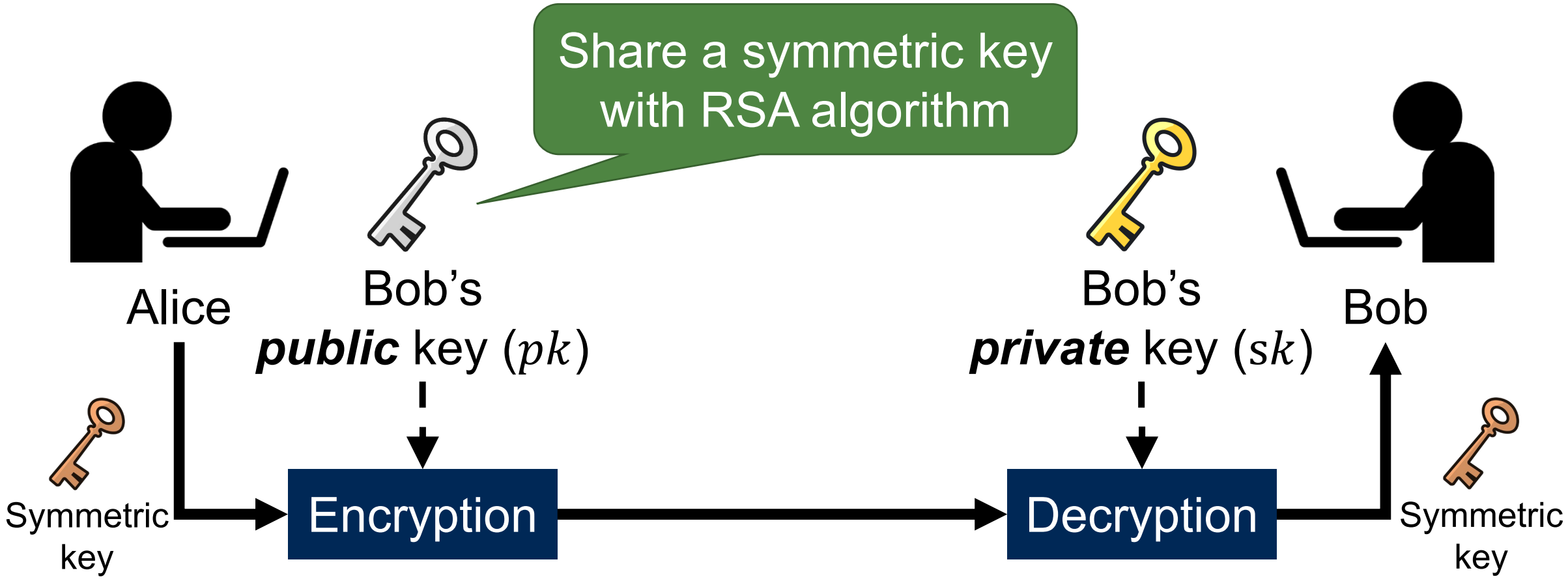
Server's ***private*** key

Pre-master secret

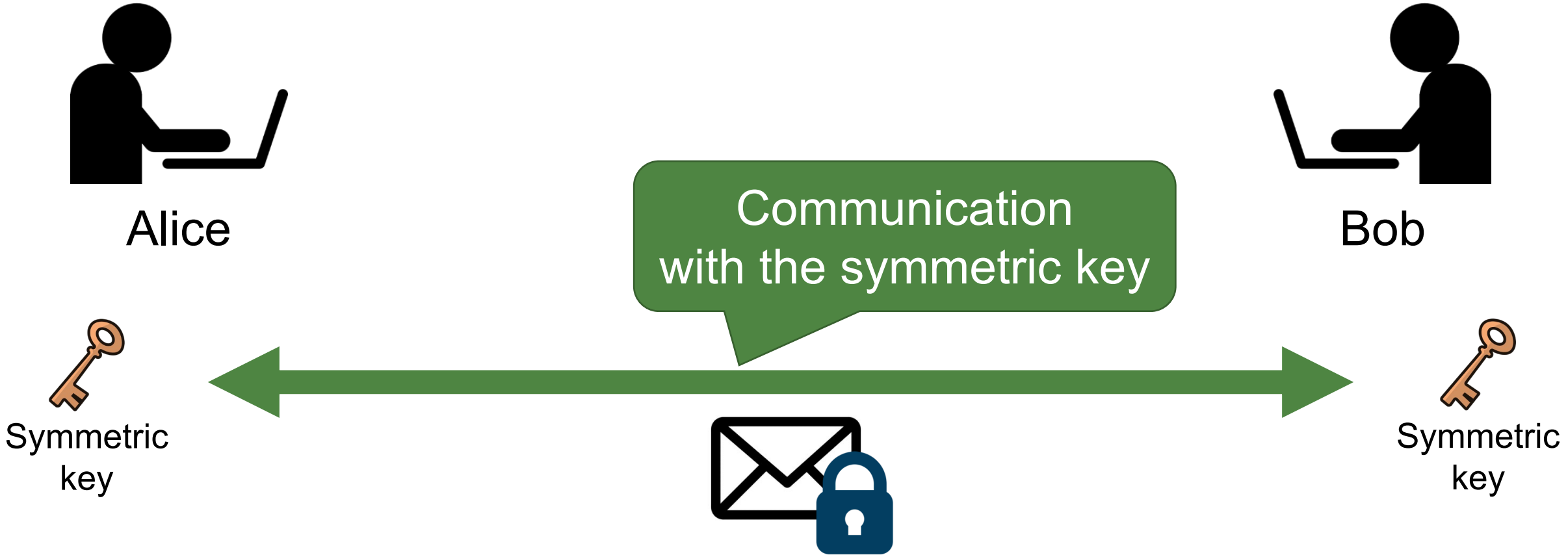
Recap: Asymmetric-key Cryptography

- Pros
  - No need to share a secret
  - Enable multiple senders to communicate privately with a single receiver
  - More applications: Digital sign
- Cons
  - **Slower in general**: due to the larger key
    - Roughly 2-3 orders of magnitude slower

Recap: Combination of Two Schemes

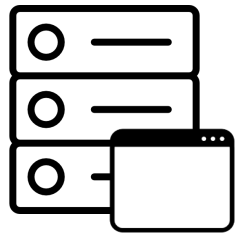


Recap: Combination of Two Schemes



Phase 3: Client Auth. and Key Exchange

Client



Server

Client random #
Server random #

Server's ***public*** key

Pre-master secret

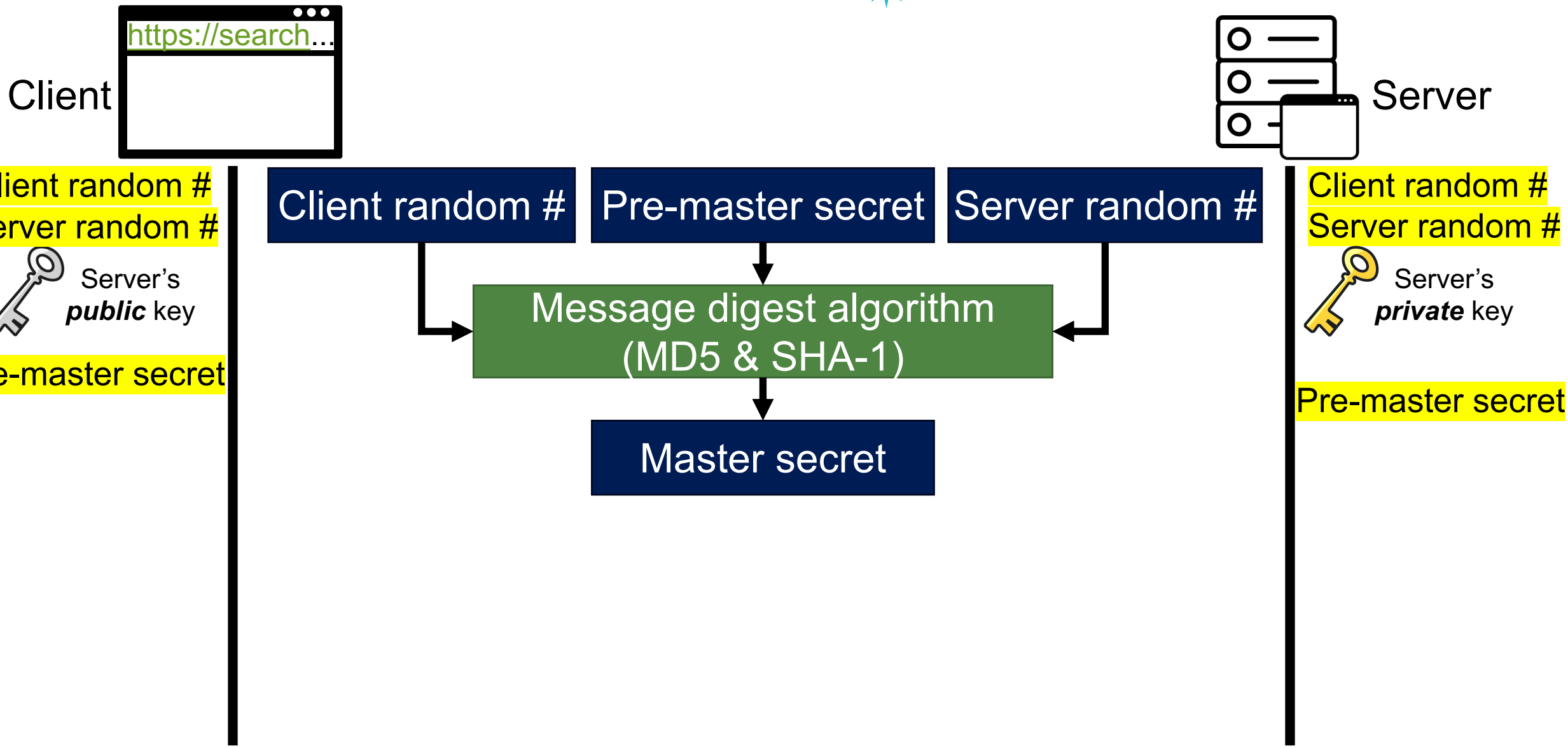
Before move on Phase 4, let's make symmetric key

Client random #
Server random #

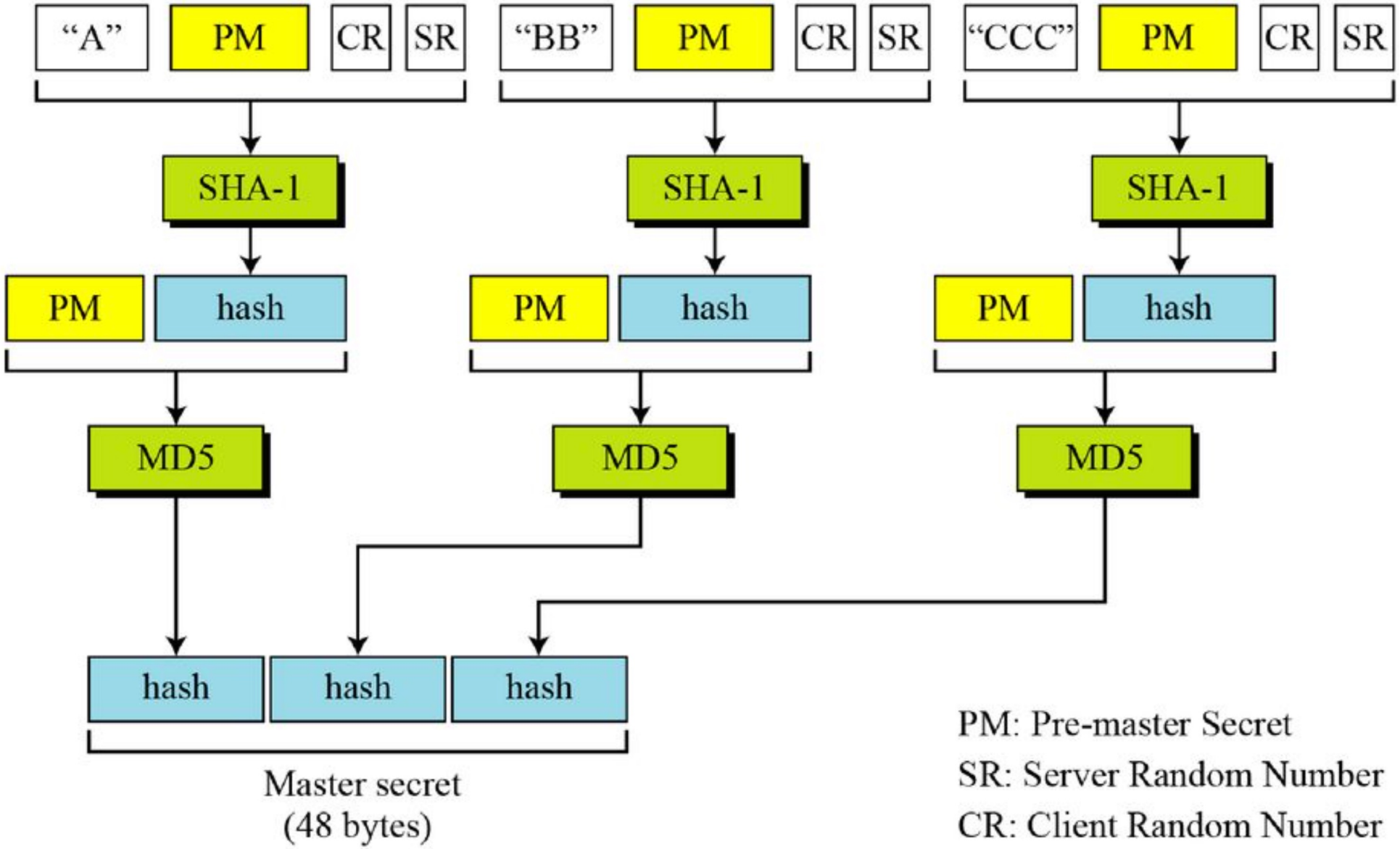
Server's ***private*** key

Pre-master secret

Calculation of Master Secret

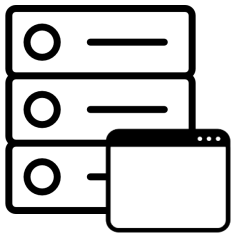


Calculation of Master Secret



Calculation of Symmetric Key

Client



Server

Client random #
Server random #

Server's ***public*** key

Pre-master secret

Client random # | Pre-master secret | Server random #

Message digest algorithm (MD5 & SHA-1)

Master secret

Message digest algorithm (MD5 & SHA-1)

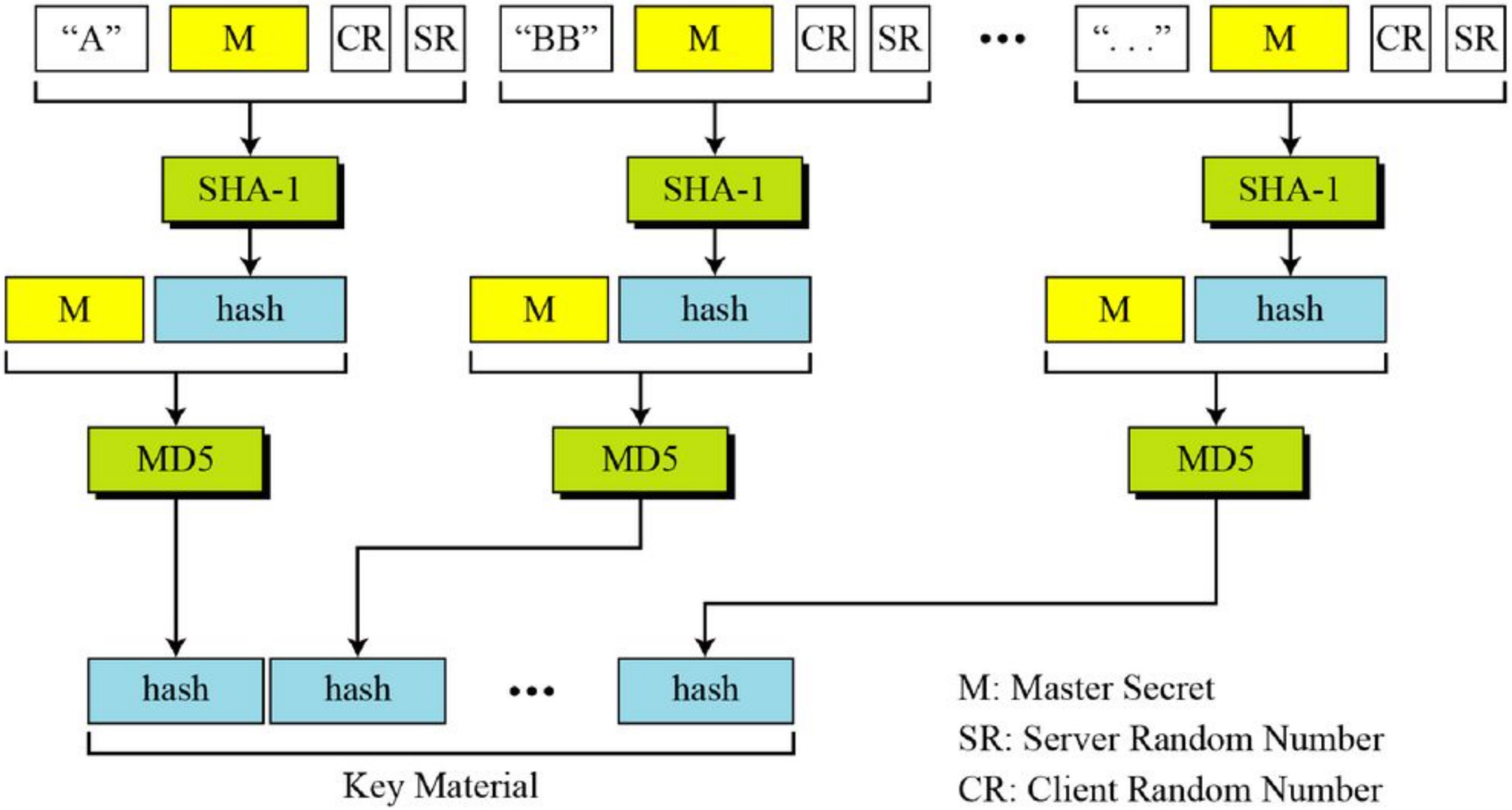
Symmetric key

Client random #
Server random #

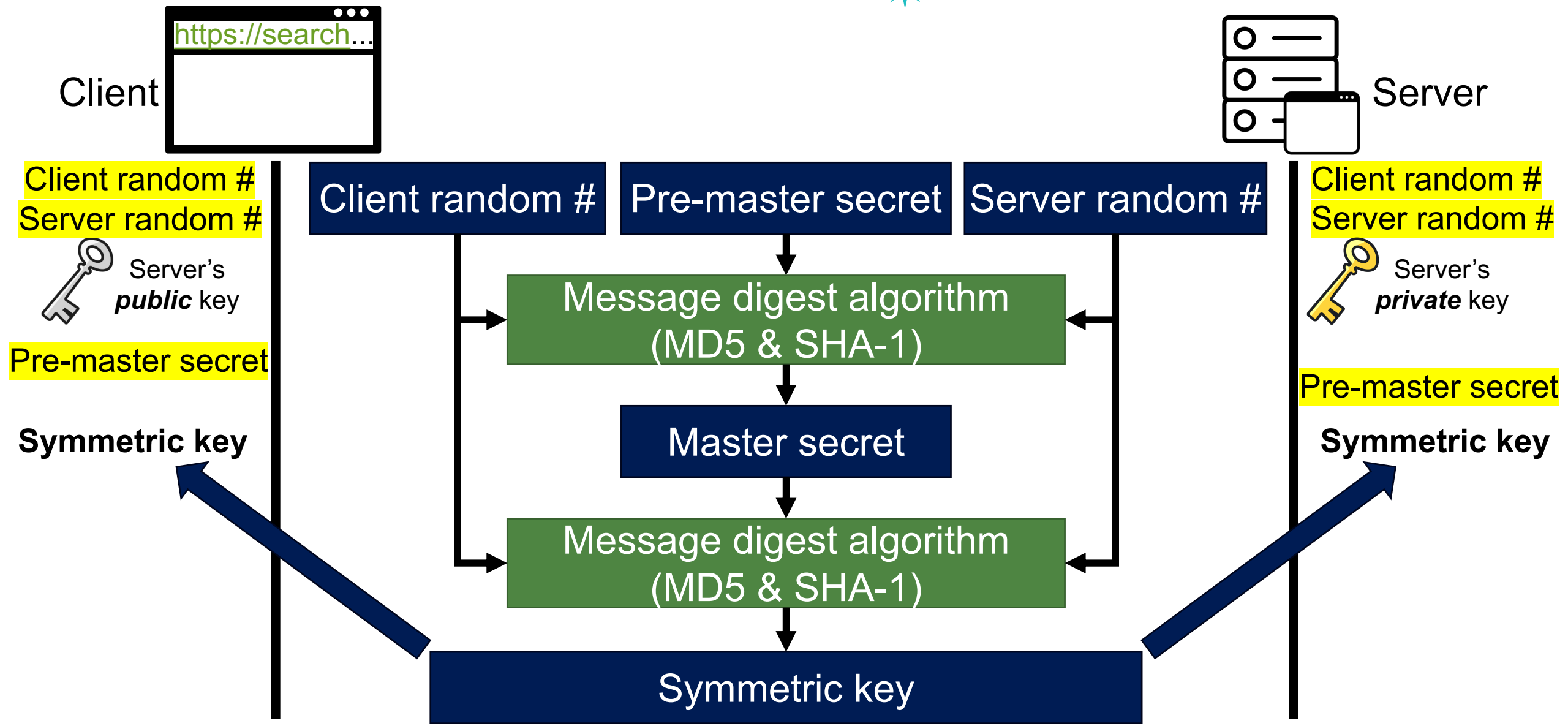
Server's ***private*** key

Pre-master secret

Calculation of Symmetric Key

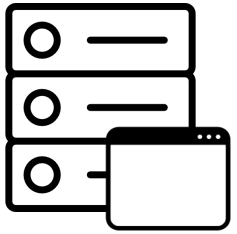


Calculation of Symmetric Key



Calculation of Symmetric Key

Client



Server

Client random #
Server random #

Server's ***public*** key

Pre-master secret

Symmetric key

Client random #

Pre-master secret

Server random #

Message digest algorithm (MD5 & SHA-1)

Master secret

Message digest algorithm (MD5 & SHA-1)

Client auth. key	Server auth. key	Client enc. key	Server enc. key	Client IV	Server IV

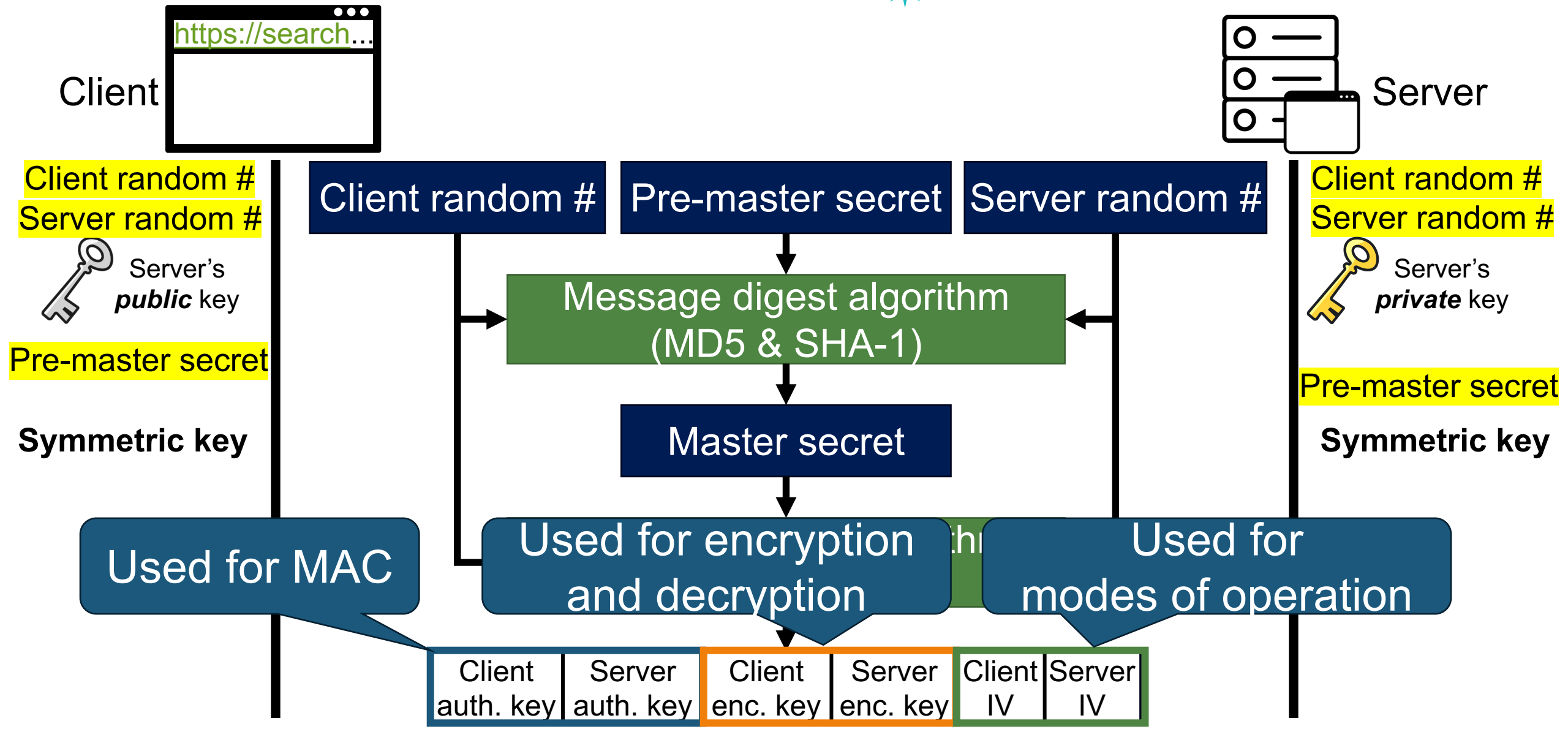
Client random #
Server random #

Server's ***private*** key

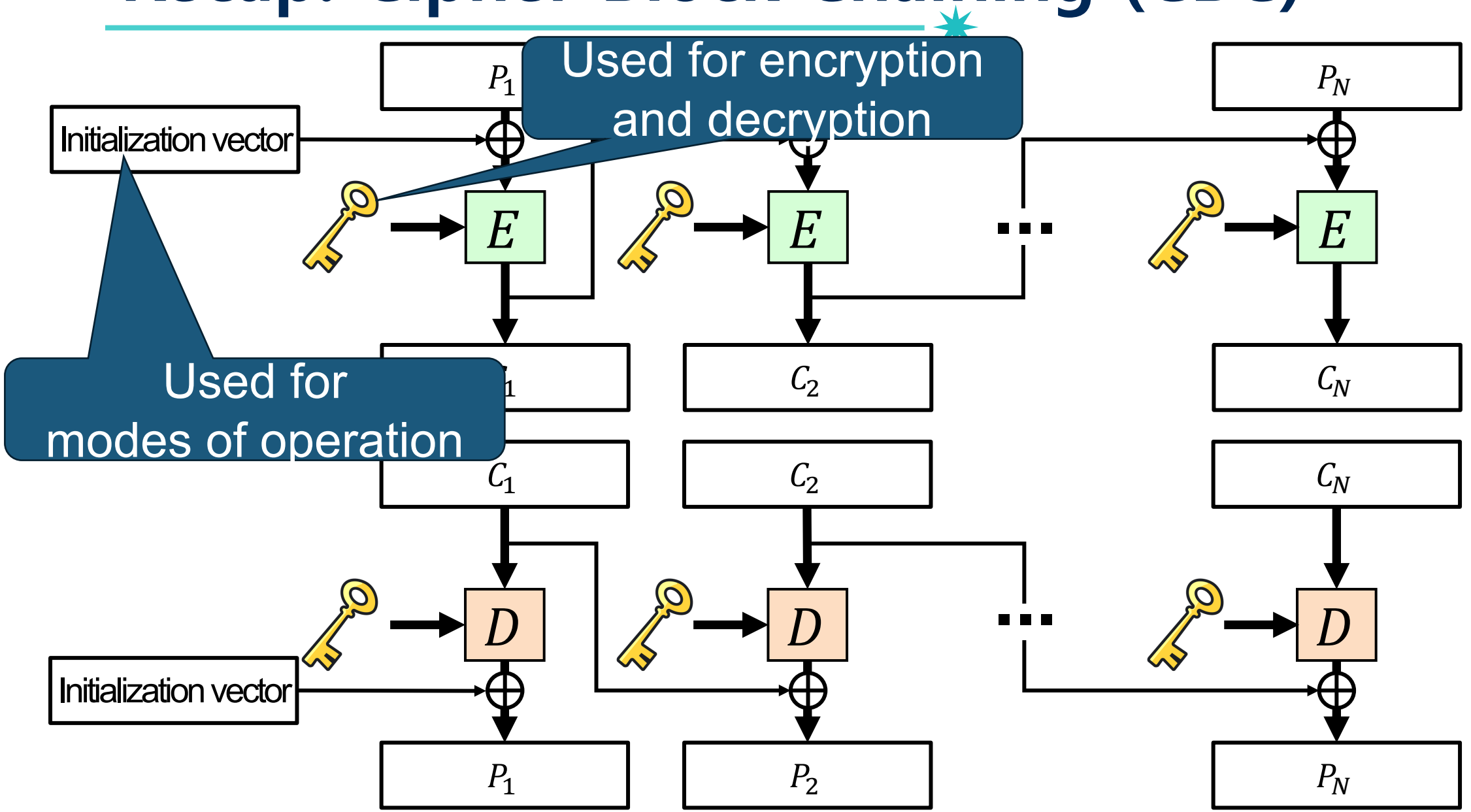
Pre-master secret

Symmetric key

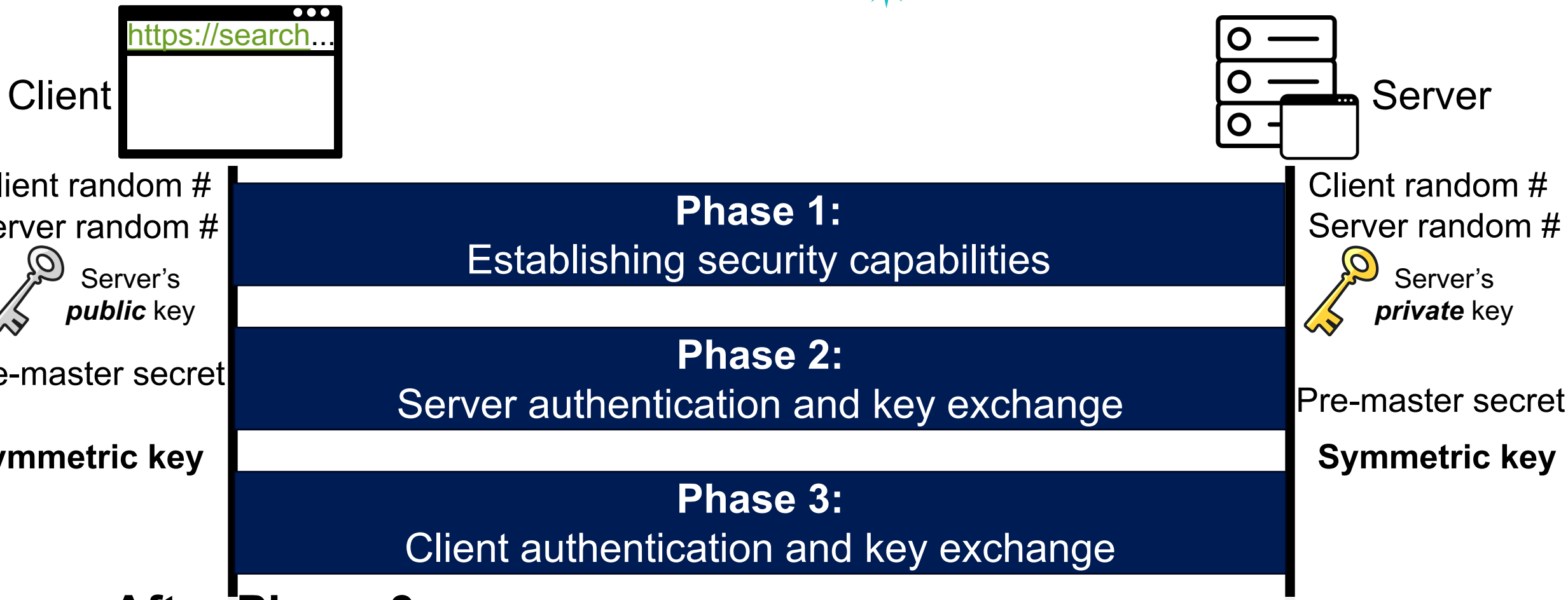
Calculation of Symmetric Key



Recap: Cipher Block Chaining (CBC)



Phase 3: Client Auth. and Key Exchange

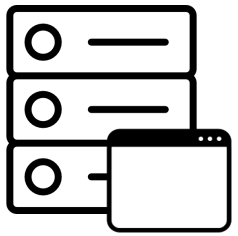


After Phase 3,

- (Optional) The client is authenticated for the server
- Both the client and the server know the pre-master secret

Phase 4: Finalizing the Handshake Protocol

Client



Server

Client random #
Server random #

Server's ***public*** key

Pre-master secret

Symmetric key

Phase 1:
Establishing security capabilities

Phase 2:
Server authentication and key exchange

Phase 3:
Client authentication and key exchange

Phase 4:
Finalizing the handshake protocol

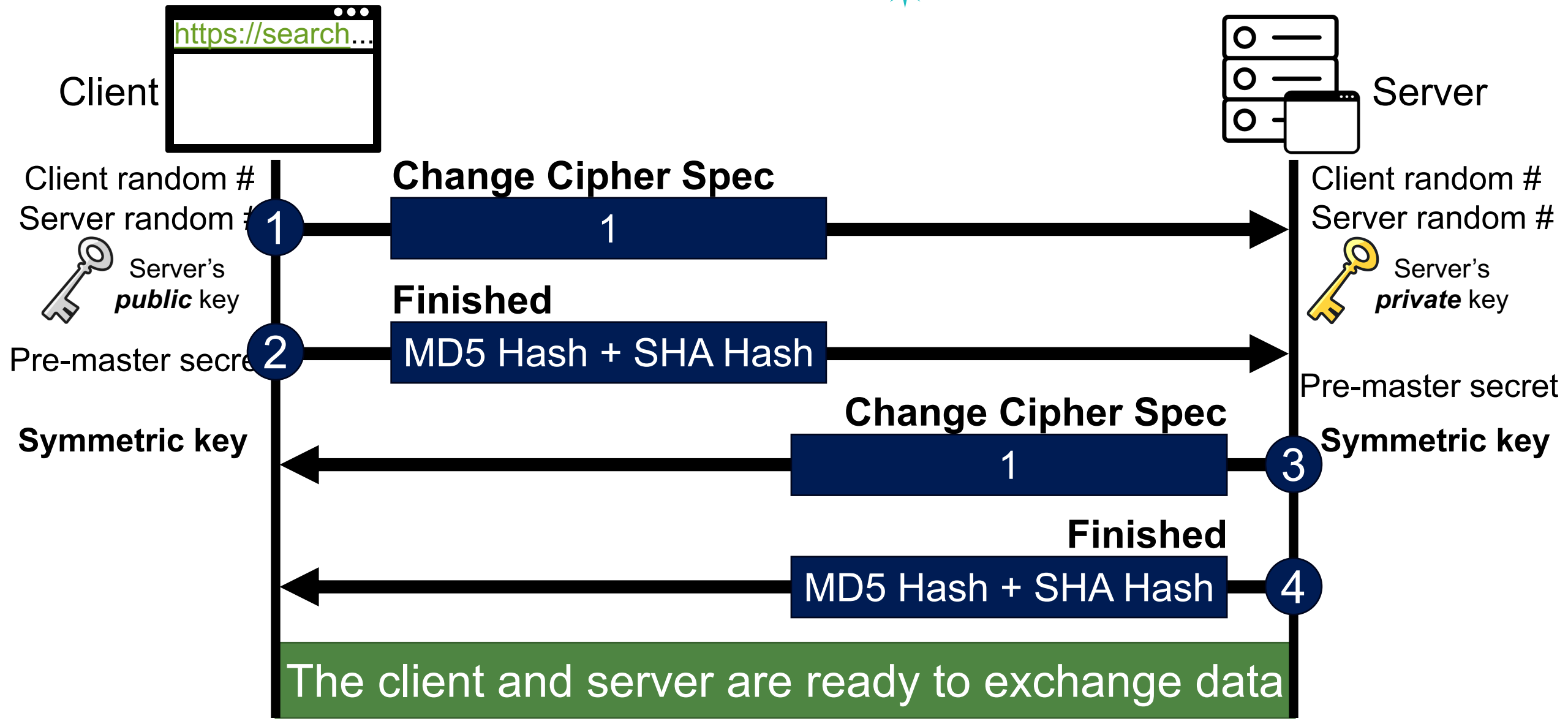
Client random #
Server random #

Server's ***private*** key

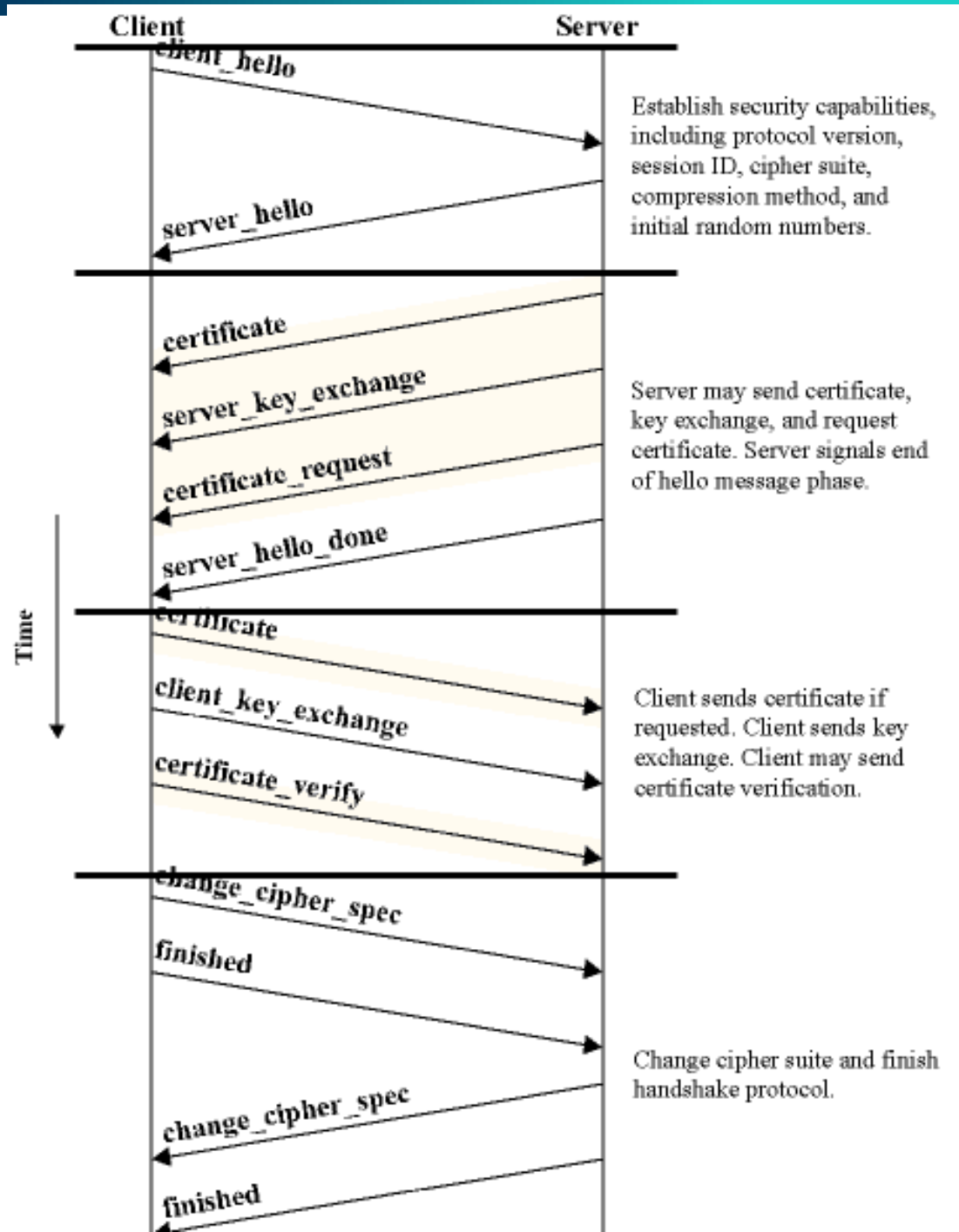
Pre-master secret

Symmetric key

Phase 4: Finalizing the Handshake Protocol



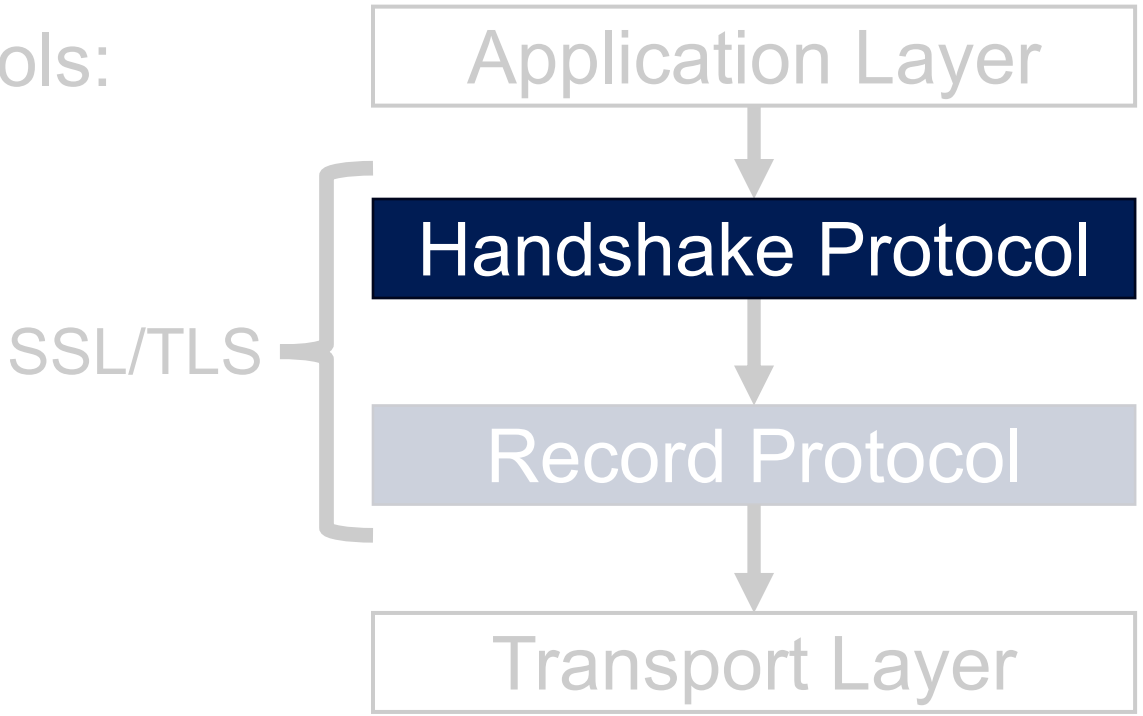
Handshake Protocol Summary



SSL/TLS Basics

- Runs in the presentation layer
- Uses symmetric crypto, asymmetric crypto, and digital signatures

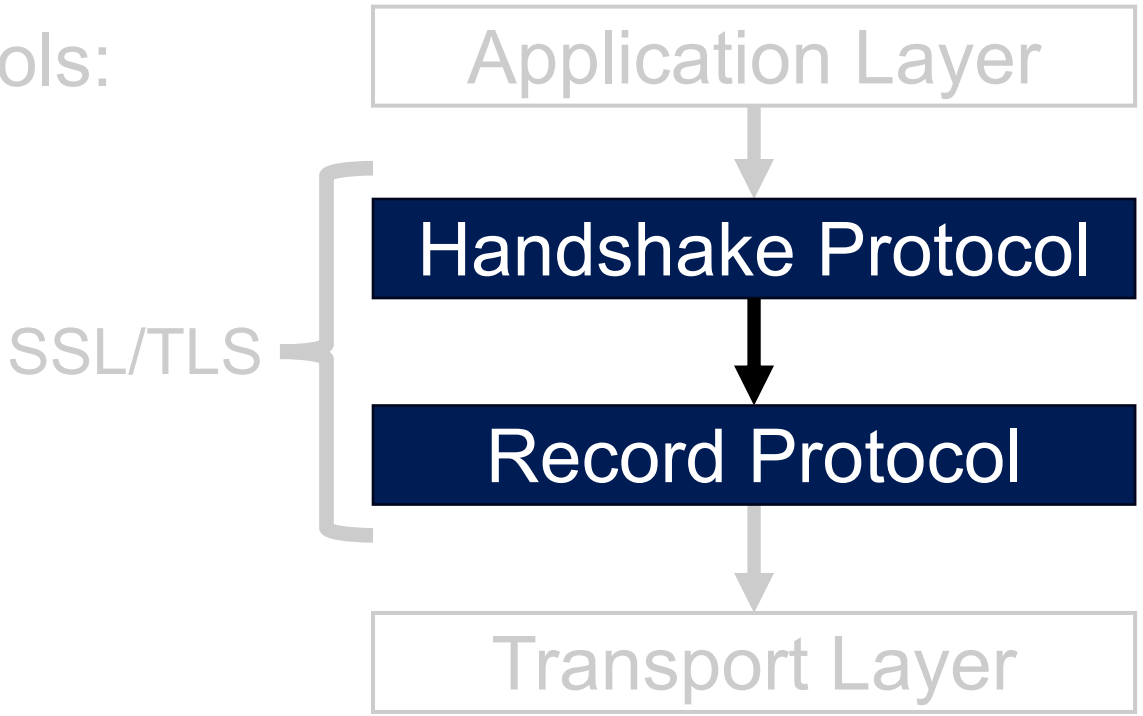
- Composed of two layers of protocols:
  1. Handshake protocol
  2. Record protocol



SSL/TLS Basics

- Runs in the presentation layer
- Uses symmetric crypto, asymmetric crypto, and digital signatures

- Composed of two layers of protocols:
  1. Handshake protocol
  2. Record protocol



SSL/TLS Record Protocol

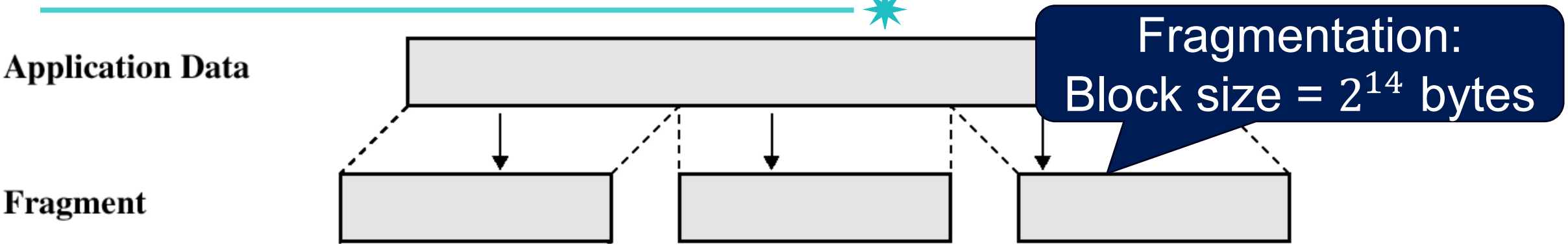
- Uses the symmetric keys established in the handshake protocol to protect **confidentiality**, **integrity**, and **authenticity** of data exchange

- **Confidentiality**
  - Using symmetric encryption

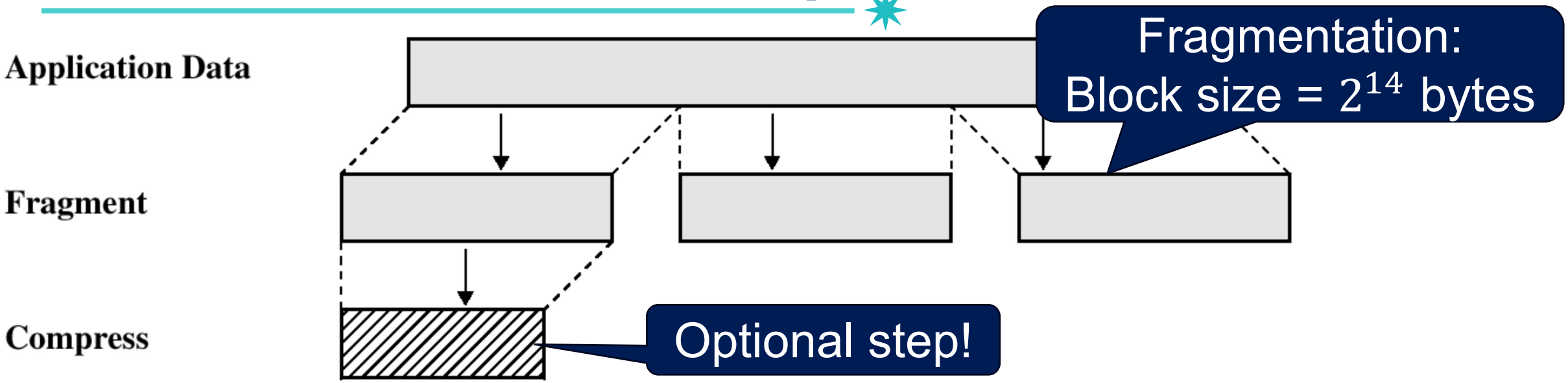
- **Integrity (+ Authenticity)**
  - Using a MAC with shared secret key

SSL Record Protocol Operation

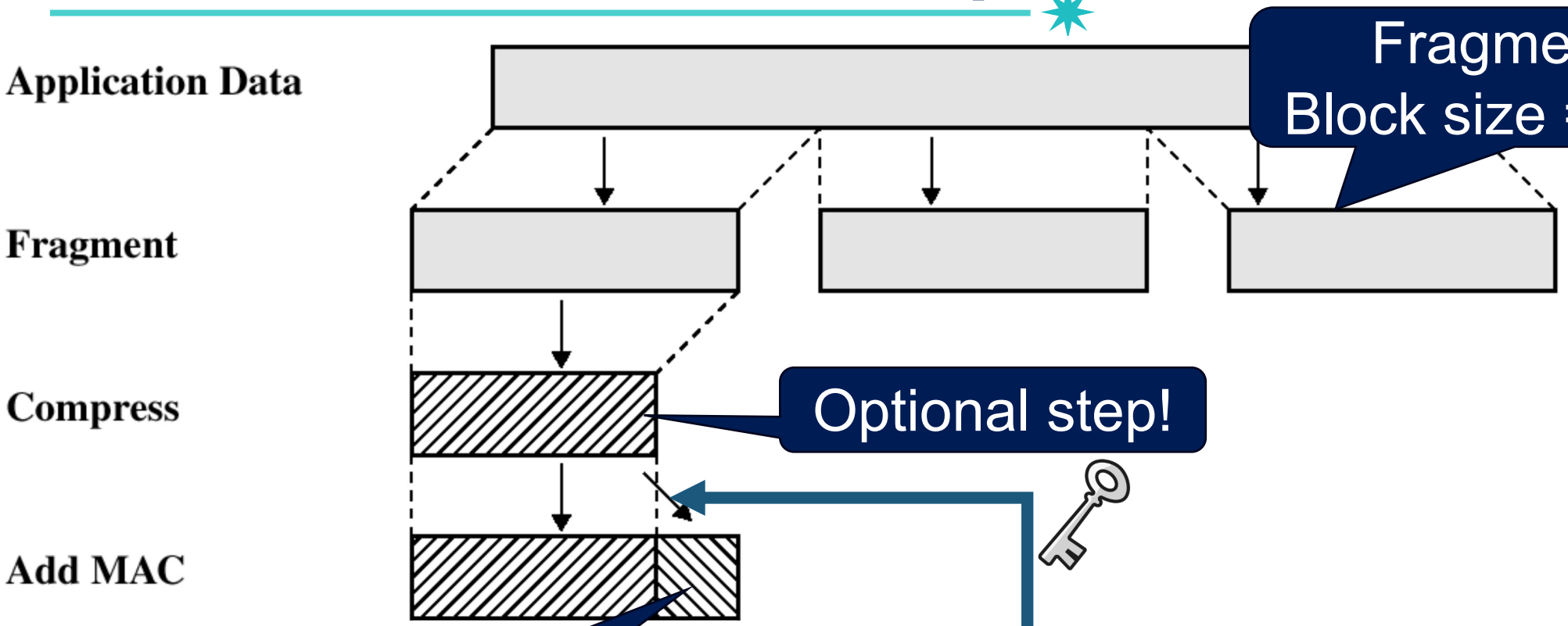
SSL Record Protocol Operation



SSL Record Protocol Operation



SSL Record Protocol Operation



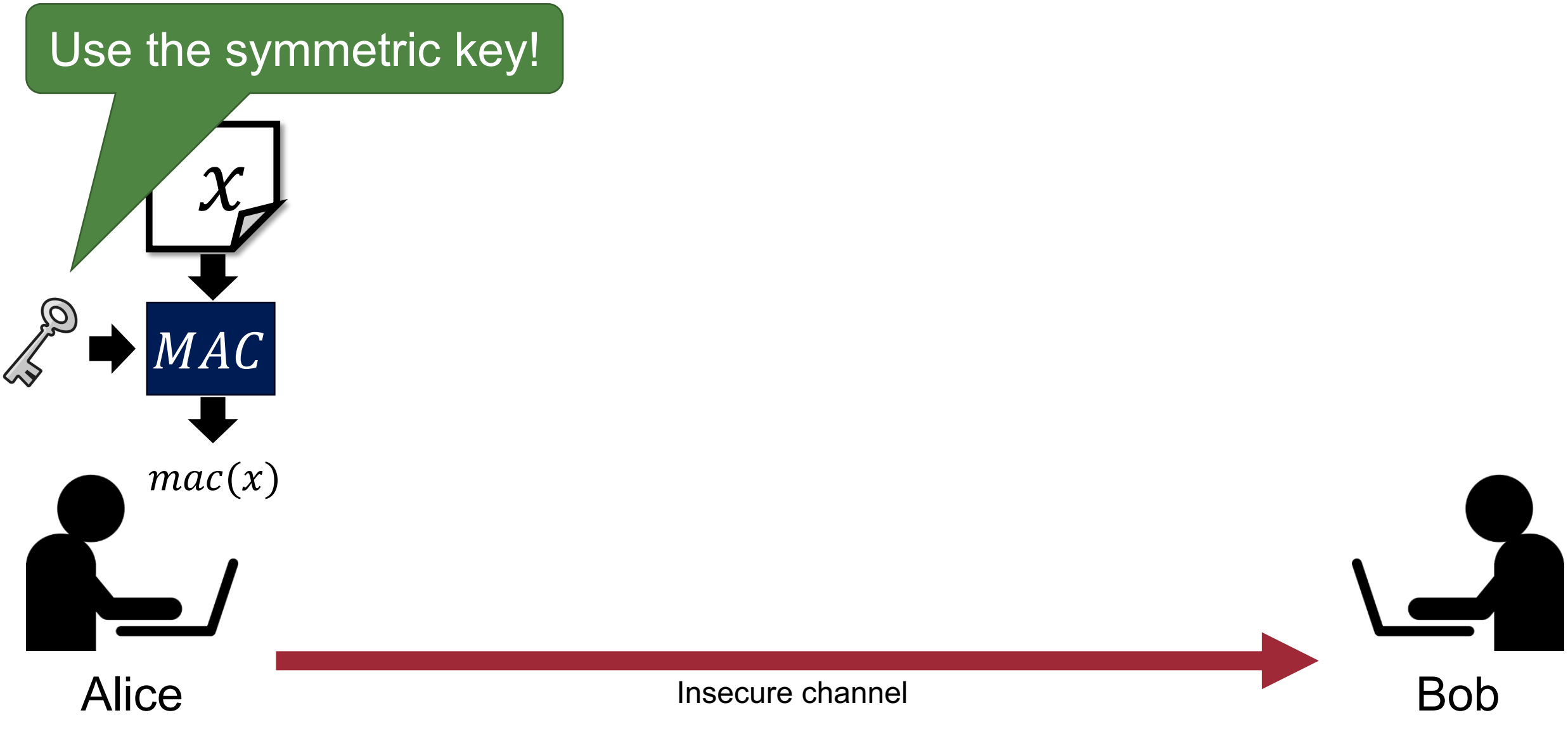
Fragmentation:
Block size = 2^{14} bytes

Optional step!

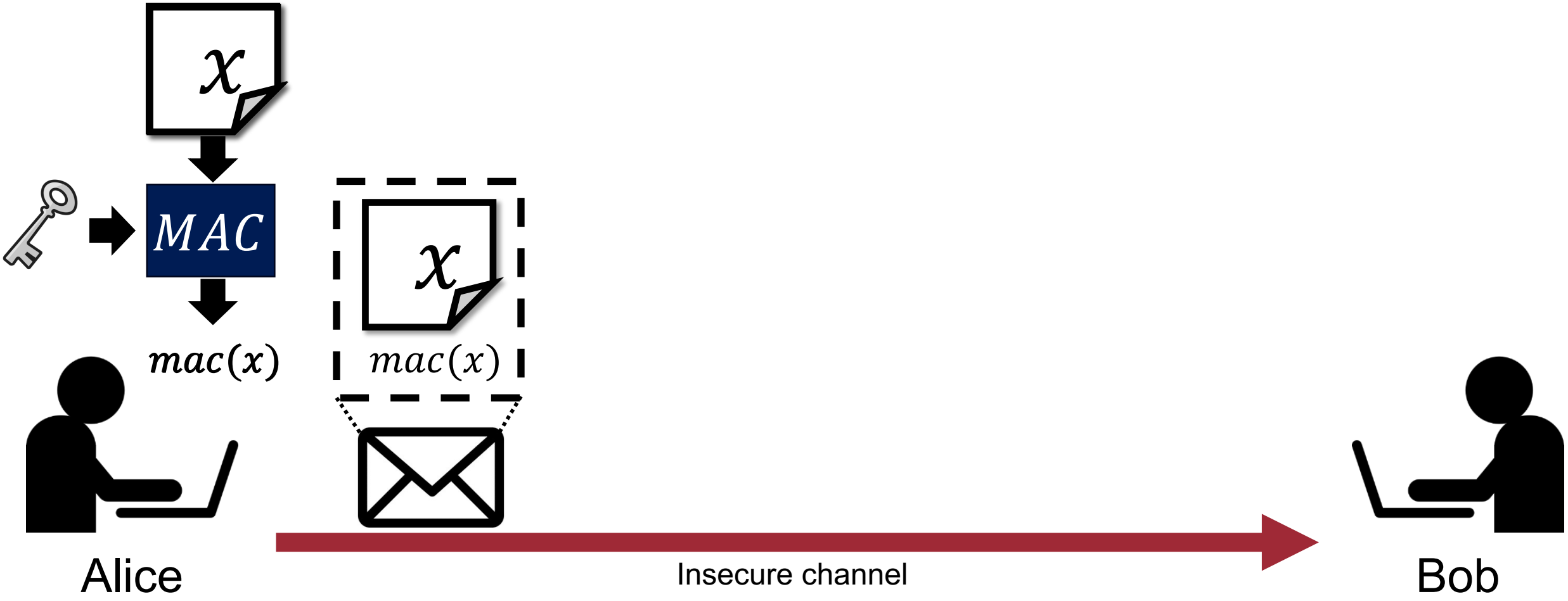
MAC: Check both integrity and authenticity

Client auth. key	Server auth. key	Client enc. key	Server enc. key	Client IV	Server IV

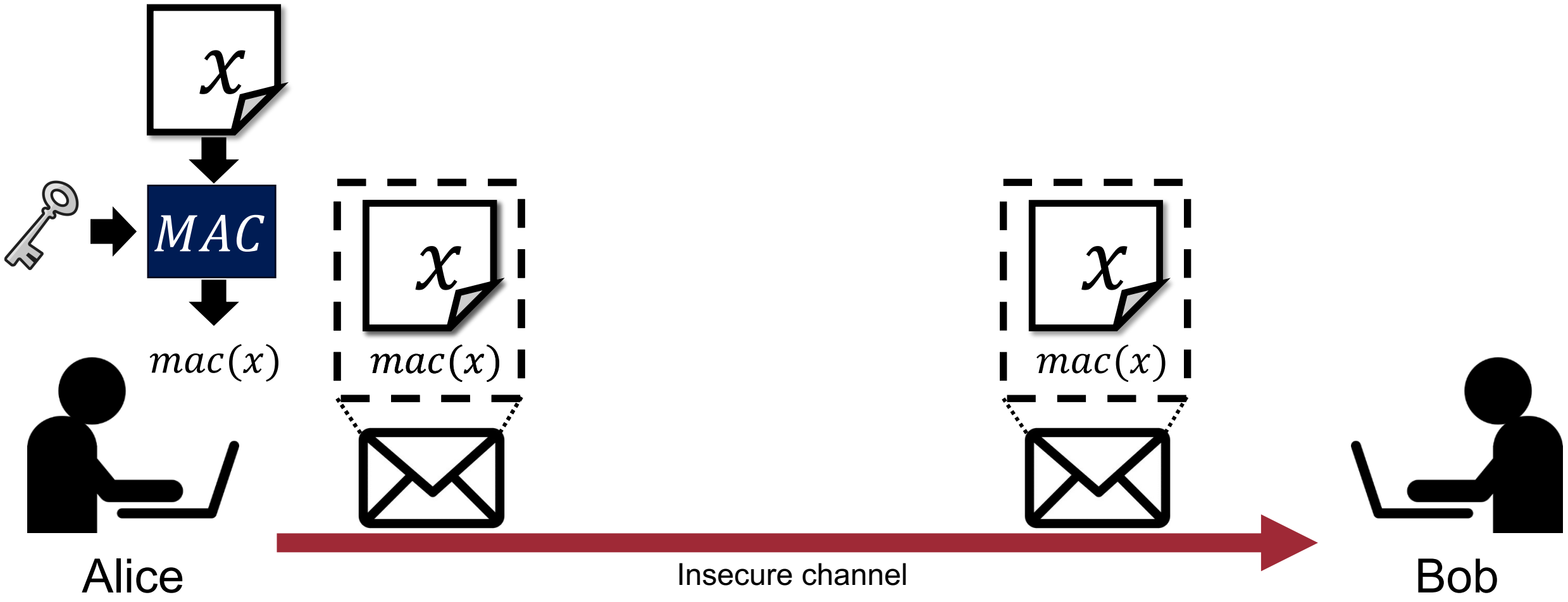
Recap: Message Authentication Codes (MAC)



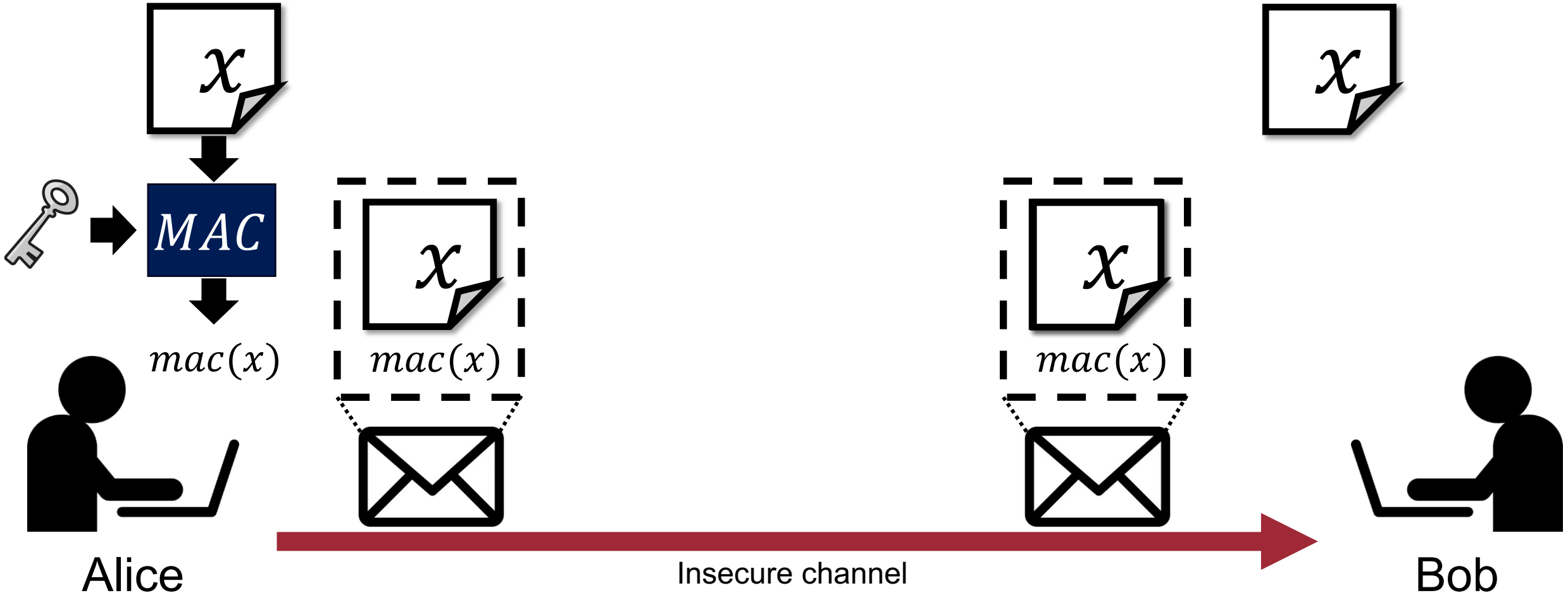
Recap: Message Authentication Codes (MAC)



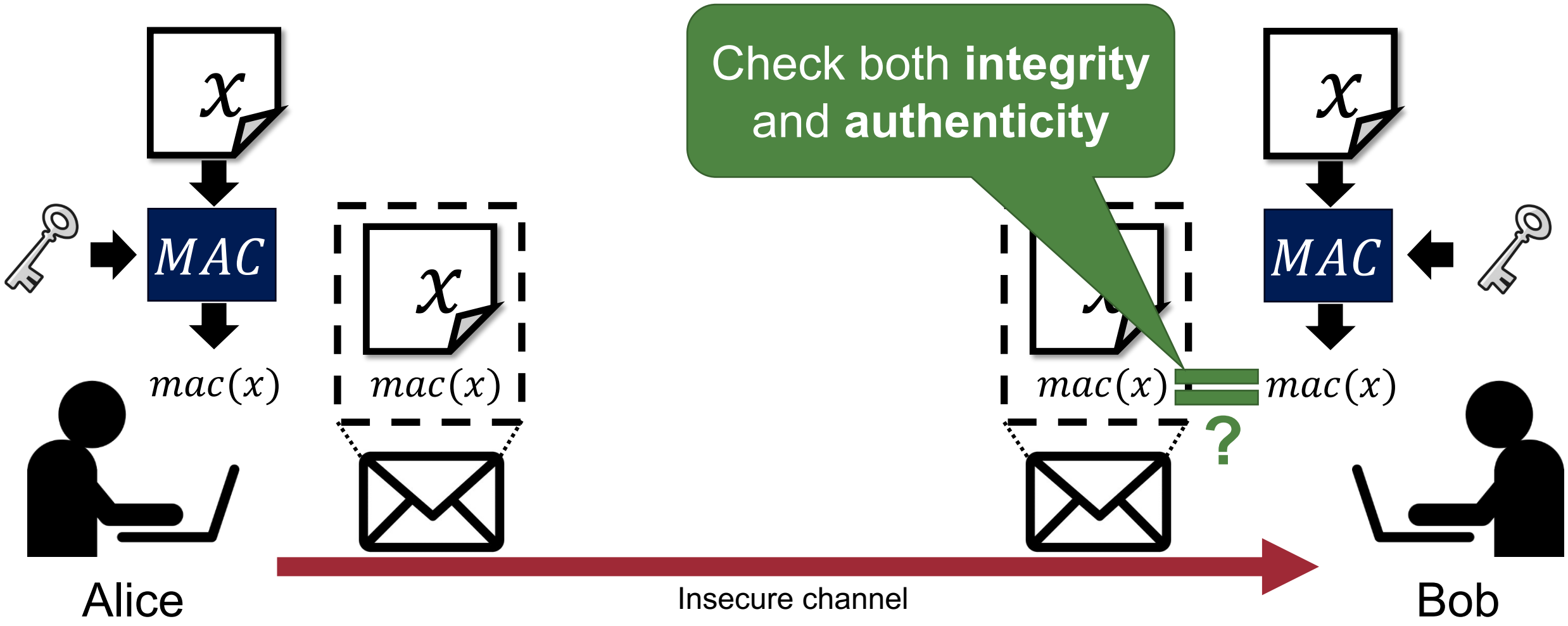
Recap: Message Authentication Codes (MAC)



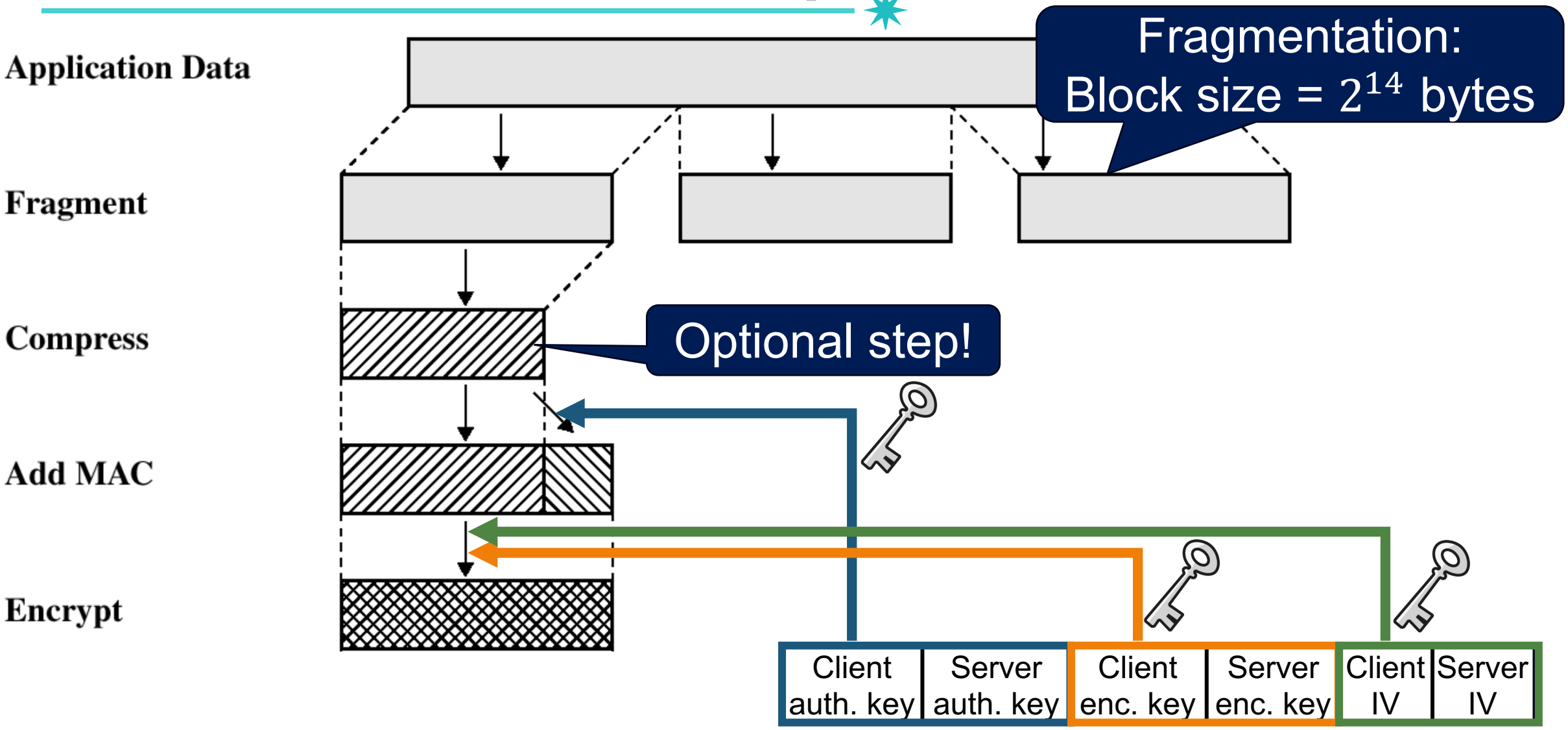
Recap: Message Authentication Codes (MAC)



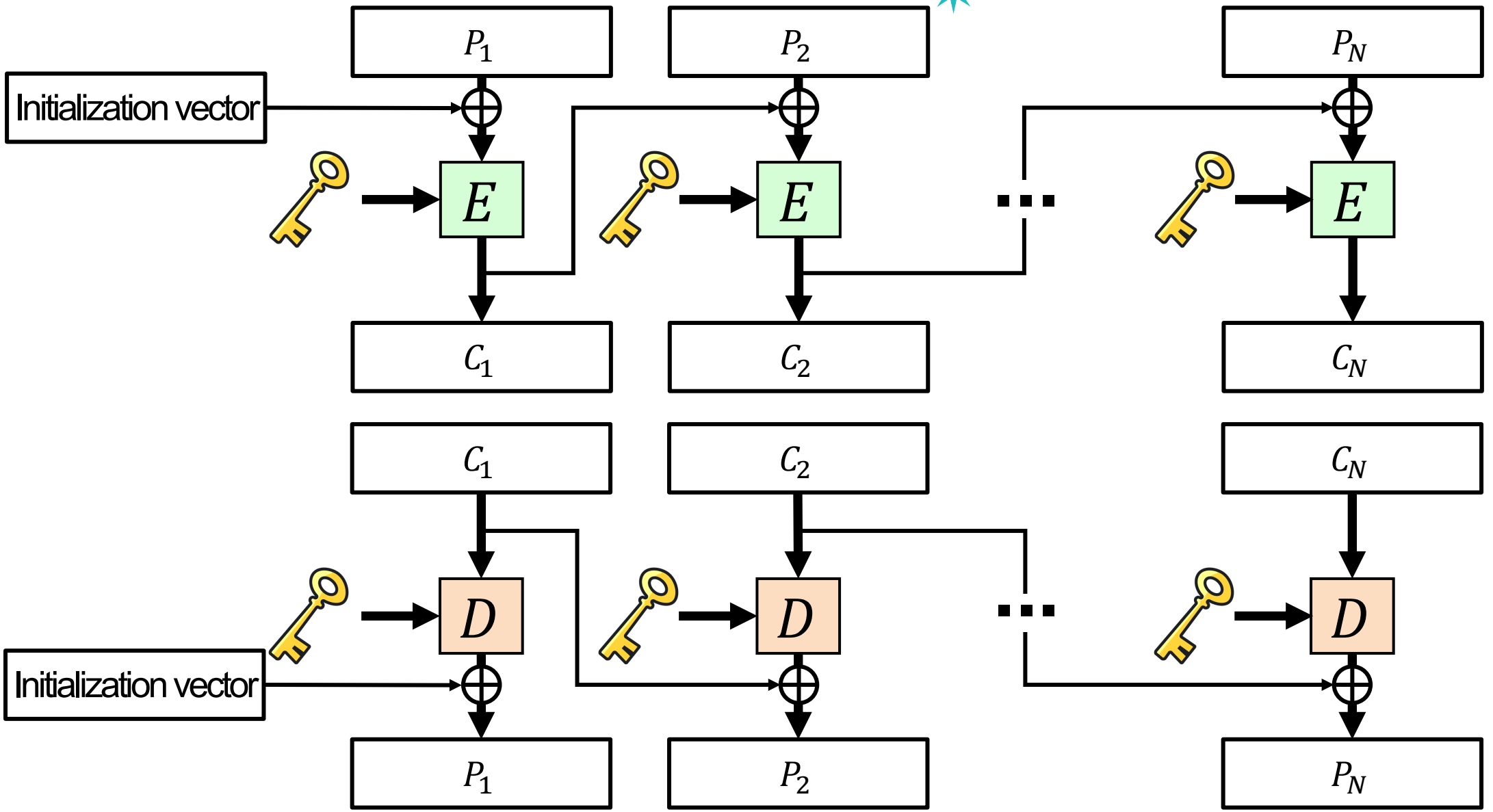
Recap: Message Authentication Codes (MAC)



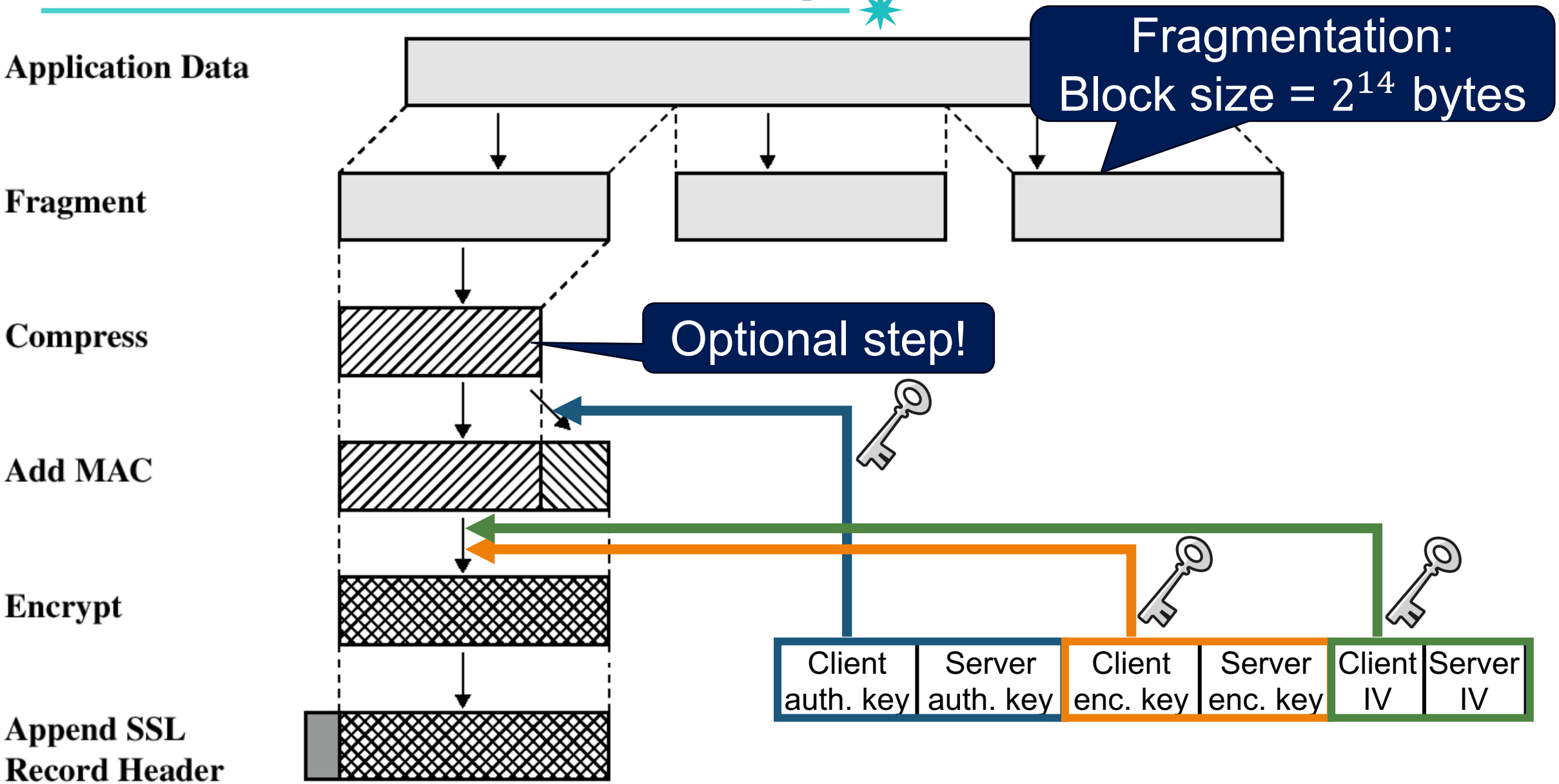
SSL Record Protocol Operation



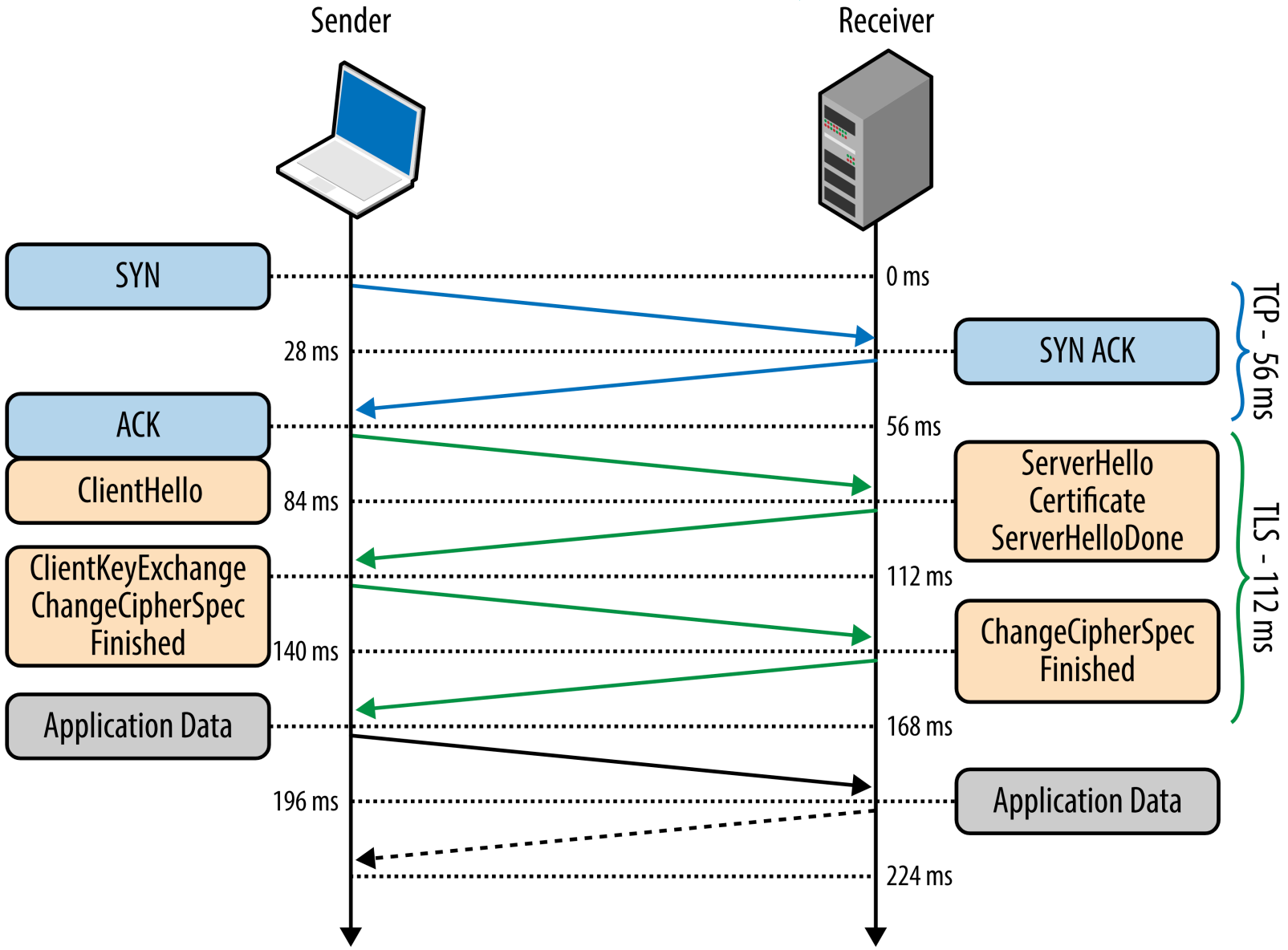
Recap: Cipher Block Chaining (CBC)



SSL Record Protocol Operation



SSL/TLS Final Overview



How SSL/TLS Provides Security Properties?

- Security goals: achieving…
  - **Confidentiality**
    - Asymmetric-key algorithm for key exchange (pre-master key)
    - Symmetric-key algorithm for data exchange
  - **Integrity**:
    - MAC (with hash algorithm)
    - If an attacker modifies the message, the recipient can detect the modification
  - **Authentication**
    - Authenticate the identity of the server using the server's certificate (and MAC)

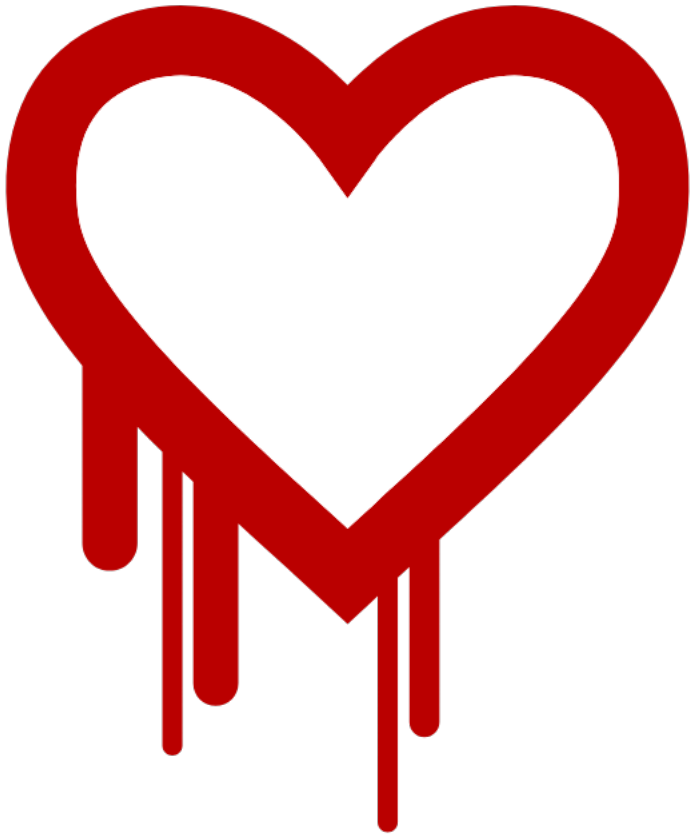
How SSL/TLS Provides Security Properties?

- Security goals: achieving…
  - **Confidentiality**
    - Asymmetric-key algorithm for key exchange (pre-master key)
    - Symmetric-key algorithm for data exchange
    - MAC (with hash algorithm)
    - If an attacker modifies the message, the recipient can detect the modification
  - **Authentication**
    - Authenticate the identity of the server using the server's certificate (and MAC)

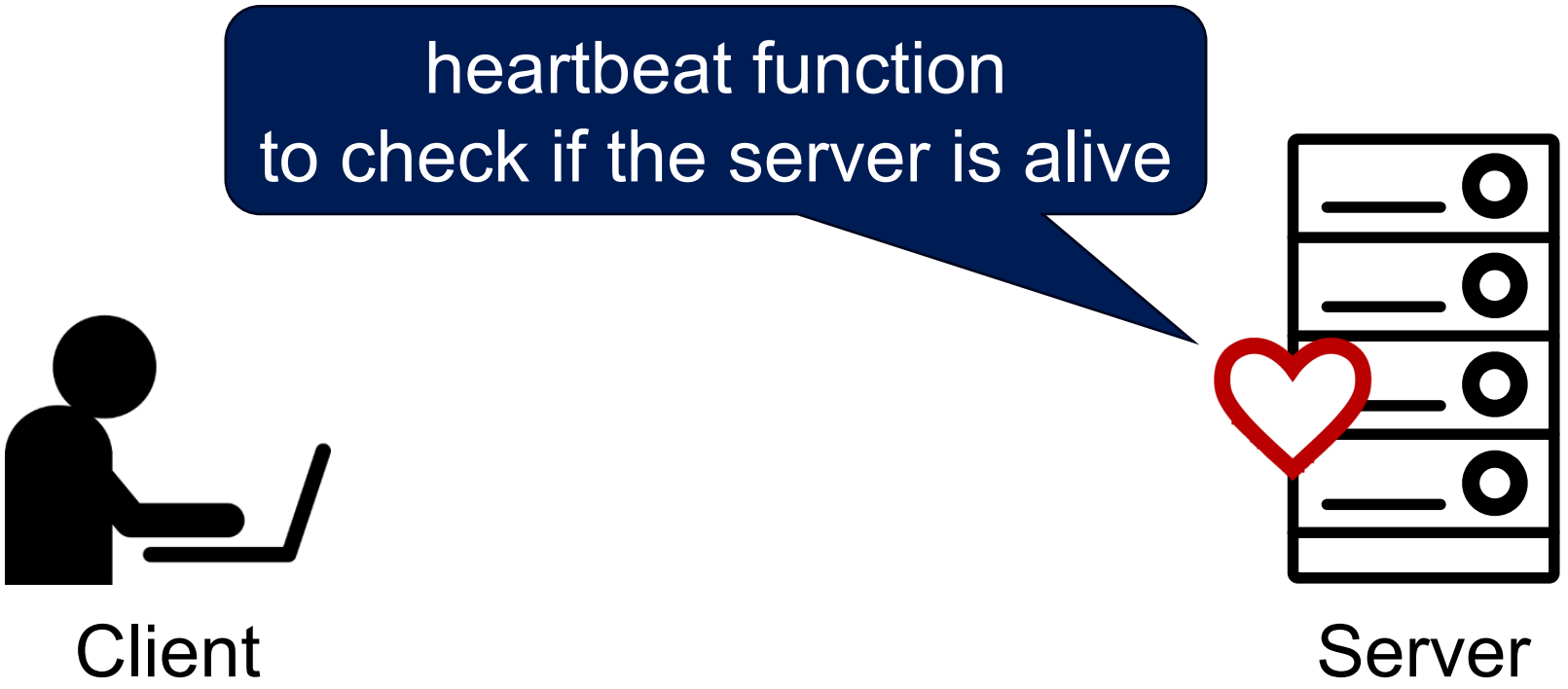
Are we safe now?

Recap: Heartbleed Bug (in 2014)

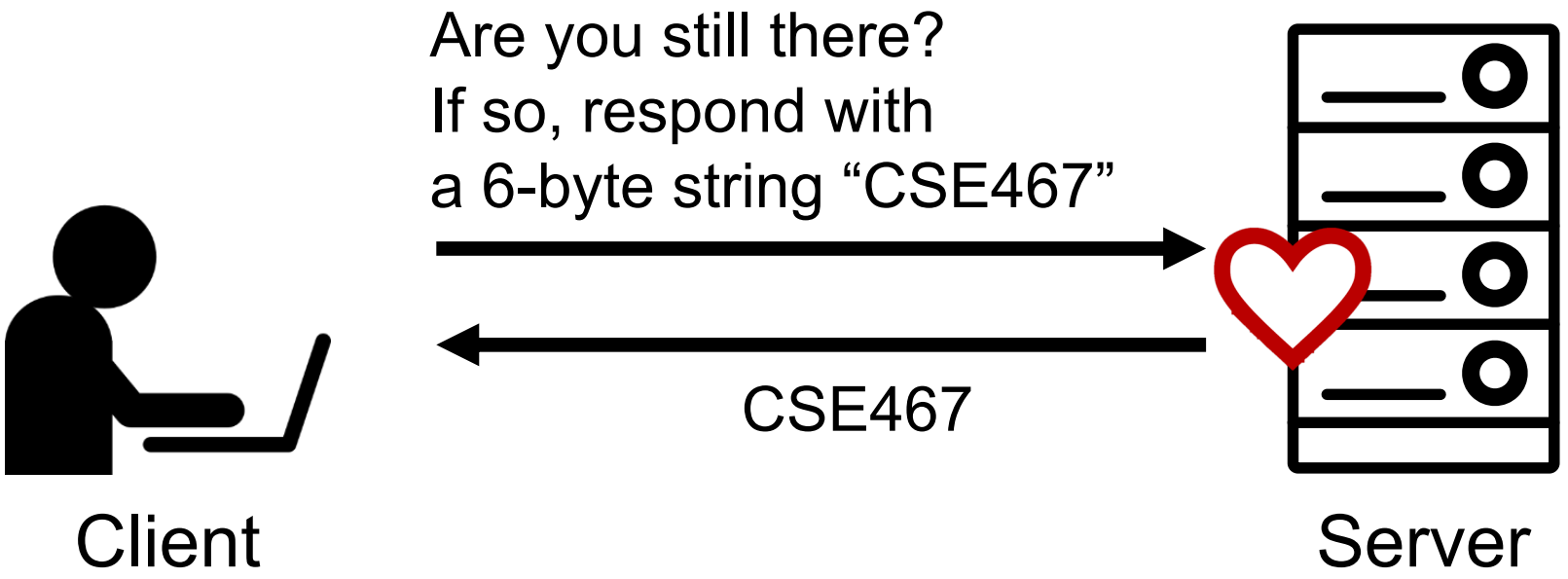
- Famous bug in OpenSSL (in TLS ***heartbeat***)
- An attacker can steal private keys



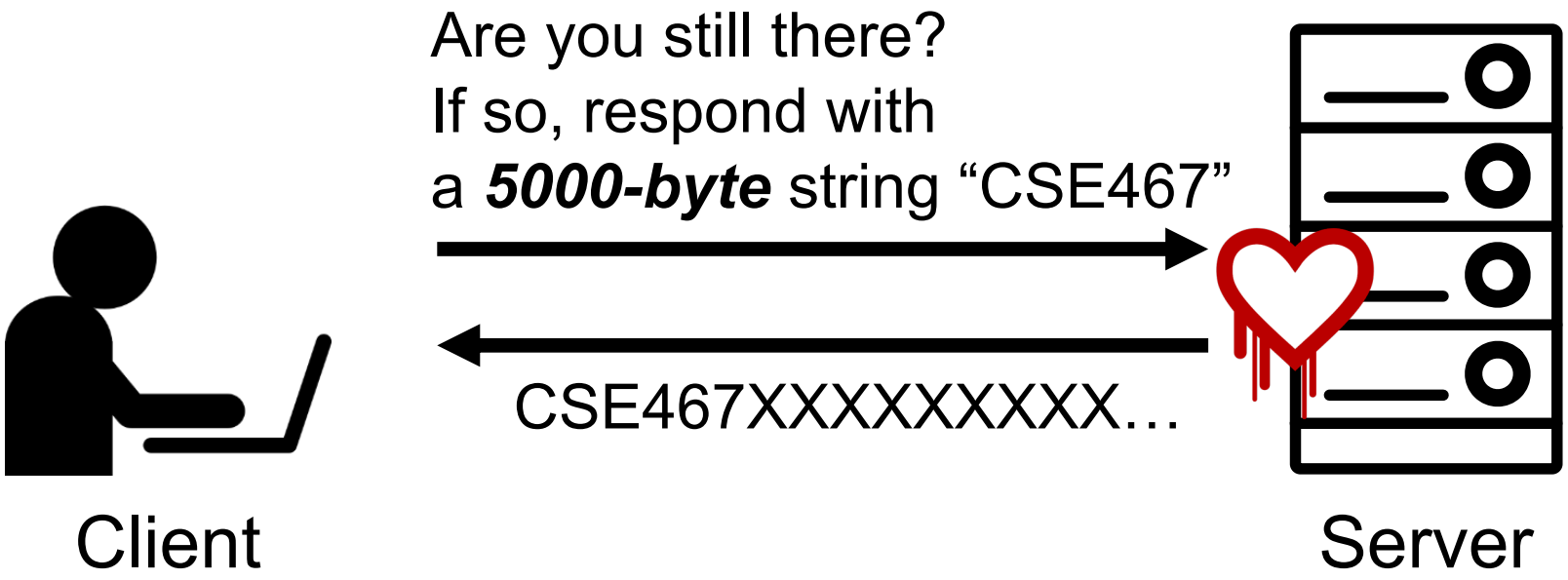
Heartbleed Bug: High-level Workflow



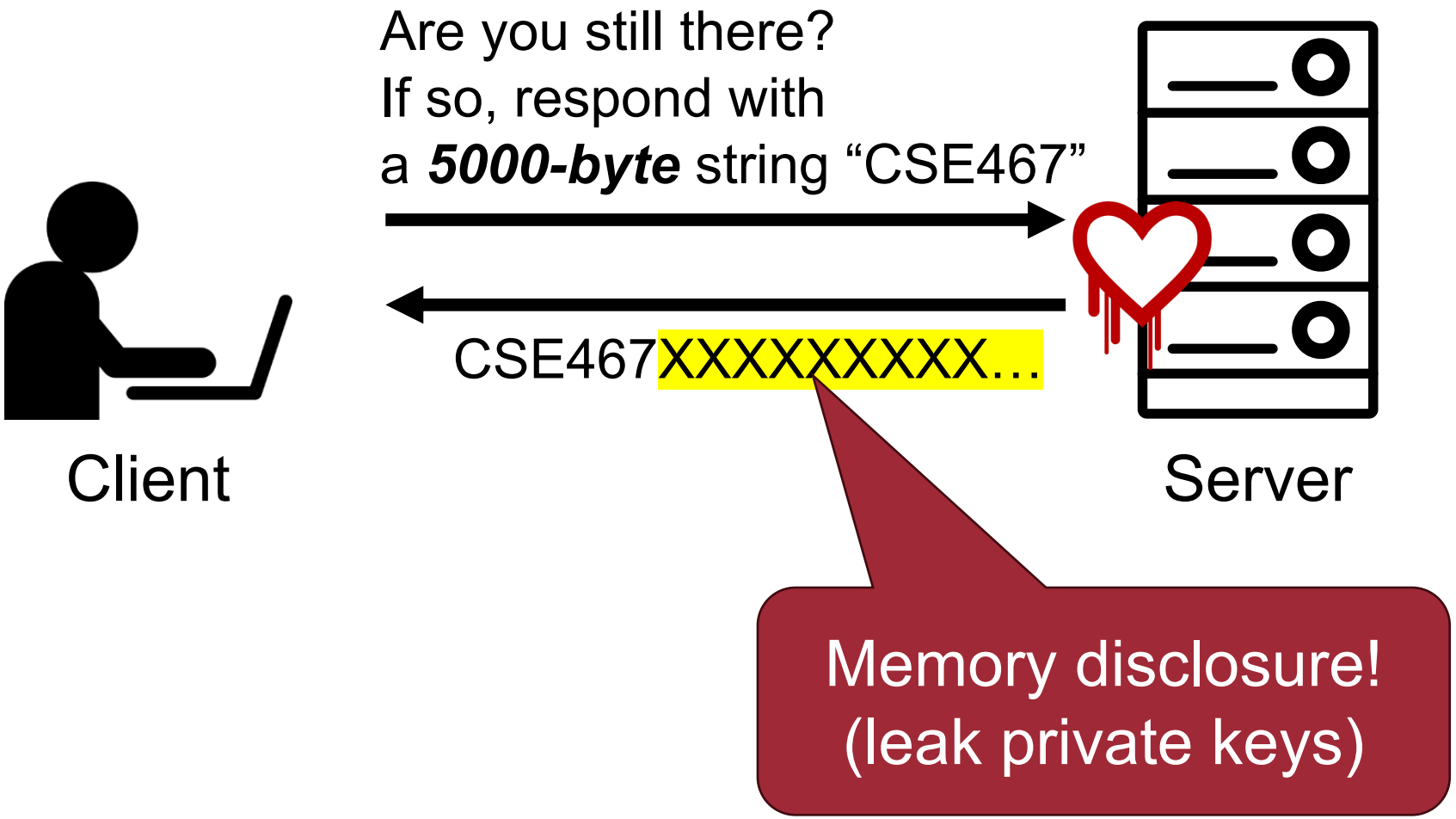
Heartbleed Bug: High-level Workflow



Heartbleed Bug: High-level Workflow



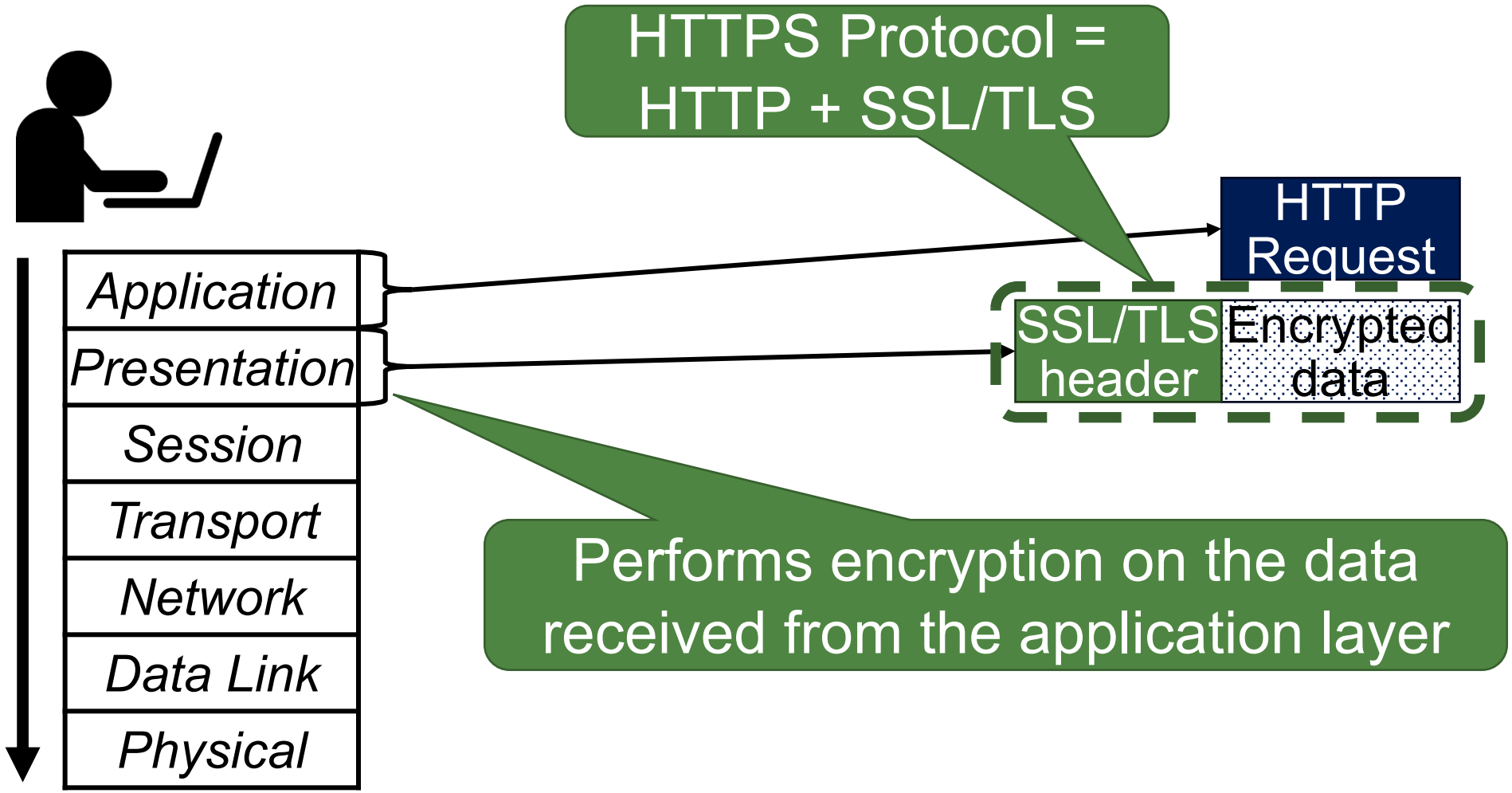
Heartbleed Bug: High-level Workflow



HTTPS 

HTTPS

- Adding a protocol layer for secure communication!



HTTPS – The Lock Icon

- Goal: the client (Human) can identify secure connection
  - SSL/TLS is being used to protect against active network attacker

- Lock icon should only be show when the page is secure against network attacker
  - All elements on the page fetched using HTTPS
  - Contents of the page have not been viewed or modified by an attacker
  - HTTPS certificate is valid – "This webpage is really comes from google.com server!"

HTTPS – The Lock Icon

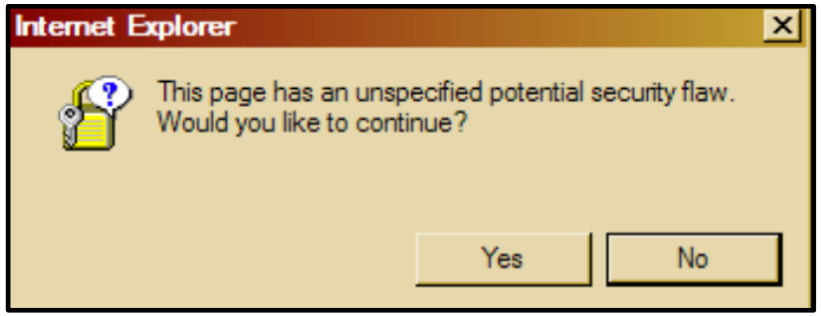
- Goal: the client (Human) can identify s
  - SSL/TLS is being used to protect against

What happens if page served over HTTPS but contains HTTP?

- Lock icon should only be show when the ge is secure against network attacker
  - All elements on the page fetched using HTTPS
  - Contents of the page have not been viewed or modified by an attacker
  - HTTPS certificate is valid – "This webpage is really comes from google.com server!"

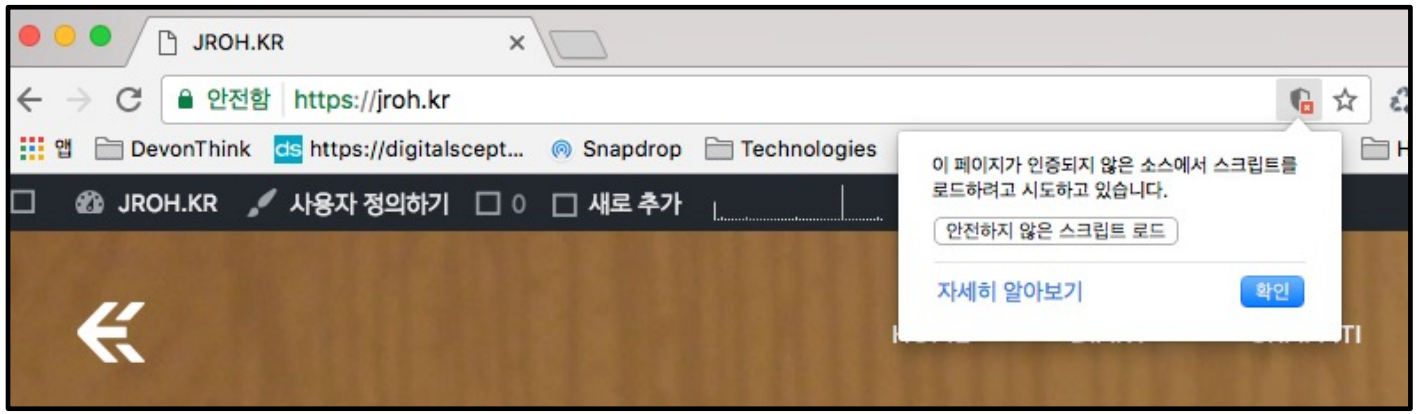
Mixed Content: Combining HTTPS and HTTP

- Page served over HTTPS but contains HTTP
  - IE 7: no lock, warning

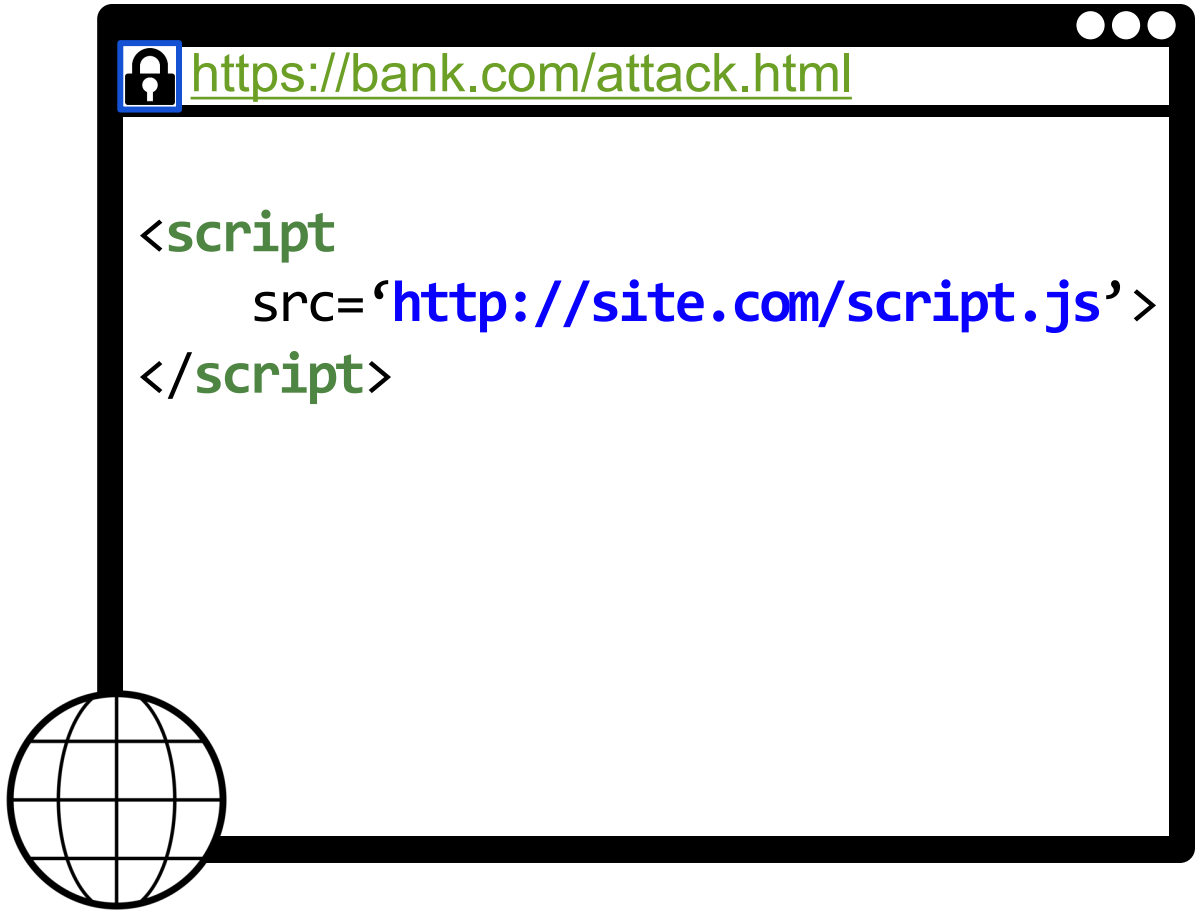


  - Firefox: “!” over lock, no warning by default

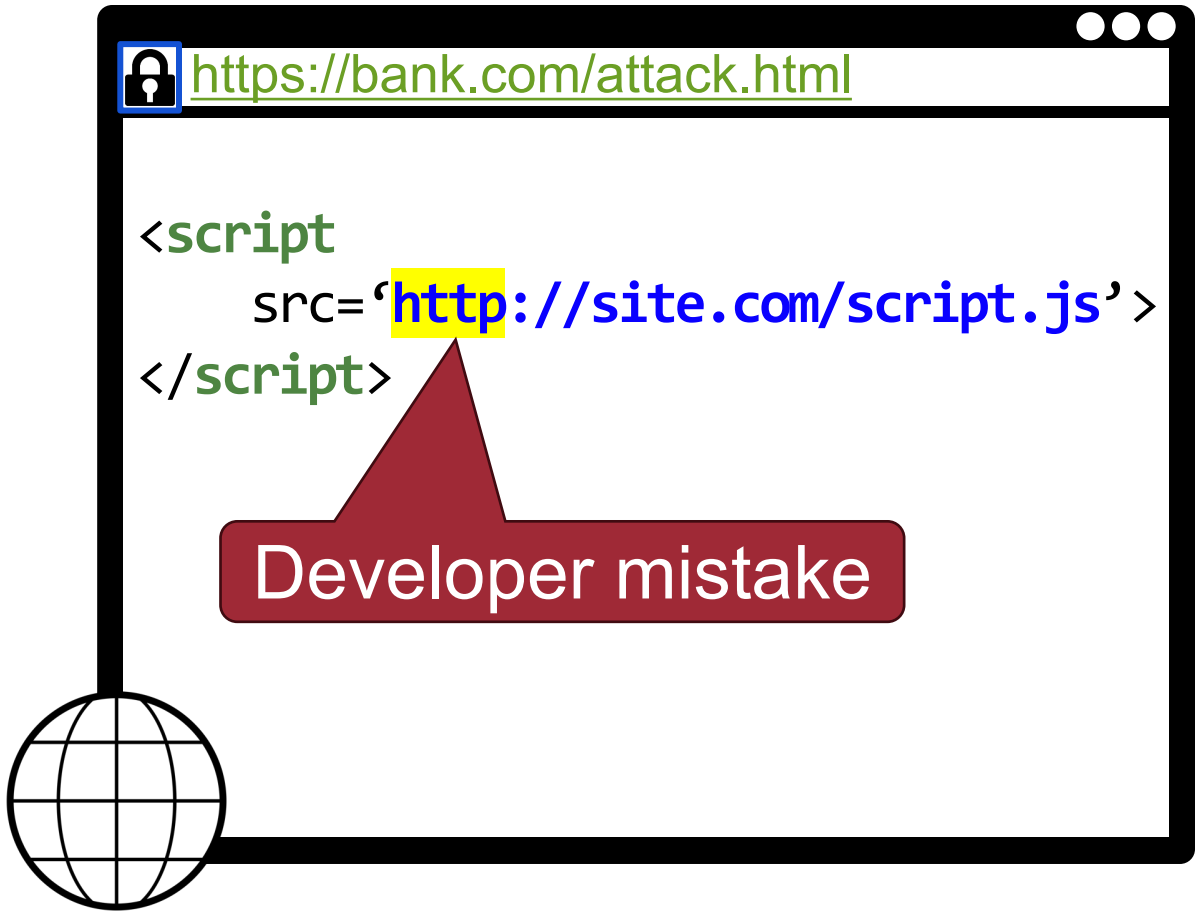
  - Safari: does not detect mixed content
  - Chrome: lock icon, warning



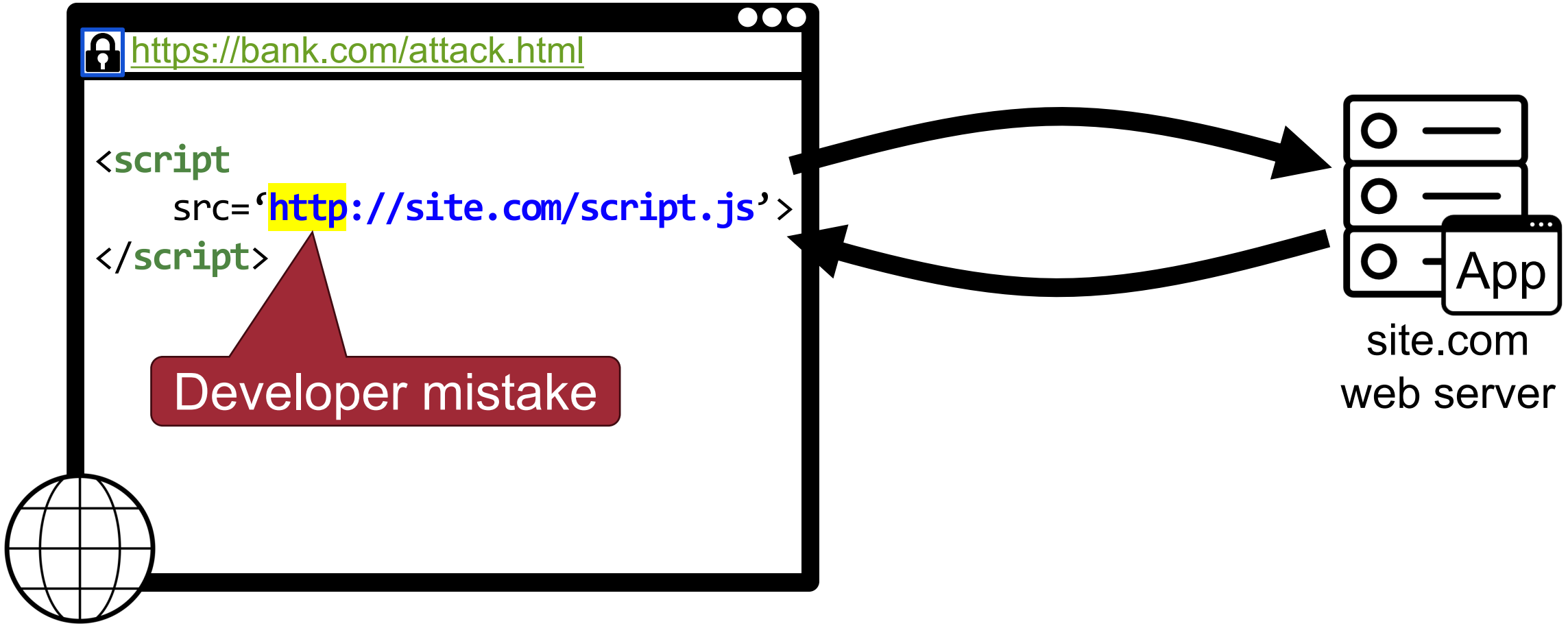
Mixed Content and Network Attacks



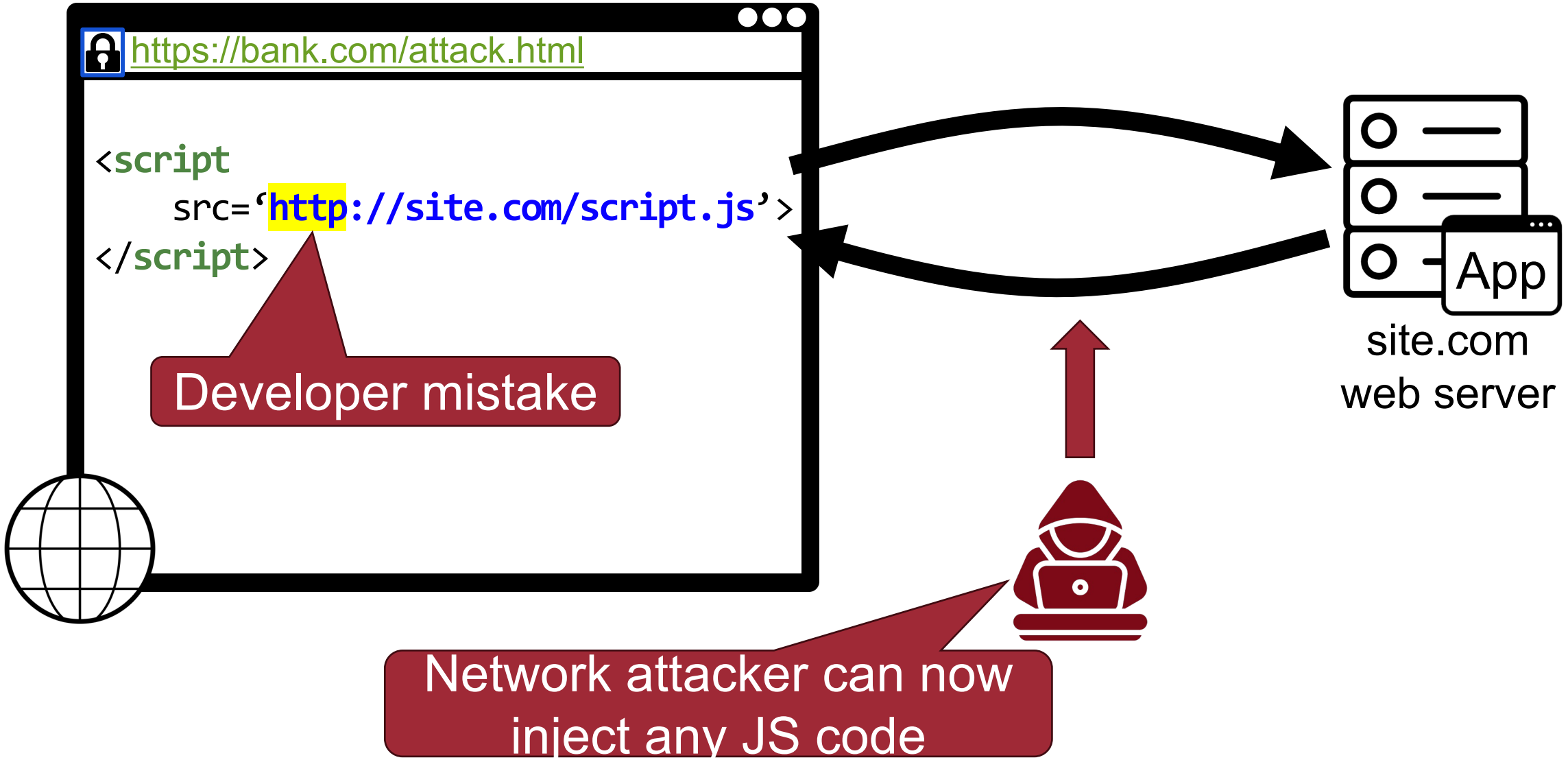
Mixed Content and Network Attacks



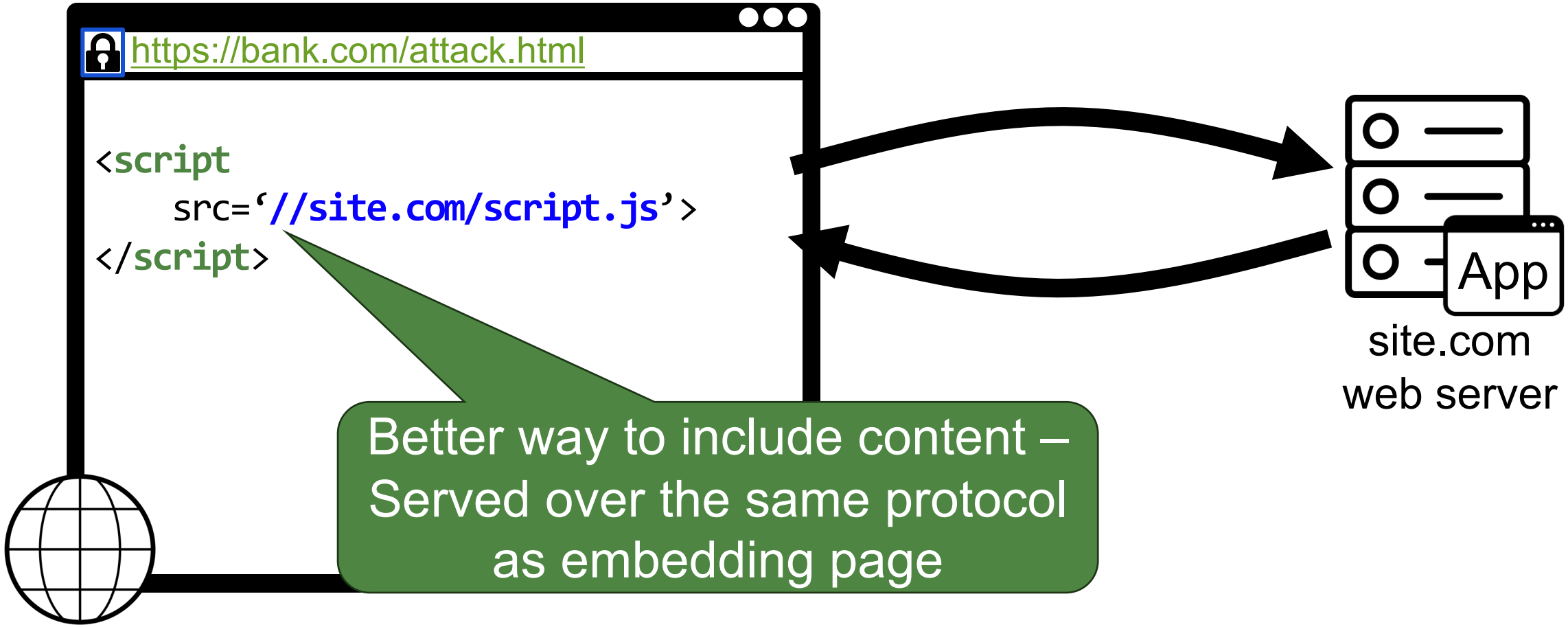
Mixed Content and Network Attacks



Mixed Content and Network Attacks

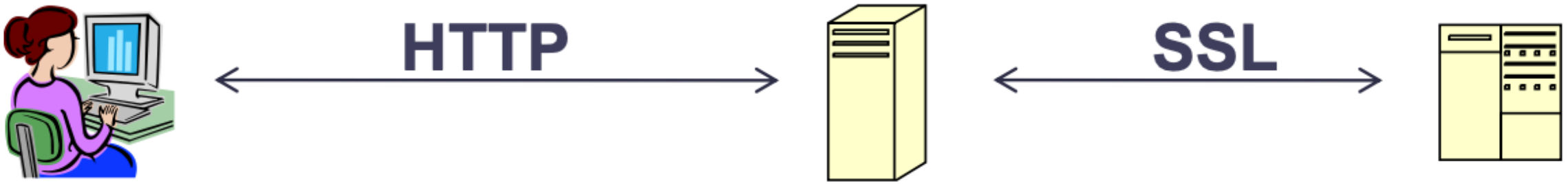


Mixed Content and Network Attacks



HTTPS – Upgrade

- Come to site over HTTP (Port no. 80), redirect to HTTPS (Port no. 443)!



Apache configuration

```
<VirtualHost *:80>
    ServerName [Domain]
    Redirect permanent / https://[Domain]/
</VirtualHost>
```

Summary

- SSL/TLS protocol
  - Satisfy confidentiality
  - Satisfy integrity
  - Satisfy authentication

- HTTPS: HTTP + SS/TLS protocol

Question?