CS 4173/5173 COMPUTER SECURITY

SSL/TLS

Based on Shmatikov's slides

AUTHENTICATION IN LARGE NETWORKS

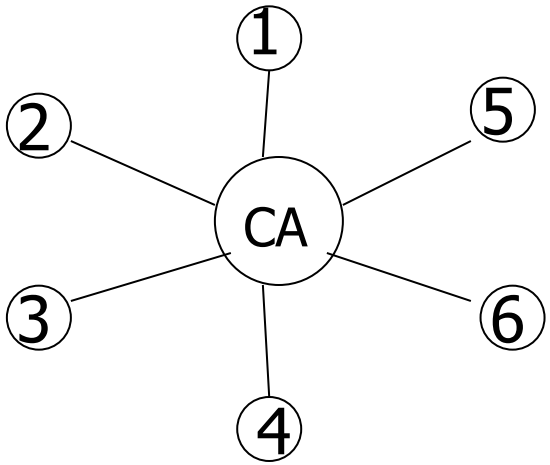
- Problem: authentication for large networks
- Solution #1
  - Key Distribution Center (KDC)
  - Based on secret key cryptography
  - Representative solution: Kerberos
- Solution #2
  - Public Key Infrastructure (PKI)
  - Based on public key cryptography
  - Representative solution: SSL/TLS

WHAT IS PKI

- Informally, the infrastructure supporting the use of public key cryptography.
- A PKI consists of
  - Certificate Authority (CA)
  - Certificates
  - A repository for retrieving certificates
  - A method of revoking/updating certificates

CERTIFICATION AUTHORITIES (CA)

- A CA is a trusted node that maintains the public keys for all nodes (Each node maintains its own private key)



If a new node is inserted in the network, only that new node and the CA need to be configured with the public key for that node

CERTIFICATES

- A CA is involved in authenticating users' public keys by generating certificates
- A certificate is a signed message vouching that a particular name goes with a particular public key
- Example:
  1. [Alice's public key is 876234]$_{carol}$
  2. [Ted's public key is 676554]$_{Alice}$ & [Alice's public key is 876234]$_{carol}$

- Knowing the CA's public key, users can verify the certificate and authenticate Alice's public key

CERTIFICATES

- Certificates can hold expiration date and time
- Alice keeps the same certificate as long as she has the same public key and the certificate does not expire
- Alice can append the certificate to her messages so that others know for sure her public key

EXAMPLE

- CA – everyone knows CA's public key.
  - CA is trusted.
- Alice wants to communicate to the real Bob
  - She sends a request to CA
  - Obtains a digital certificate from CA:

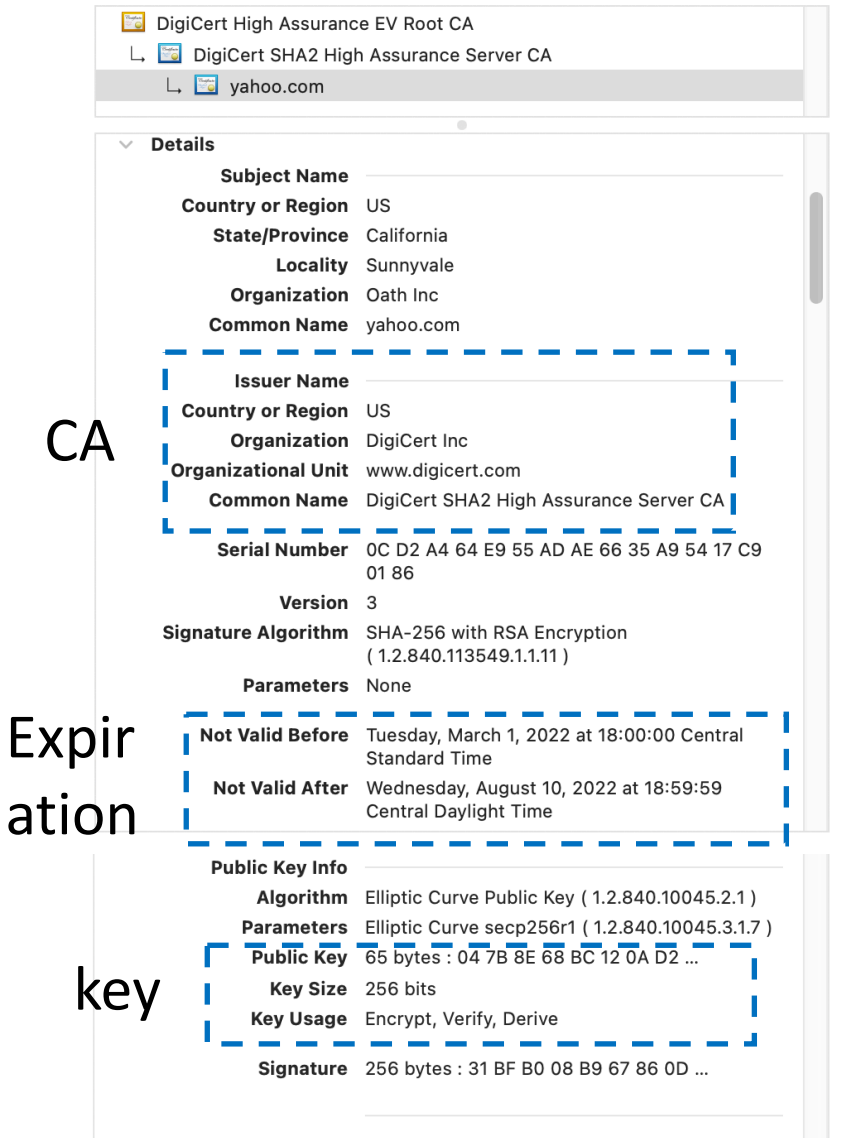
Bob's public key is
1902A12B2318871BF1
Expiration: 1/1/2023
[signed by CA]

Bob's D-H g, p, and T are
129381,102A7182019284FF, 910A81213
Expiration: 1/1/2023
[signed by CA]

Q: digital certificate vs digital signature?

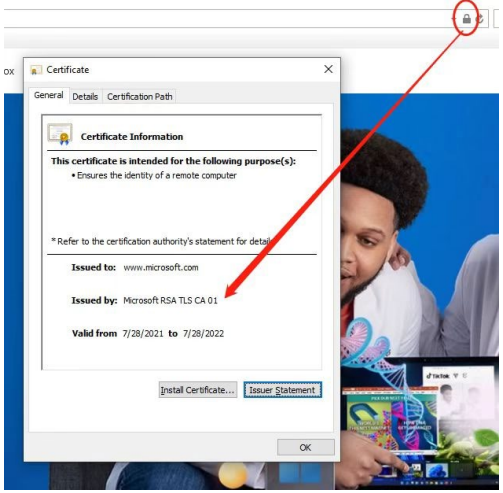
EXAMPLE

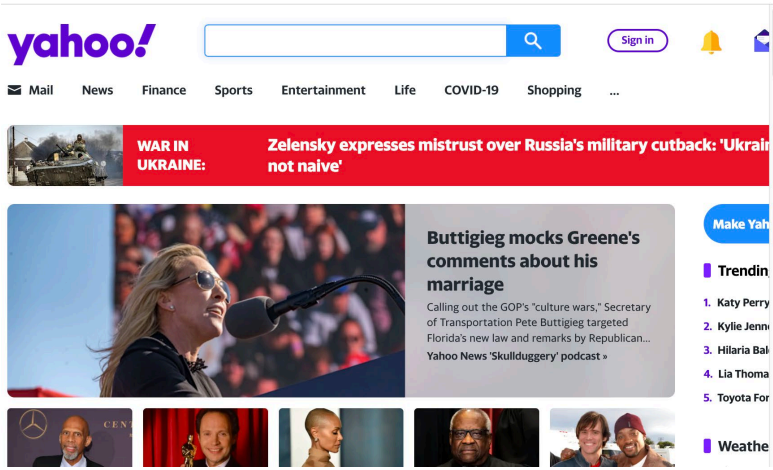
- Everyone knows CA's public key.
  - CA is trusted
- You wants to visit Yahoo
  - You send a request to CA,
  - then obtain a digital certificate from CA

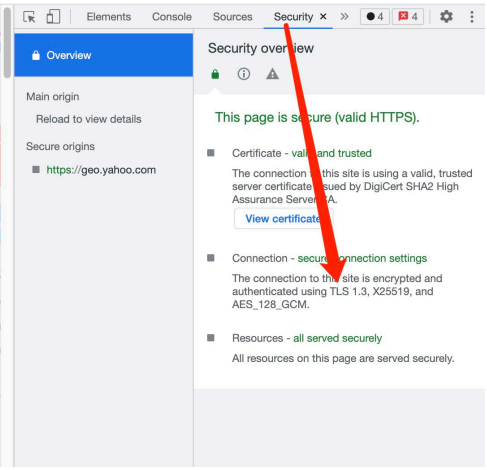


WHAT IS SSL / TLS?

- Secure Sockets Layer and Transport Layer Security protocols
  - Same protocol design, different crypto algorithms. TLS is a recent upgraded version of SSL.
- De facto standard for Internet security
  - "The primary goal of the TLS protocol is to provide privacy and data integrity between two communicating applications"
- Deployed in every web browser; also VoIP, payment systems, distributed systems, cloud drives, etc.







SSL / TLS GUARANTEES

- End-to-end secure communications in the presence of a network attacker
  - Attacker completely owns the network: controls Wi-Fi, DNS, routers, his own websites, can listen to any packet, modify packets in transit, inject his own packets into the network
    - Including man-in-the-middle
- Scenario: you are reading your email from a dangerous café connected via a rooted Wi-Fi access point to a dodgy ISP in a hostile authoritarian country

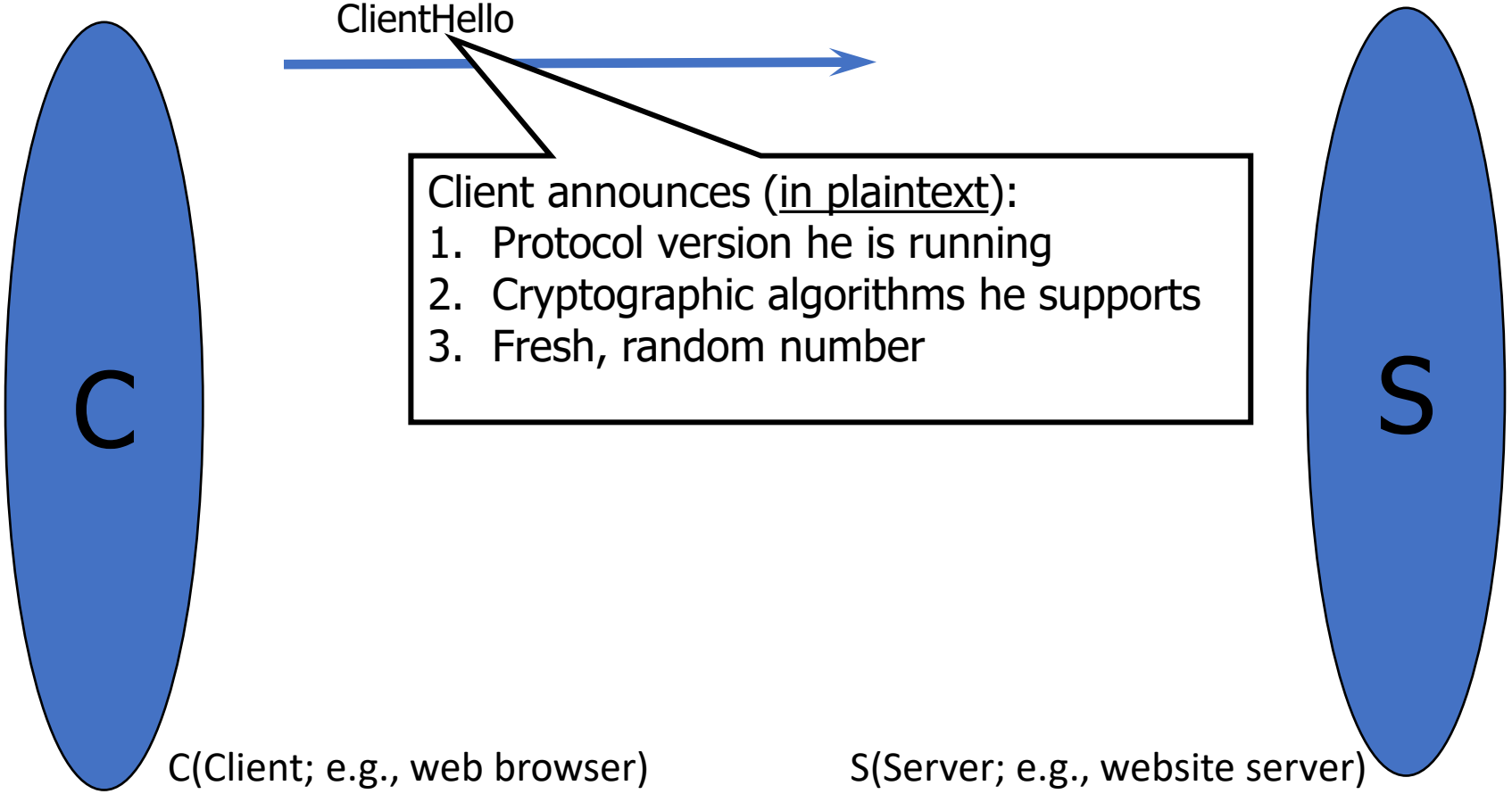
HISTORY OF THE PROTOCOL

- SSL 1.0 – internal Netscape design, early 1994?
  - Lost in the mists of time
- SSL 2.0 – Netscape, Nov 1994
  - Several weaknesses
- SSL 3.0 – Netscape and Paul Kocher, Nov 1996
- TLS 1.0 – Internet standard, Jan 1999
  - Based on SSL 3.0, but not interoperable (uses different cryptographic algorithms)
- TLS 1.1 – Apr 2006
- TLS 1.2 – Aug 2008
- TLS 1.3 – Aug 2018

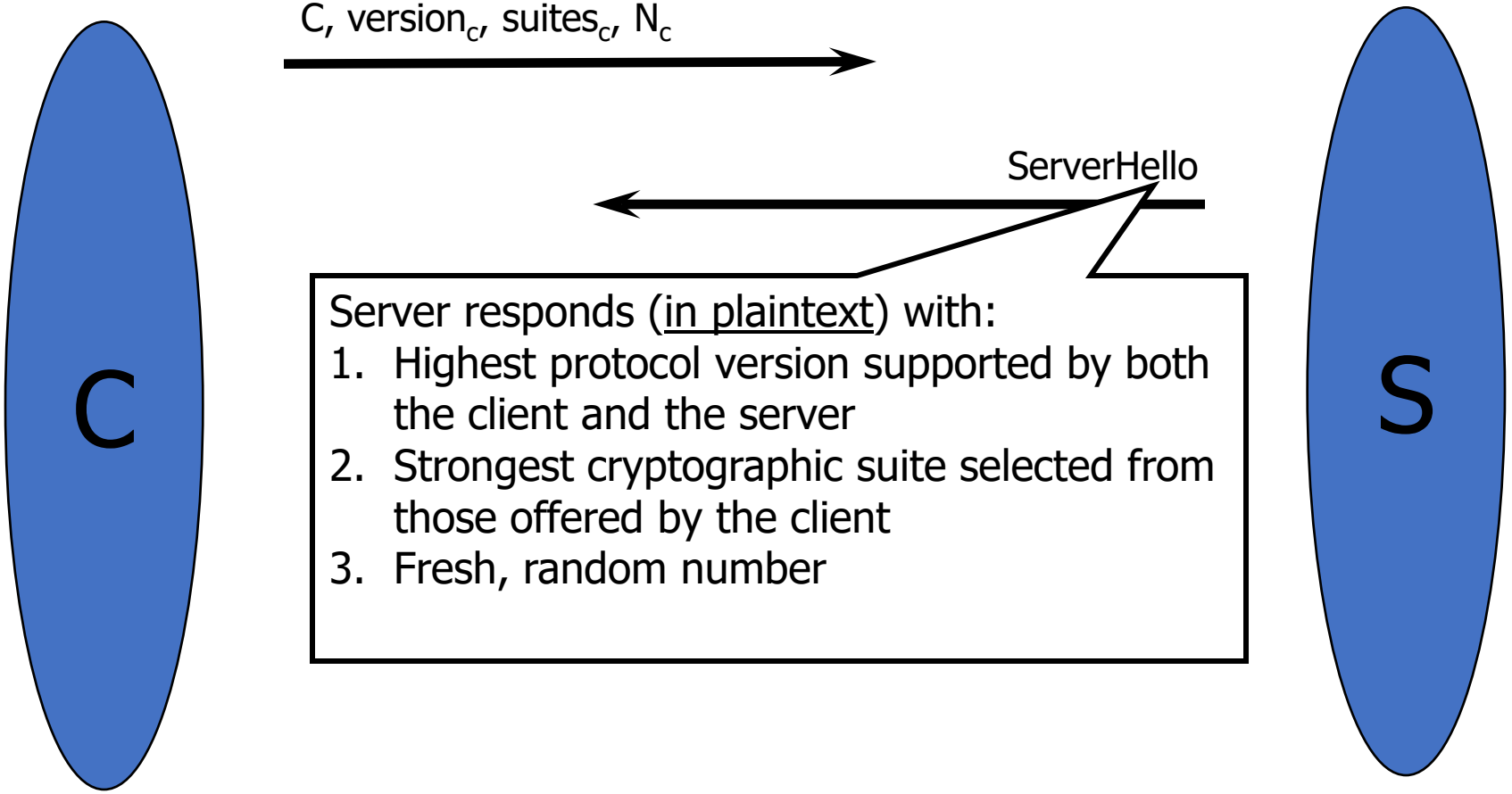
SSL BASICS

- SSL consists of two protocols
- Handshake protocol
  - Uses public-key cryptography to establish several shared secret keys between the client and the server
- Record protocol
  - Uses the secret keys established in the handshake protocol to protect confidentiality, integrity, and authenticity of data exchange between the client and the server

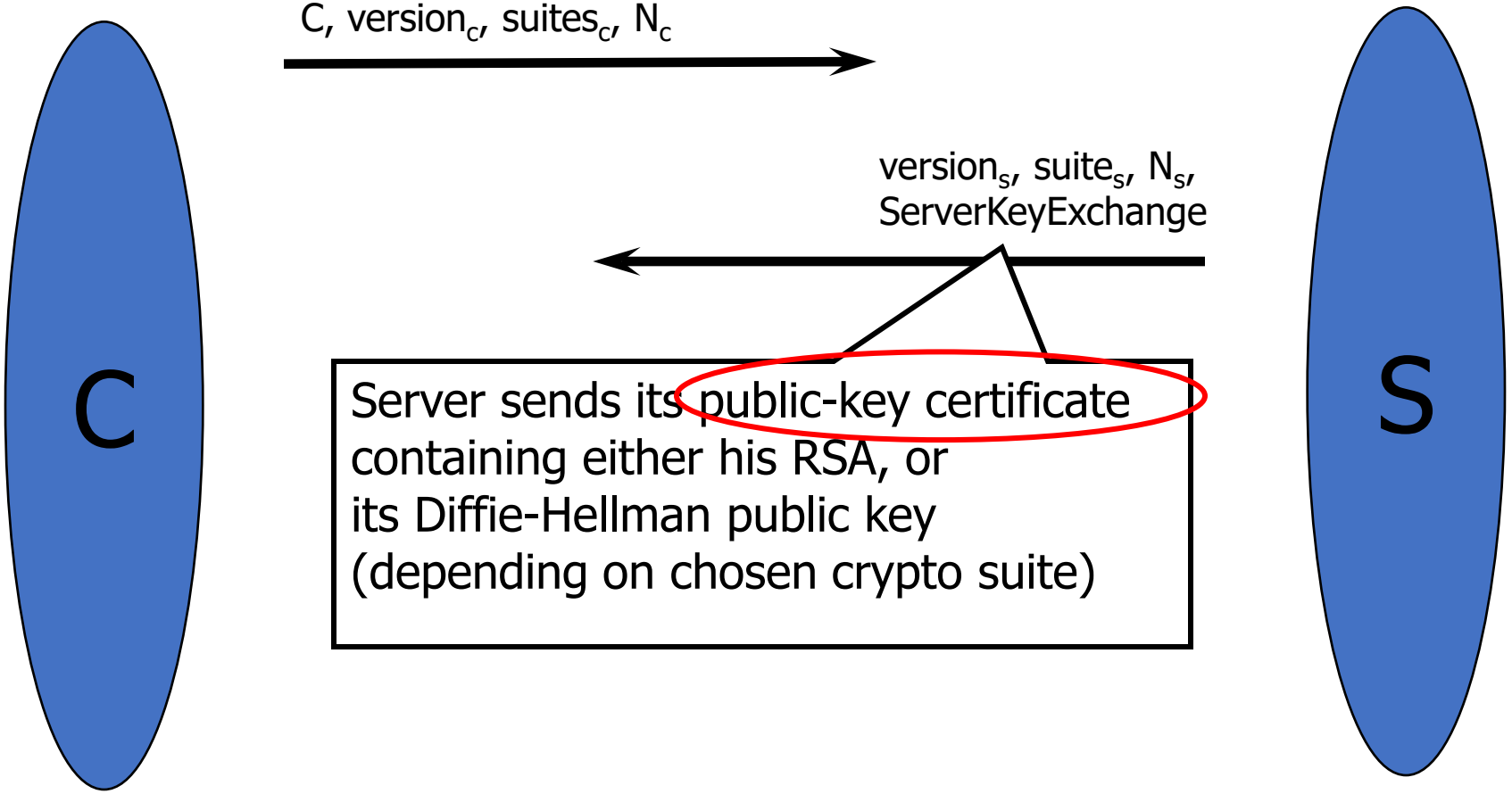
CLIENT HELLO



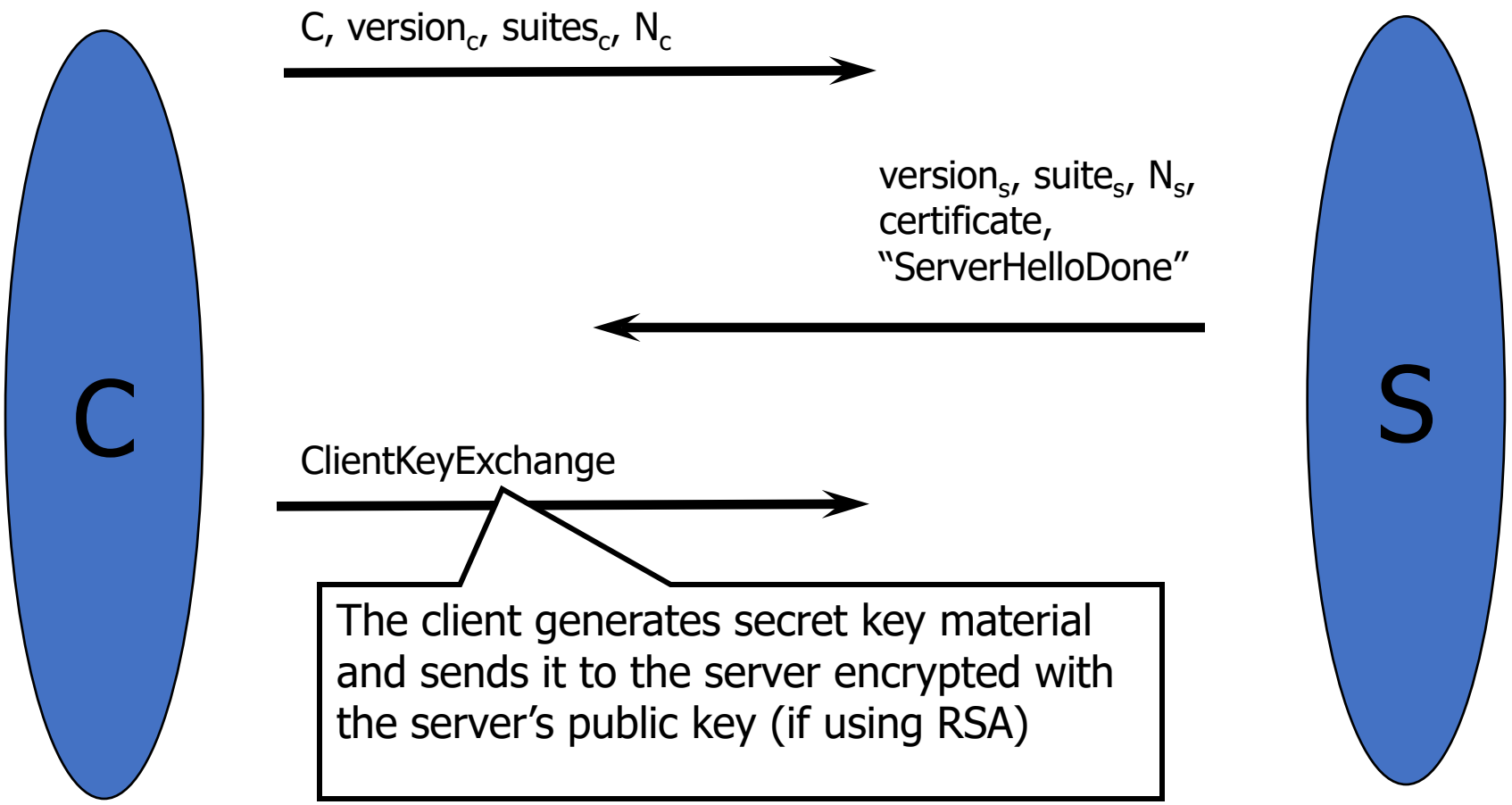
SERVER HELLO



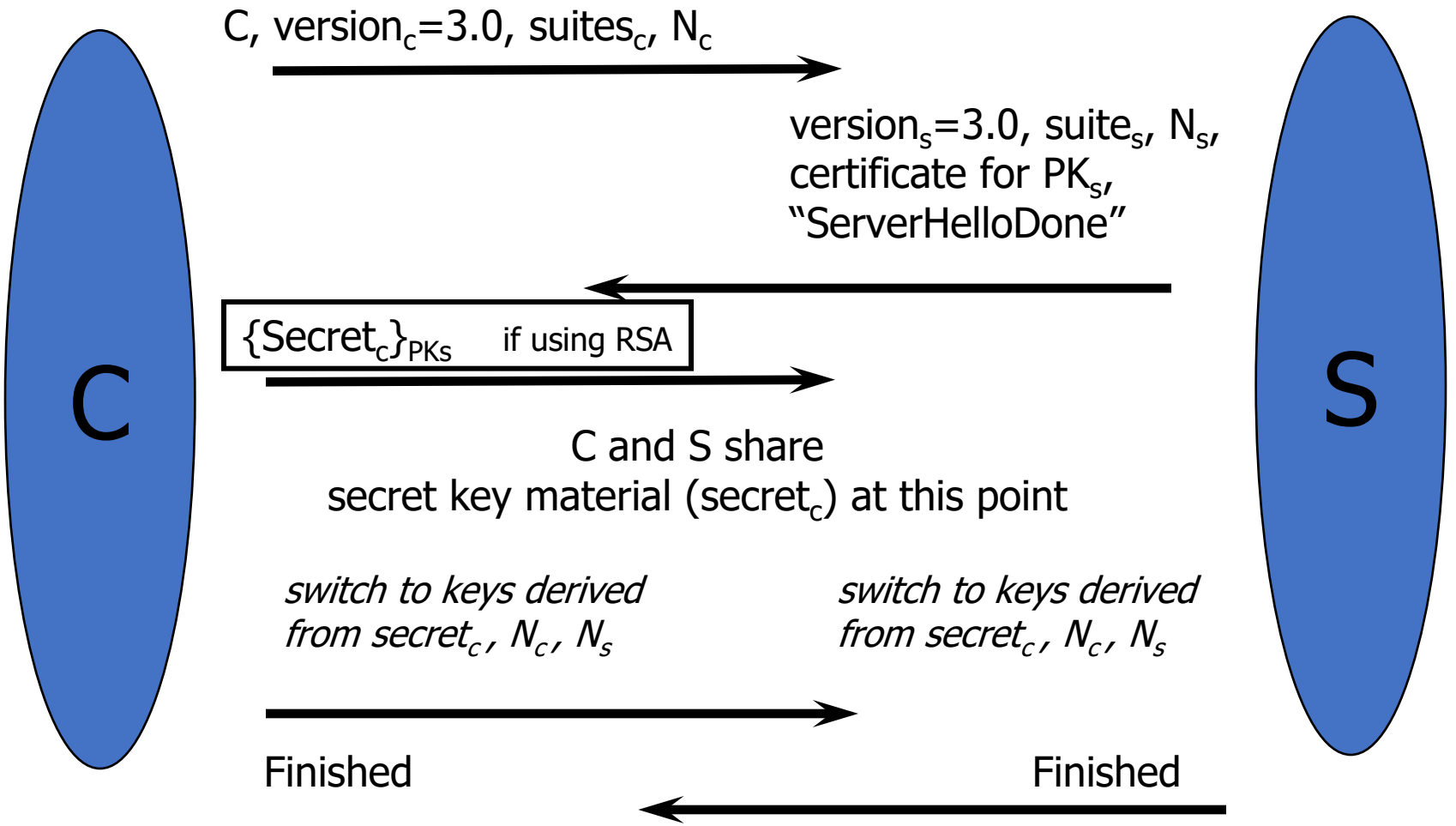
SERVER KEY EXCHANGE



CLIENT KEY EXCHANGE



"CORE" SSL 3.0 HANDSHAKE



SSL/TLS HANDSHAKE

CHECK EVERYTHING IN CERTIFICATE

- A certificate includes important information
  - Server's public key
  - Domain name
  - Issuer
  - Expiration date
  - …
- Must verify everything!

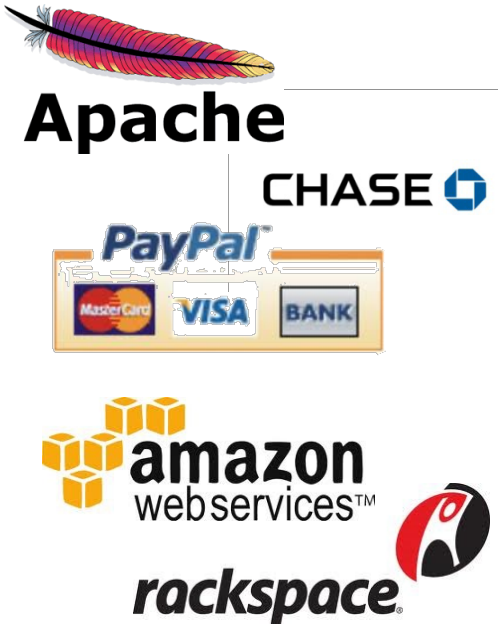
SSL/TLS HANDSHAKE: ATTACK EXAMPLES

Hello

I am Chase.com
Here is my certificate

Issued by GoDaddy to
AllYourSSLAreBelongTo.us

FAILING TO CHECK HOSTNAME



"Researchers at the University of Texas at Austin and Stanford University have discovered that poorly designed APIs used in SSL implementations are to blame for vulnerabilities in many critical non-browser software packages. Serious security vulnerabilities were found in programs such as Amazon's EC2 Java library, Amazon's and PayPal's merchant SDKs, Trillian and AIM instant messaging software, popular integrated shopping cart software packages, Chase mobile banking software, and several Android applications and libraries. SSL connections from these programs and many others are vulnerable to a man in the middle attack…"

- Threatpost (Oct 2012)

Major payment processing gateways, client software for cloud computing, integrated e-commerce software, etc.