Week 8

An Introduction to Cyber Security – CS 573

Instructor: Dr. Edward G. Amoroso
eamoroso@tag-cyber.com

Week 8

Required Week Eight Readings

1. “Blind Signatures for Untraceable Payments,” David Chaum
https://sceweb.sce.uhcl.edu/yang/teaching/csci5234WebSecurityFall2011/Chaum-blind-signatures.PDF

***2.* Finish *From CIA to APT: An Introduction to Cyber Security*, E. Amoroso & M. Amoroso**

LinkedIn: Edward Amoroso

Week 8: Key Distribution, Digital Signing, SSL, and Secure eCommerce

How are Keys Distributed?

Diffie-Hellman Key Exchange

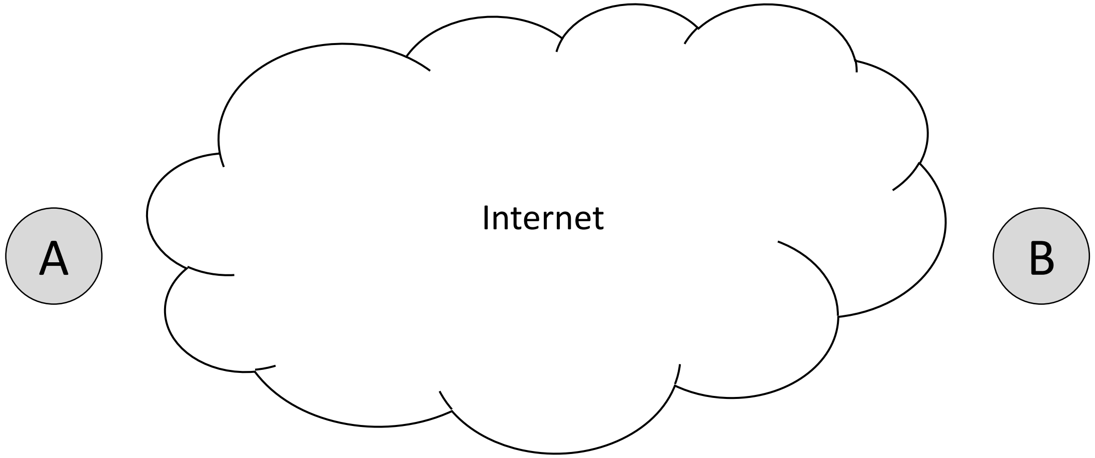
Does support symmetric key exchange

Does not support strong authentication

It is therefore vulnerable to man-in-the-middle attacks

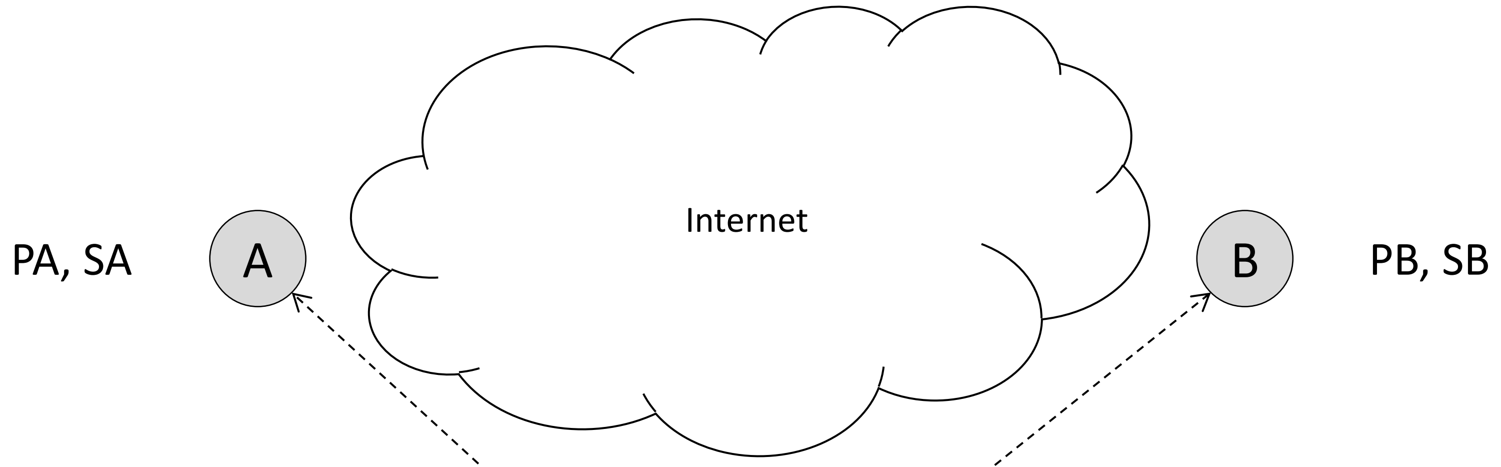
Something else is needed to support authenticated key exchange . . .

Public Key Distribution



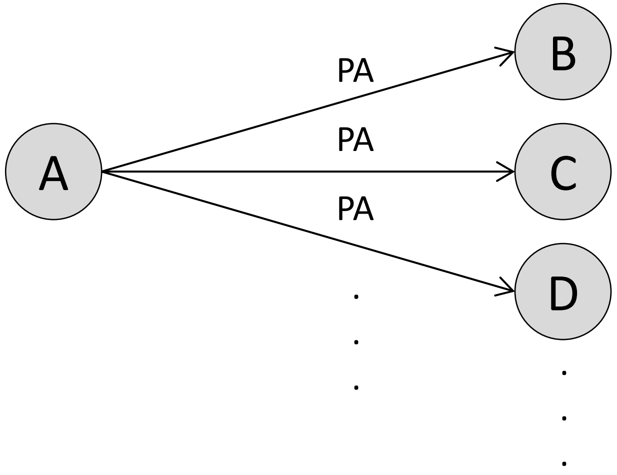
Week 8

Public Key Distribution



Initial State: A and B generate their own key pairs but do not possess other public keys

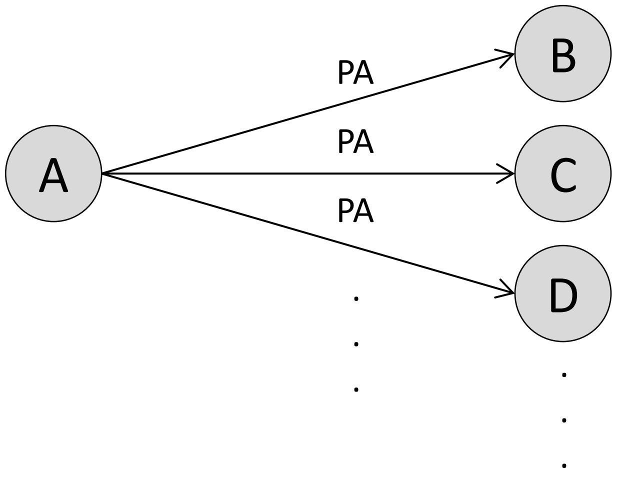
Public Key Distribution – Manual Distribution

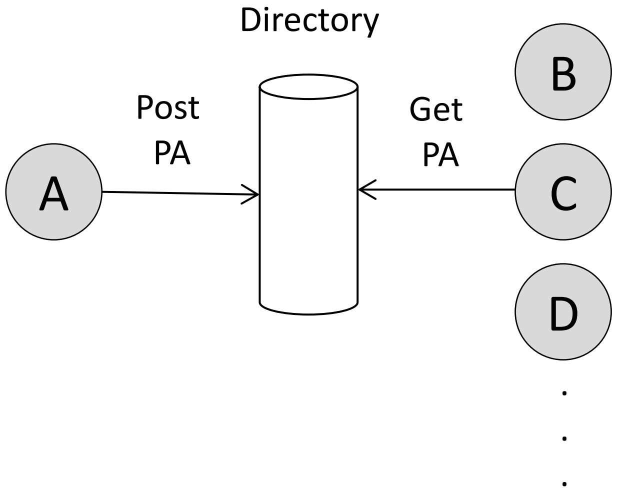


Manual Distribution:

- Easy, attach to email, etc.
- Does not scale across large groups
- One new participant to group of size X, requires X key actions

Public Key Distribution – Directory Post





Manual Distribution:

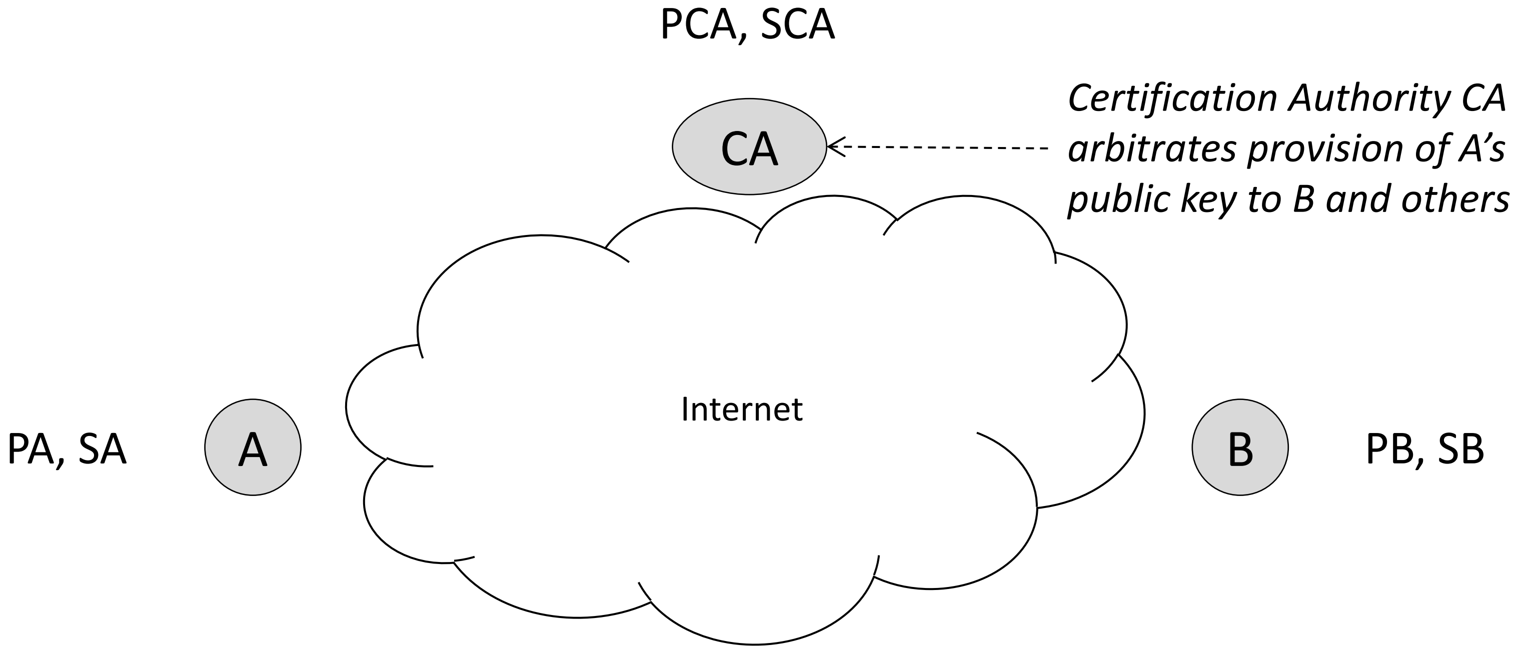
- Easy, attach to email, etc.
- Does not scale across large groups
- One new participant to group of size X, requires X key actions

Directory Post Distribution:

- Easy for enterprise directories
- Does not scale across large groups
- Vulnerable to outage – SPOF
- One new participant to group of size X, requires 1 post to directory

Week 8

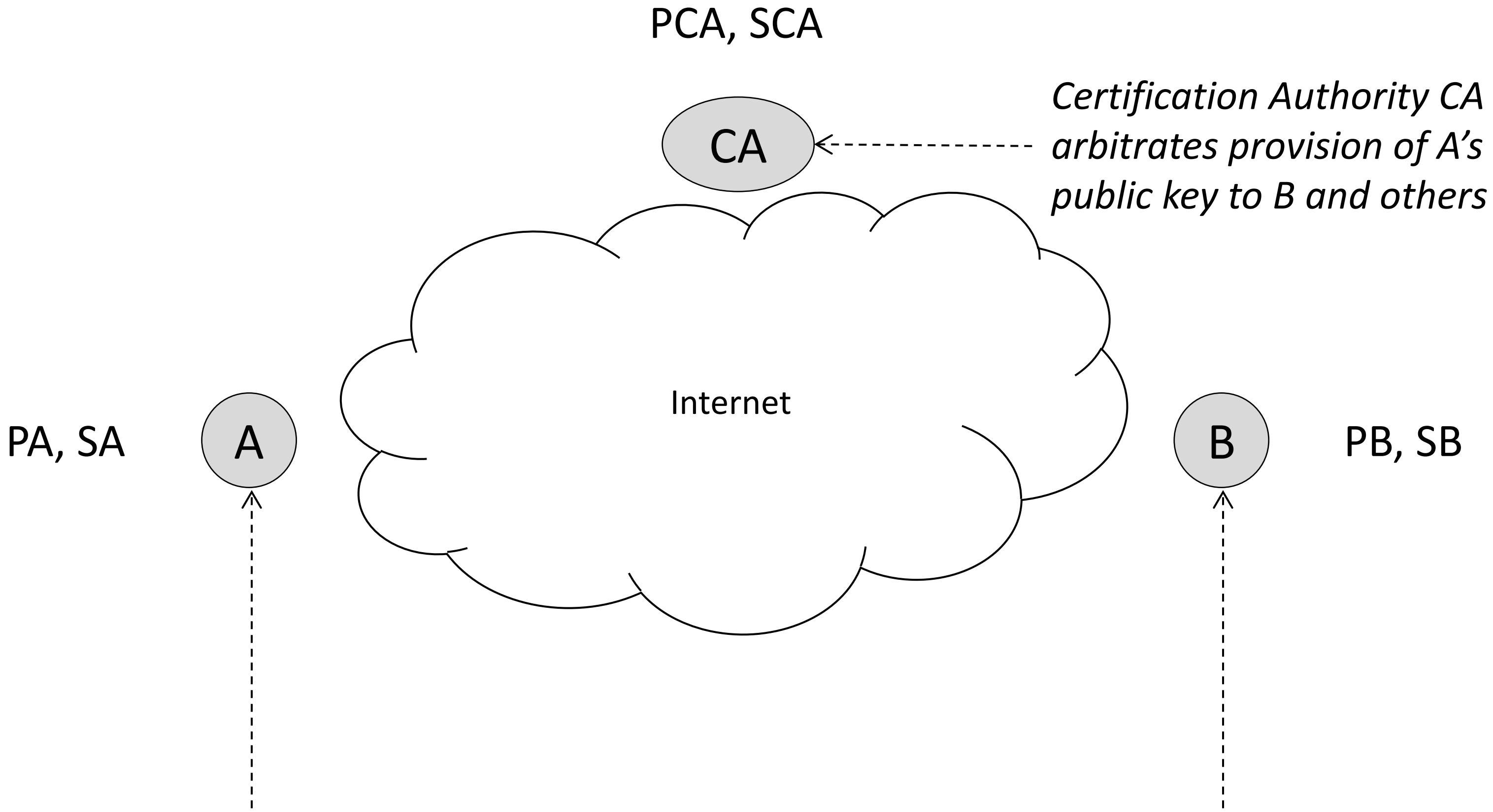
Public Key Distribution – Certification Authority



Certification Authority CA arbitrates provision of A's public key to B and others

Week 8

Public Key Distribution – Certification Authority

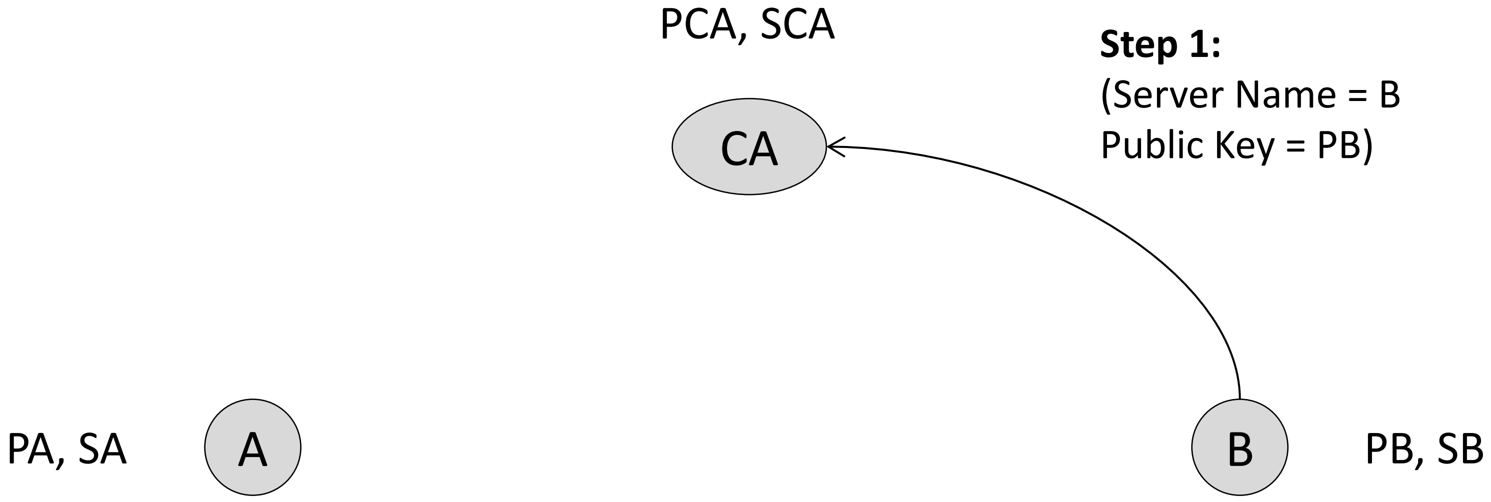


Certification Authority CA arbitrates provision of A's public key to B and others

Assume A is a Client Browser
"Wants to Buy"

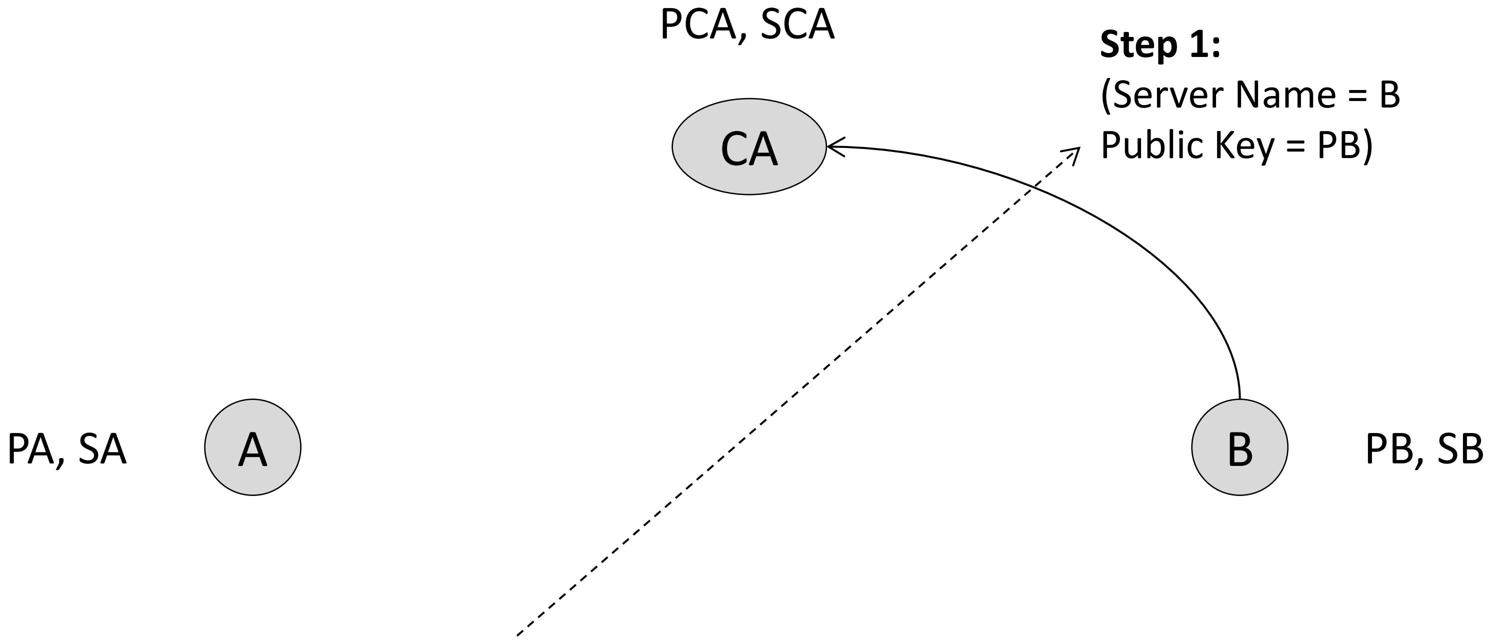
Assume B is an eCommerce Website
"Wants to Sell"

Public Key Distribution – Certification Authority



Step 1:
(Server Name = B
Public Key = PB)

Public Key Distribution – Certification Authority



PCA, SCA

CA

PA, SA

A

B

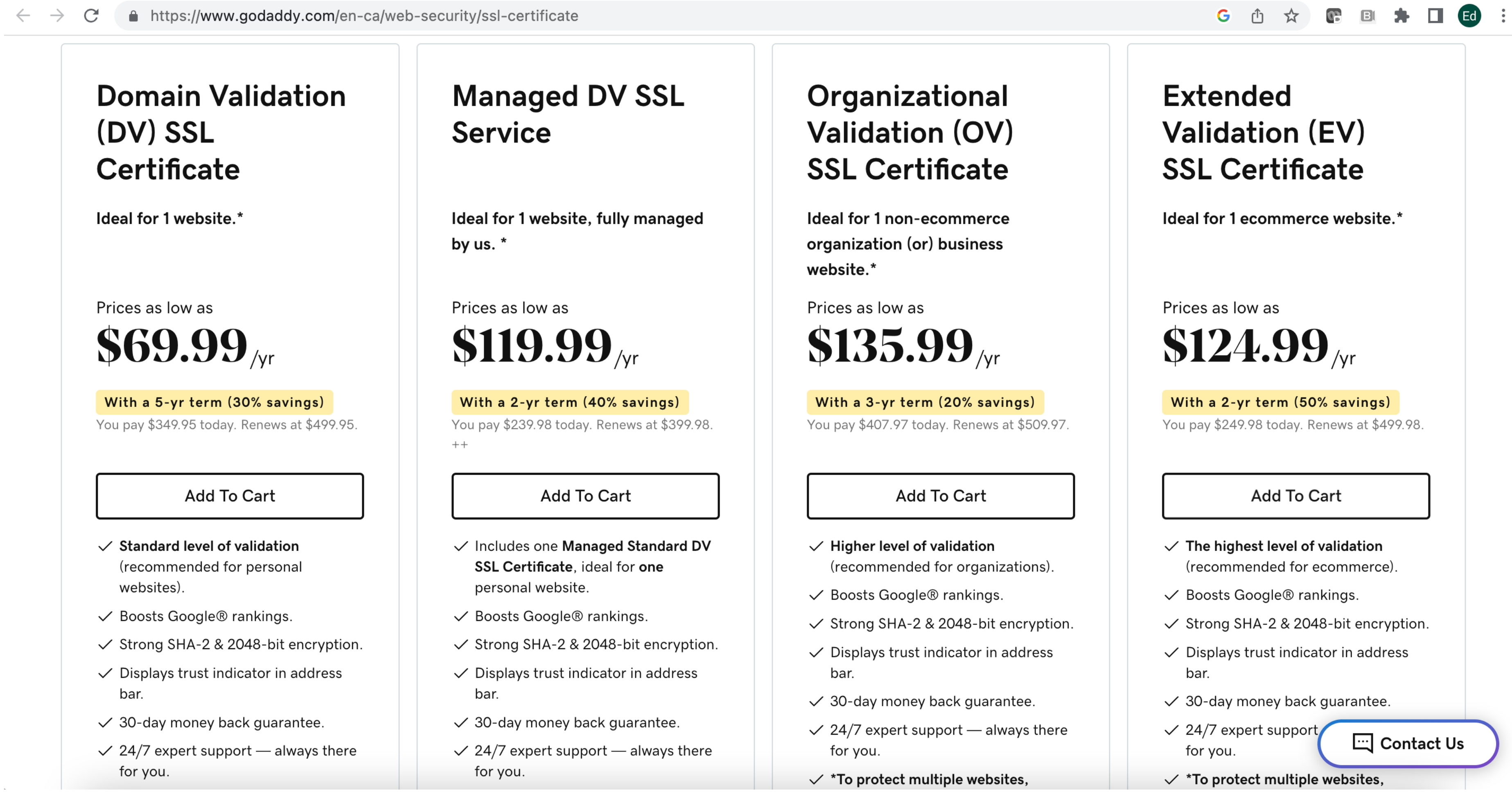
PB, SB

Step 1:
(Server Name = B
Public Key = PB)

Three Potential Assurance Levels Between B and CA:

- *Low*: Attributable Email from B’s Server to CA
- *Medium*: Out of Band Authentication of B’s Server by CA
- *High*: Thorough Vetting of B’s Server Administered by CA

Week 8


https://www.godaddy.com/en-ca/web-security/ssl-certificate

Domain Validation (DV) SSL Certificate

Ideal for 1 website.*

Prices as low as

$69.99 /yr

With a 5-yr term (30% savings)

You pay $349.95 today. Renews at $499.95.

Add To Cart

- ✓ **Standard level of validation** (recommended for personal websites).
- ✓ Boosts Google® rankings.
- ✓ Strong SHA-2 & 2048-bit encryption.
- ✓ Displays trust indicator in address bar.
- ✓ 30-day money back guarantee.
- ✓ 24/7 expert support — always there for you.

Managed DV SSL Service

Ideal for 1 website, fully managed by us. *

Prices as low as

$119.99 /yr

With a 2-yr term (40% savings)

You pay $239.98 today. Renews at $399.98. ++

Add To Cart

- ✓ Includes one **Managed Standard DV SSL Certificate**, ideal for **one** personal website.
- ✓ Boosts Google® rankings.
- ✓ Strong SHA-2 & 2048-bit encryption.
- ✓ Displays trust indicator in address bar.
- ✓ 30-day money back guarantee.
- ✓ 24/7 expert support — always there for you.

Organizational Validation (OV) SSL Certificate

Ideal for 1 non-ecommerce organization (or) business website.*

Prices as low as

$135.99 /yr

With a 3-yr term (20% savings)

You pay $407.97 today. Renews at $509.97.

Add To Cart

- ✓ **Higher level of validation** (recommended for organizations).
- ✓ Boosts Google® rankings.
- ✓ Strong SHA-2 & 2048-bit encryption.
- ✓ Displays trust indicator in address bar.
- ✓ 30-day money back guarantee.
- ✓ 24/7 expert support — always there for you.
- ✓ ***To protect multiple websites,**

Extended Validation (EV) SSL Certificate

Ideal for 1 ecommerce website.*

Prices as low as

$124.99 /yr

With a 2-yr term (50% savings)

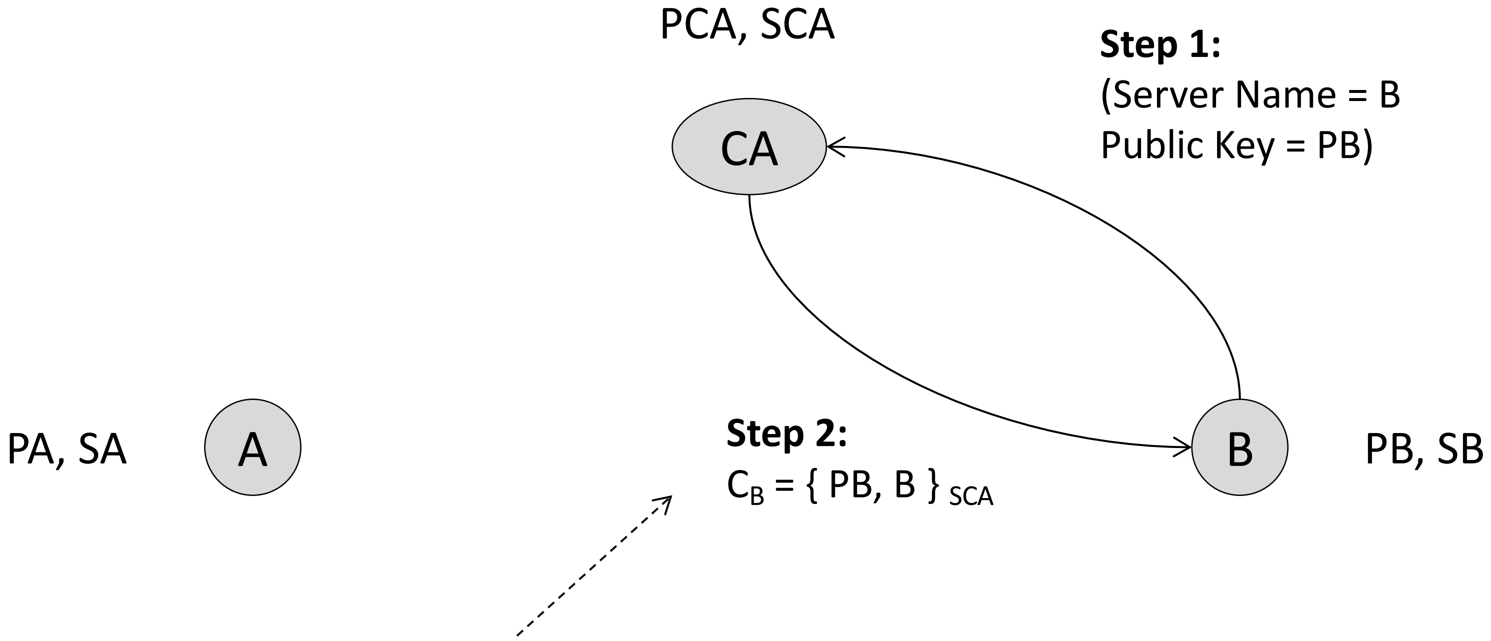
You pay $249.98 today. Renews at $499.98.

Add To Cart

- ✓ **The highest level of validation** (recommended for ecommerce).
- ✓ Boosts Google® rankings.
- ✓ Strong SHA-2 & 2048-bit encryption.
- ✓ Displays trust indicator in address bar.
- ✓ 30-day money back guarantee.
- ✓ 24/7 expert support for you.
- ✓ ***To protect multiple websites,**

Contact Us

Public Key Distribution – Certification Authority



PCA, SCA

Step 1:
(Server Name = B
Public Key = PB)

PA, SA

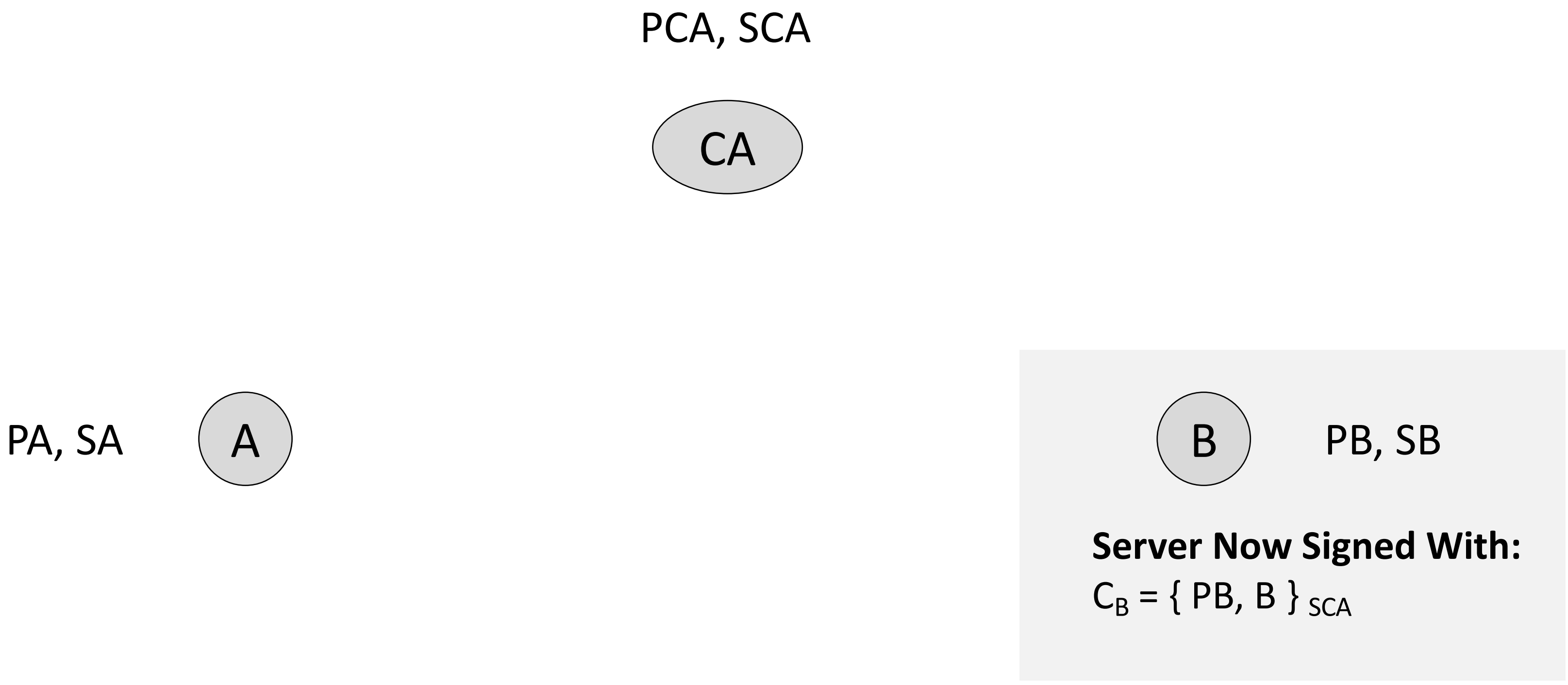
PB, SB

Step 2:
$C_B = \{ PB, B \}_{SCA}$

CA Sign’s the Server B with Certificate C_B:

- *Certificate follows X.509 v3 Standard*
- *Certificate encrypted with CA’s Private Key SCA*

Public Key Distribution – Certification Authority



PCA, SCA

CA

PA, SA

A

B

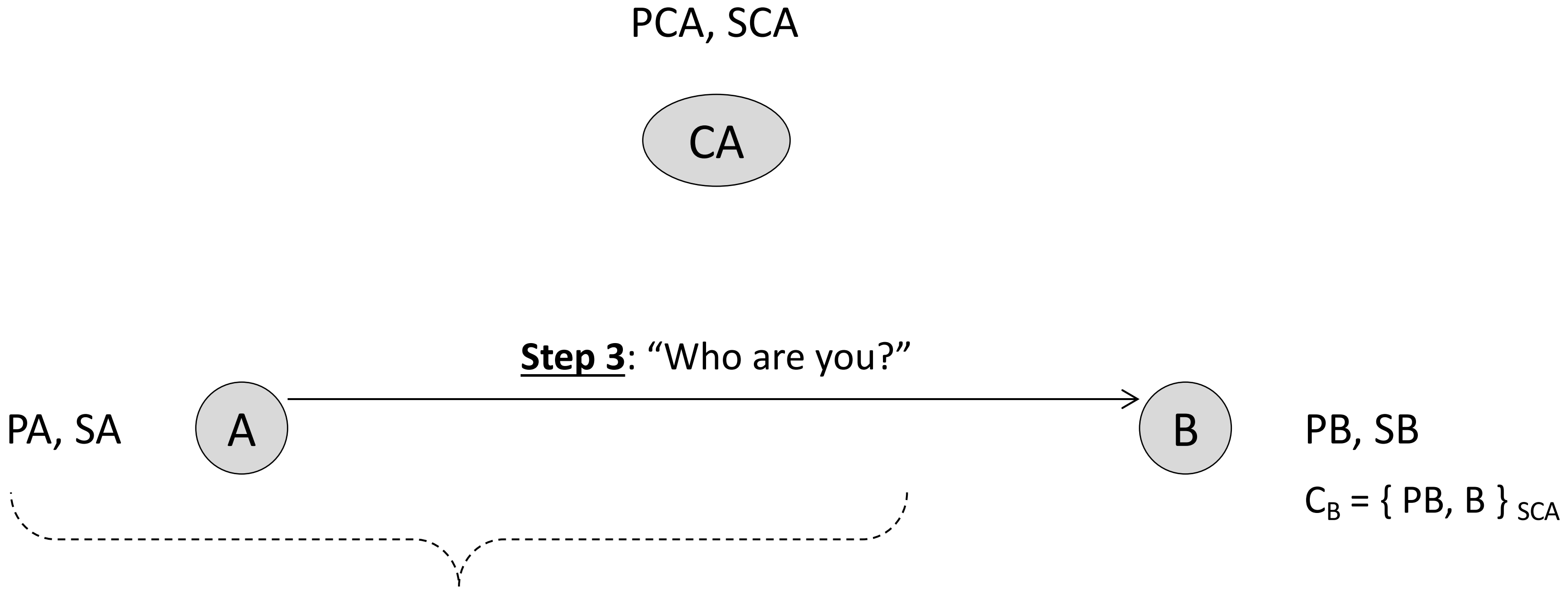
PB, SB

Server Now Signed With:

$C_B = \{ PB, B \}_{SCA}$

Week 8

Public Key Distribution – Certification Authority



PCA, SCA

CA

<u>Step 3</u>: “Who are you?”

PA, SA

A

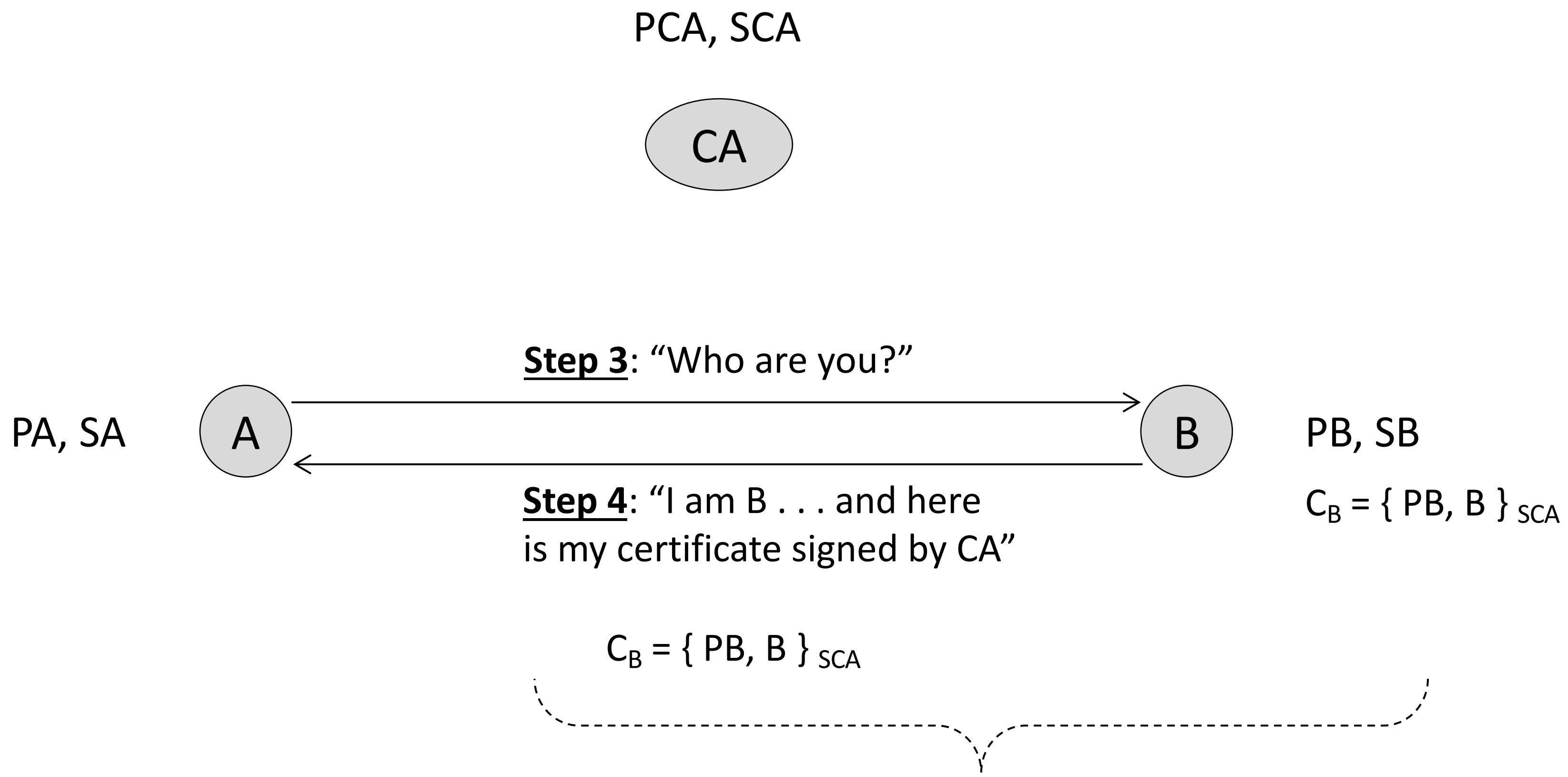
B

PB, SB

$C_B = \{ PB, B \}_{SCA}$

Server Authentication:
A has a browser and presumably wants
to buy something on B’s Website

Public Key Distribution – Certification Authority



PCA, SCA

<u>Step 3</u>: “Who are you?”

PA, SA

PB, SB

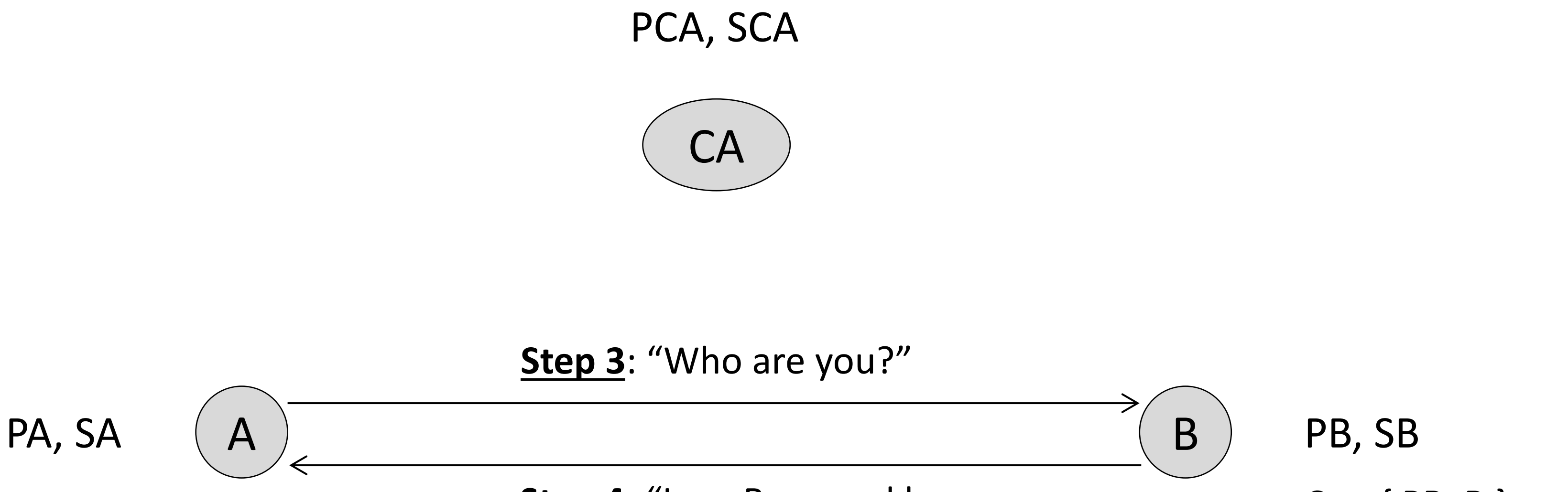
$C_B = \{ PB, B \}_{SCA}$

<u>Step 4</u>: “I am B . . . and here is my certificate signed by CA”

$C_B = \{ PB, B \}_{SCA}$

Server Response:
B send its certificate to A in order to authenticate and send PB for encryption

Public Key Distribution – Certification Authority



PCA, SCA

CA

Step 3: “Who are you?”

PA, SA

A

B

PB, SB

$C_B = \{ PB, B \}_{SCA}$

Step 4: “I am B . . . and here is my certificate signed by CA”

$C_B = \{ PB, B \}_{SCA}$

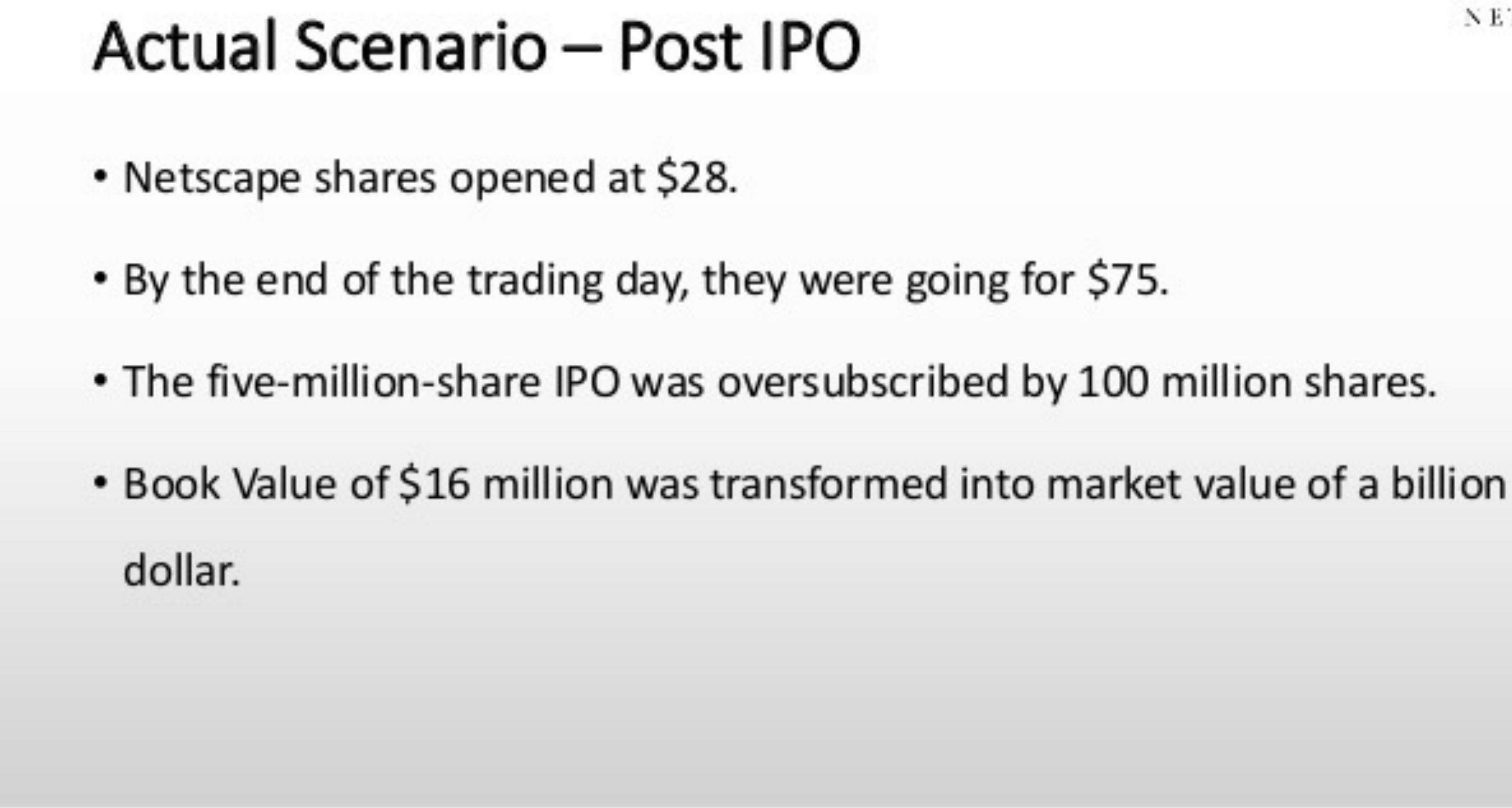
Step 5: “A needs the public key PCA of CA to decrypt:

$\{ \{ PB, B \}_{SCA} \}_{PCA}$ ➔ PB

A’s Dilemma:
How does it get PCA into its browser to decrypt the certificate signed by CA?

How Did Netscape Solve the CA Public Key Problem?

Netscape’s Historic IPO



Actual Scenario – Post IPO

- Netscape shares opened at $28.
- By the end of the trading day, they were going for $75.
- The five-million-share IPO was oversubscribed by 100 million shares.
- Book Value of $16 million was transformed into market value of a billion dollar.

NETSCAPE

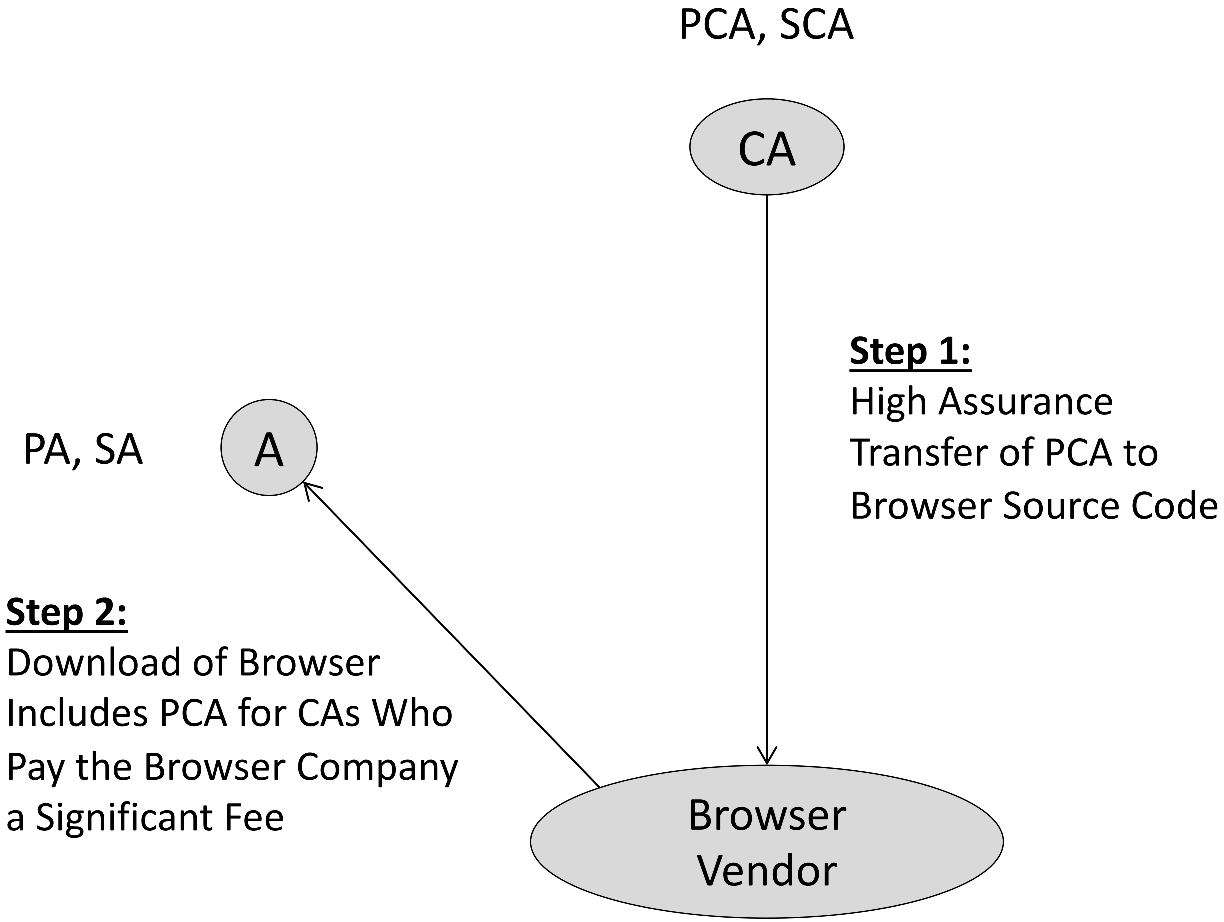
Resulting Protocol: Secure Sockets Layer (SSL)

TIME
TV VIOLENCE AND SEX:
Will the V Chip Help?
THE GOLDEN GEEKS
They invent. They start companies. And the stock market has made them INSTANTAIRES. Who are they? How do they live? And what do they mean for America's future?
Netscape's
Marc Andreessen

Marc Andreessen
Netscape Browser
Founder and Internet
Billionaire Shown
in Mid-1990's

Week 8

Solution: Embedding Certificates into Browsers

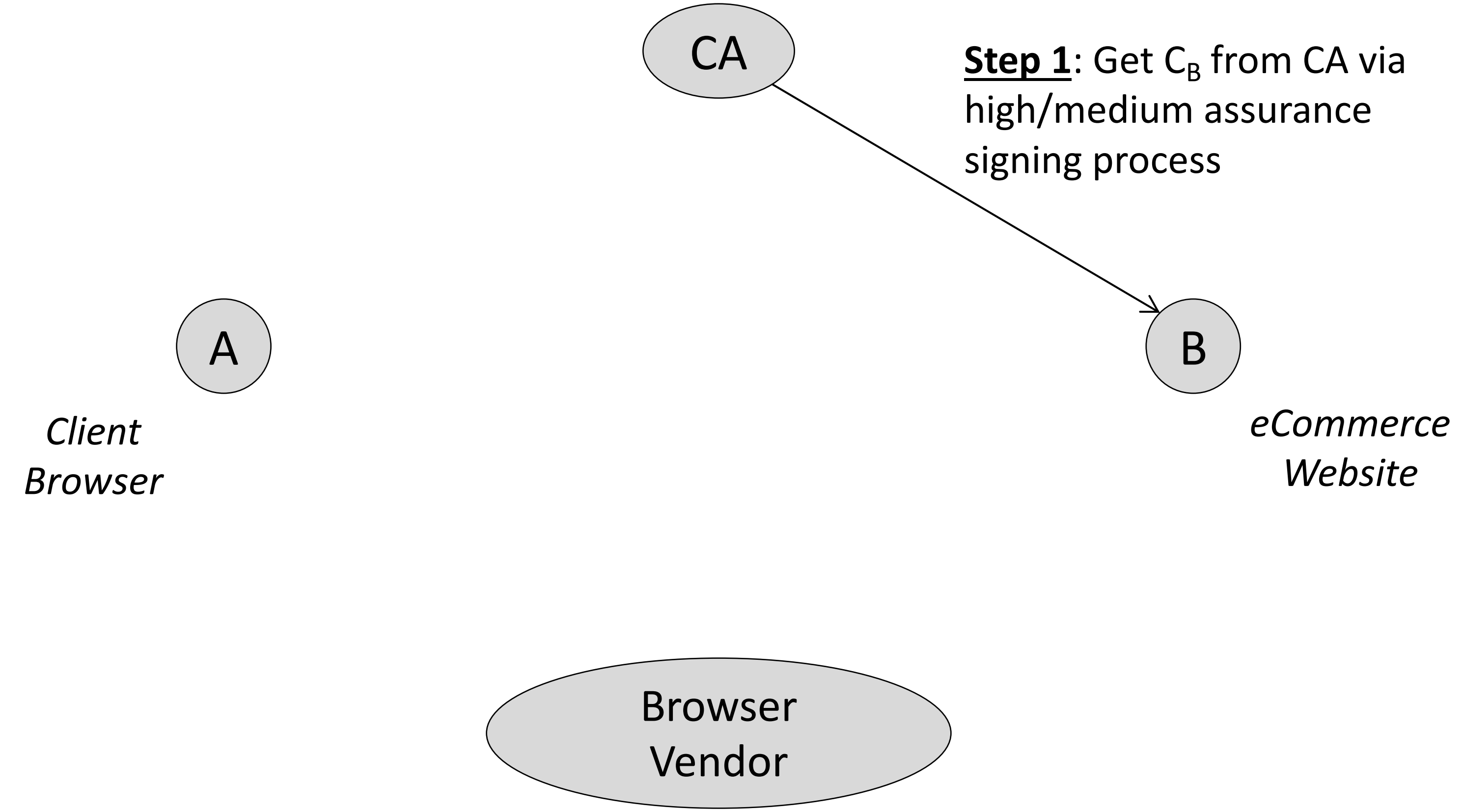


Step 1:
High Assurance Transfer of PCA to Browser Source Code

Step 2:
Download of Browser Includes PCA for CAs Who Pay the Browser Company a Significant Fee

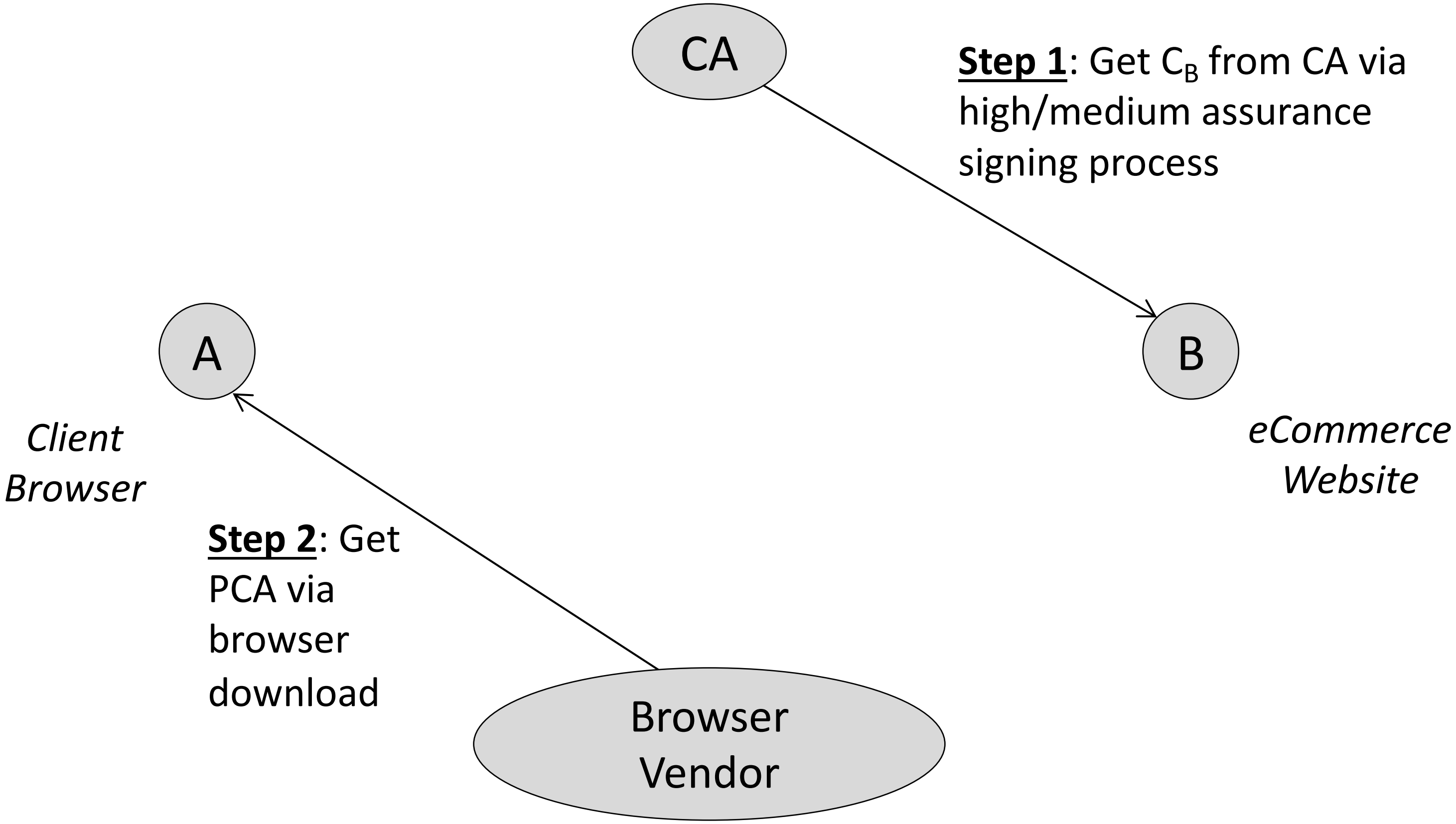
Week 8

SSL PKI/CA – Secure eCommerce



Step 1: Get C_B from CA via high/medium assurance signing process

SSL PKI/CA – Secure eCommerce



CA

Step 1: Get C_B from CA via high/medium assurance signing process

A

Client Browser

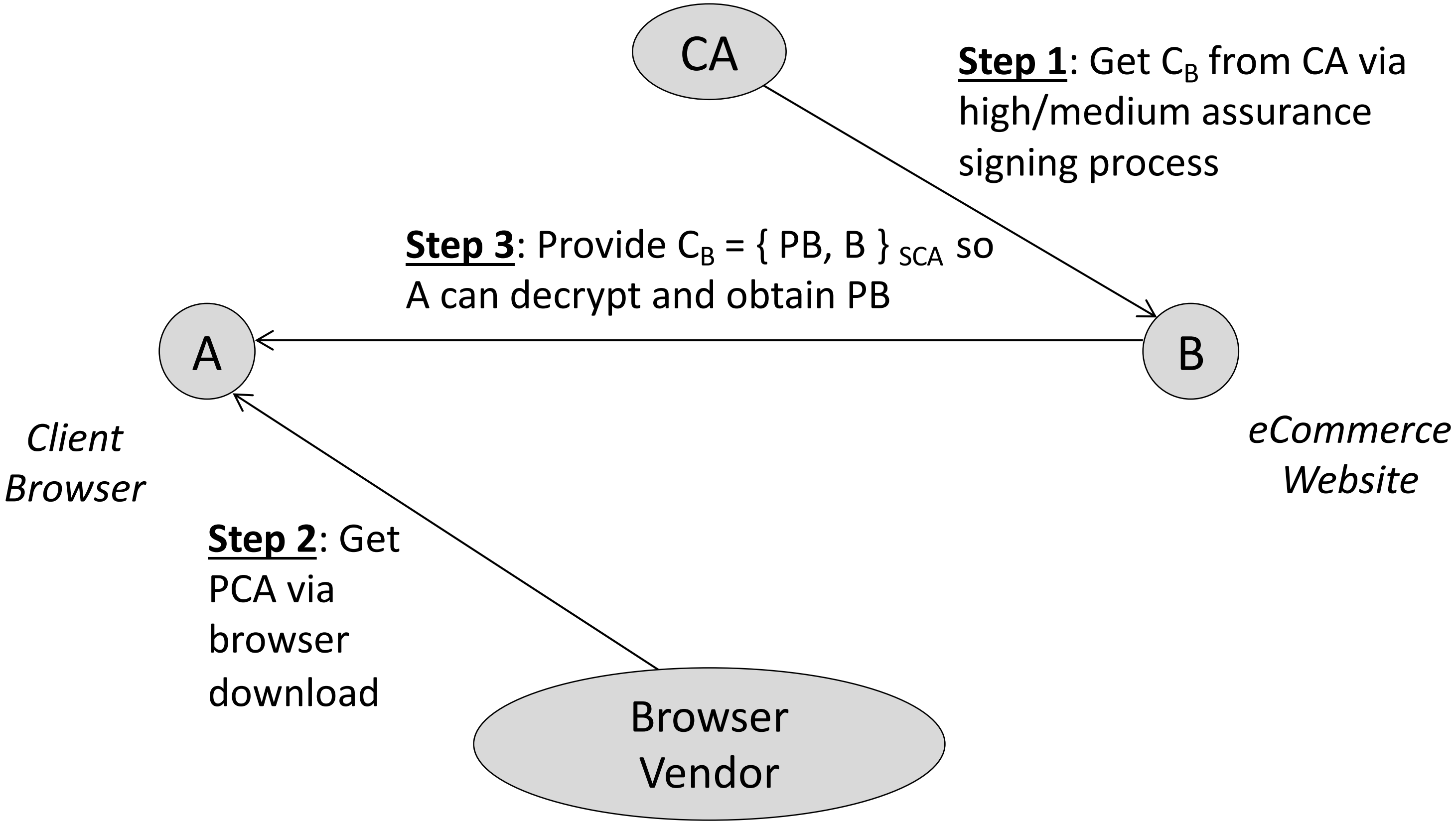
B

eCommerce Website

Step 2: Get PCA via browser download

Browser Vendor

SSL PKI/CA – Secure eCommerce



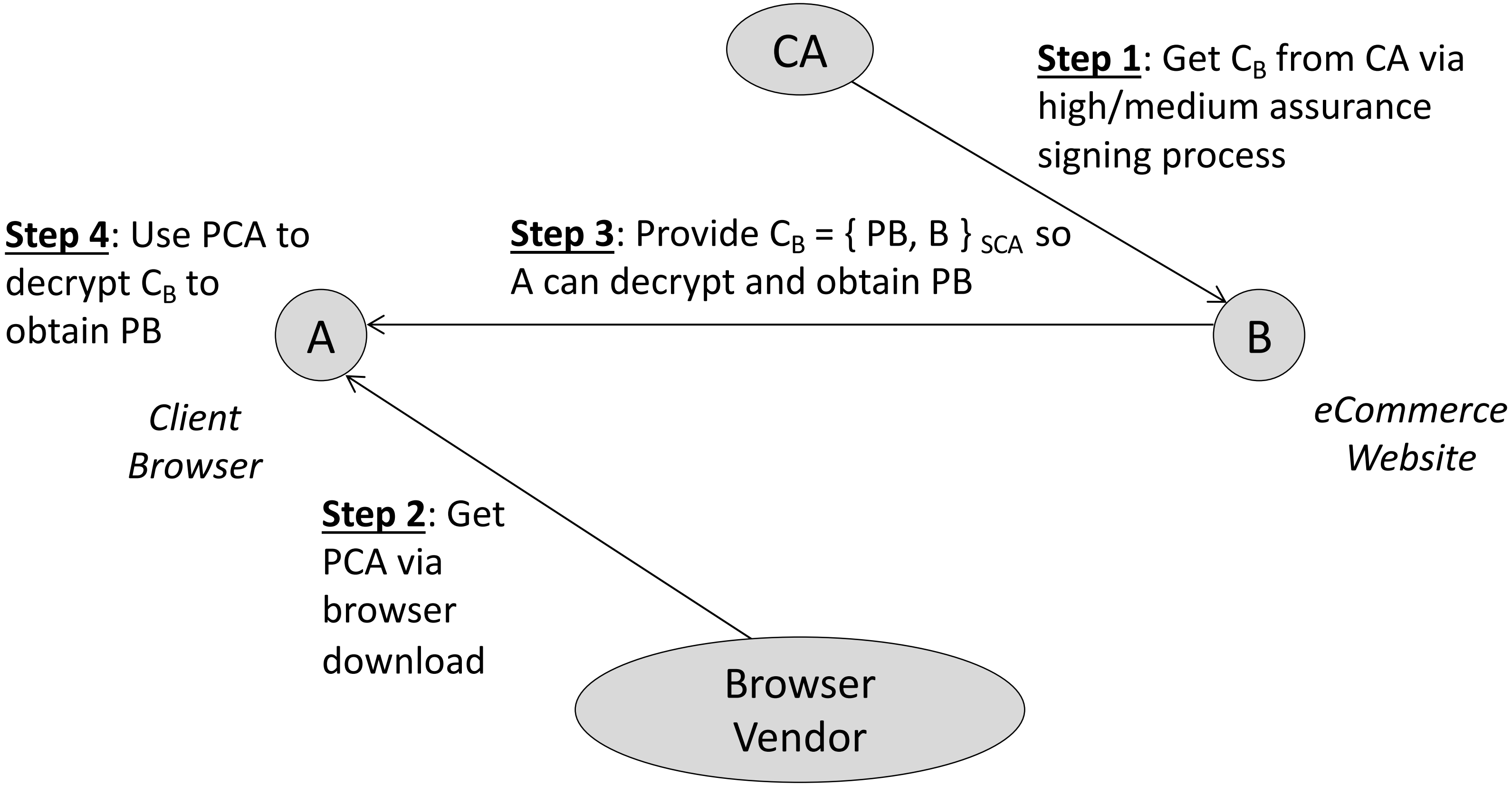
Step 1: Get C_B from CA via high/medium assurance signing process

Step 3: Provide $C_B = \{ PB, B \}_{SCA}$ so A can decrypt and obtain PB

Step 2: Get PCA via browser download

Week 8

SSL PKI/CA – Secure eCommerce



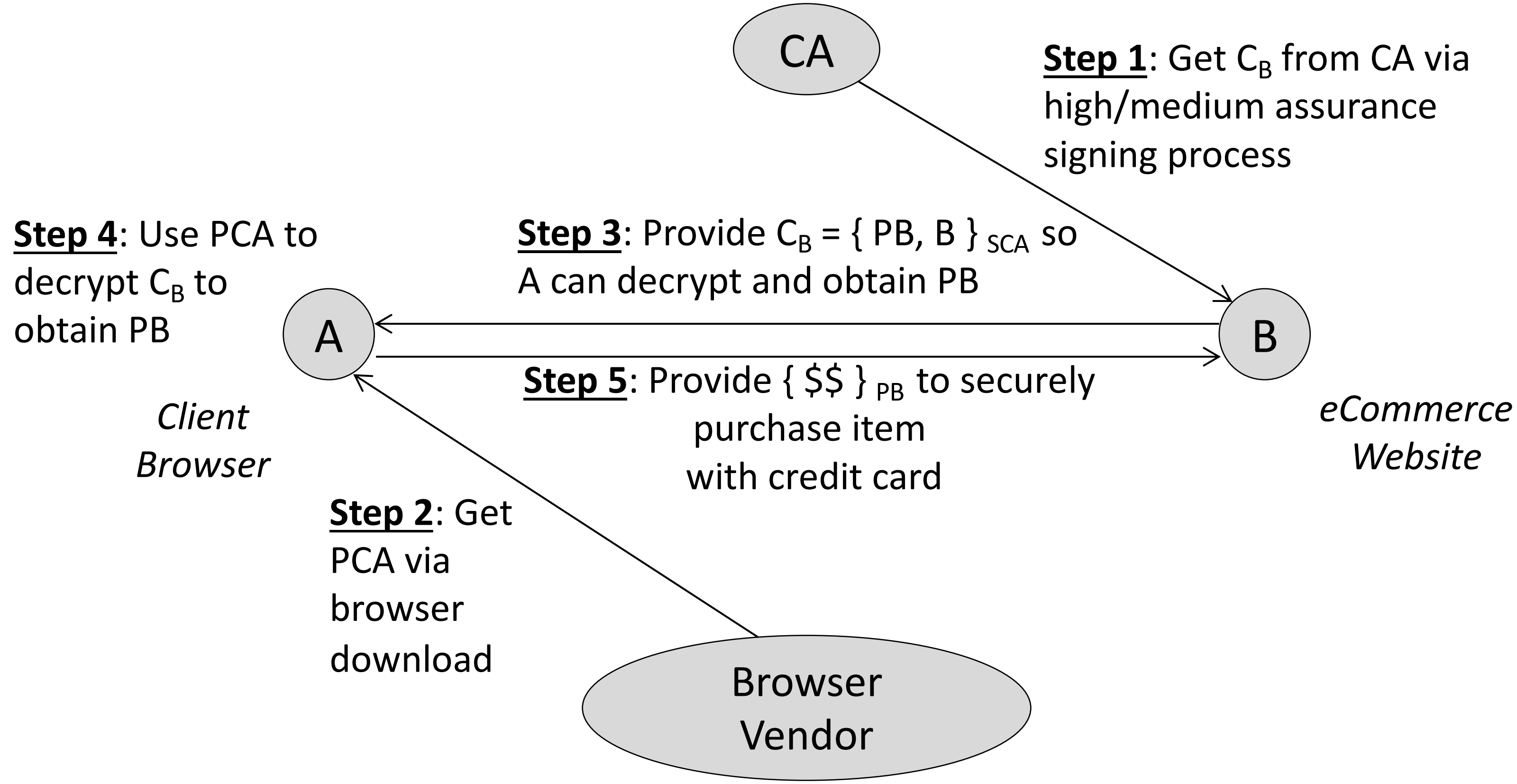
Step 1: Get C_B from CA via high/medium assurance signing process

Step 2: Get PCA via browser download

Step 3: Provide $C_B = \{ PB, B \}_{SCA}$ so A can decrypt and obtain PB

Step 4: Use PCA to decrypt C_B to obtain PB

SSL PKI/CA – Secure eCommerce



CA

Step 1: Get C_B from CA via high/medium assurance signing process

Step 4: Use PCA to decrypt C_B to obtain PB

Step 3: Provide $C_B = \{ PB, B \}_{SCA}$ so A can decrypt and obtain PB

A

Client Browser

B

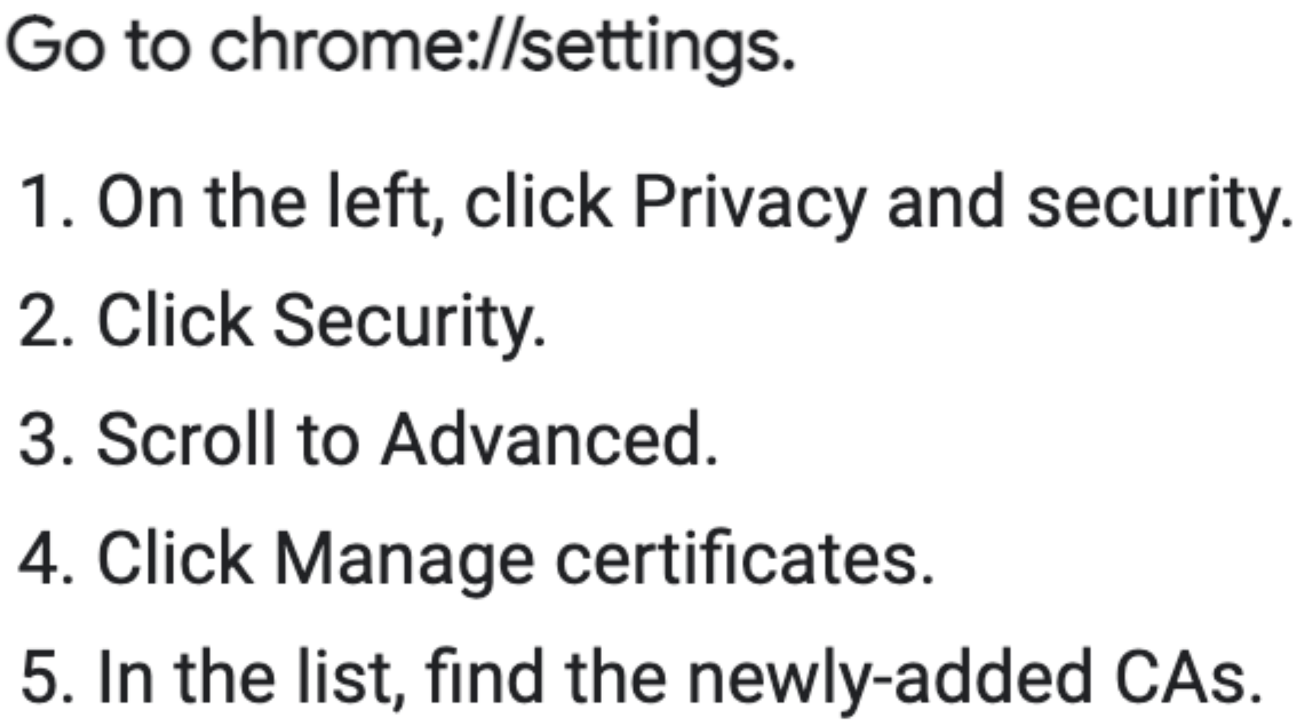
eCommerce Website

Step 5: Provide $\{ \$\$ \}_{PB}$ to securely purchase item with credit card

Step 2: Get PCA via browser download

Browser Vendor

Week 8



Go to chrome://settings.

1. On the left, click Privacy and security.
2. Click Security.
3. Scroll to Advanced.
4. Click Manage certificates.
5. In the list, find the newly-added CAs.

Reviewing Certificates in Your Chrome Browser

Transport Layer Security (TLS)

- TLS 1.0: An upgrade to SSL v3.0 released in January 1999; it allows connection downgrade to SSL v3.0 without needing a protocol change if necessary.

- TLS 1.1: TLS 1.1 released in April 2006 to update the TLS v1.0 version, which added protection against CBC (Cipher Block Chaining) attacks.

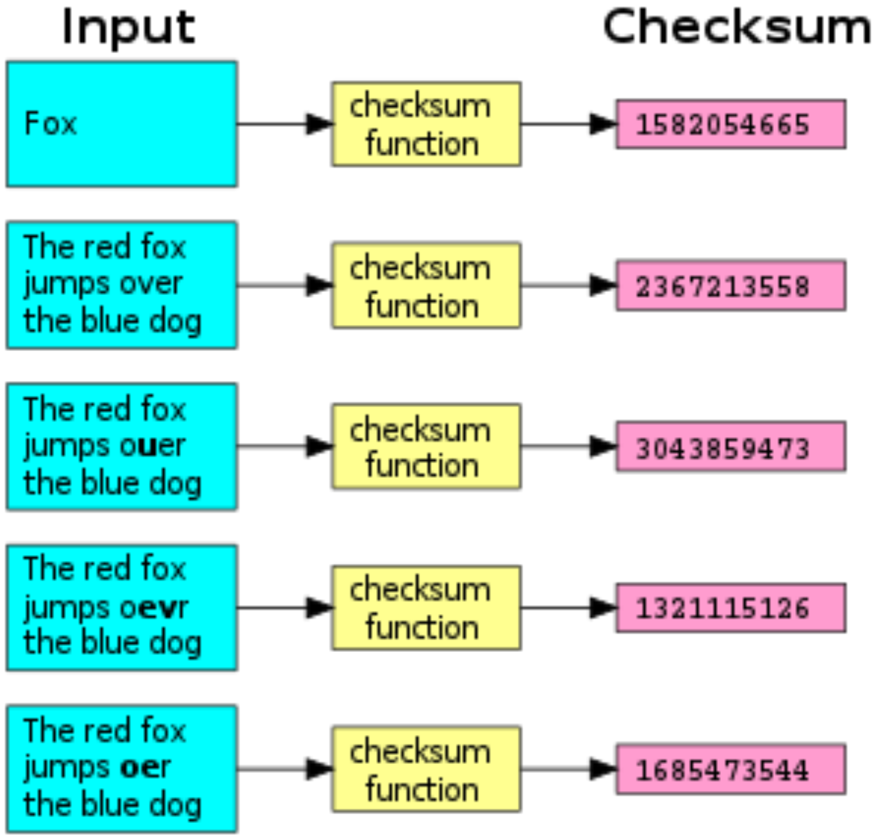
- TLS 1.2: TLS v1.2 released in 2008, allows the specification of hash and algorithm used by both client and server and authenticated encryption with extra data modes for more support. TLS 1.2 can verify length based on cipher suite type, making it much harder to relay attack messages because they are not formatted correctly.

- TLS 1.3: Newest version of TLS with MD5 hashing (SHA-224 support no longer used); digital signatures must be required for earlier configuration with key exchange methods to ensure Perfect Forward Secrecy in case there are public keys involved during this process handshake messages will now be encrypted.

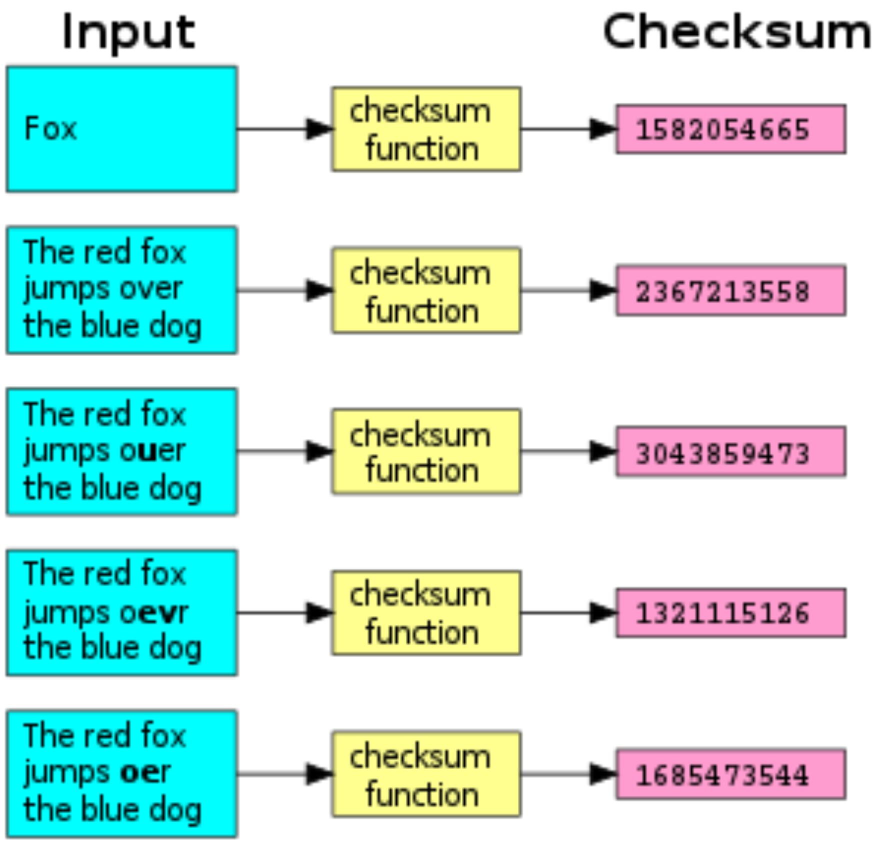
Week 8

How Does Hashing Work?

Unix **`cksum`** function

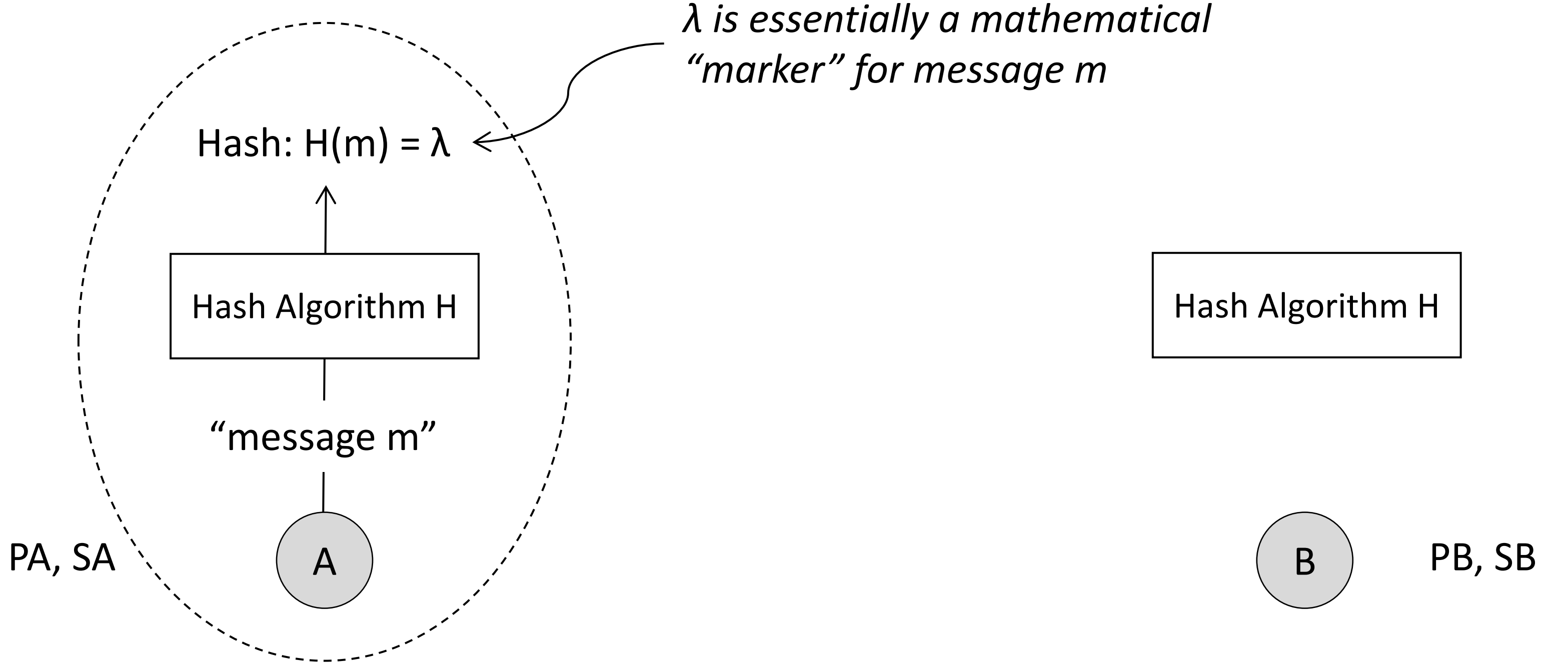

Input
Checksum
Fox
checksum function
1582054665
The red fox jumps over the blue dog
checksum function
2367213558
The red fox jumps o**u**er the blue dog
checksum function
3043859473
The red fox jumps o**ev**r the blue dog
checksum function
1321115126
The red fox jumps o**e**r the blue dog
checksum function
1685473544

Unix **cksum** function



- ***Hash Algorithm*: "Variable length input" (domain) to "fixed length output" (co-domain)**
- ***Hash Algorithm + Keys = Message Digest Algorithm***

Hashing for Digital Signature

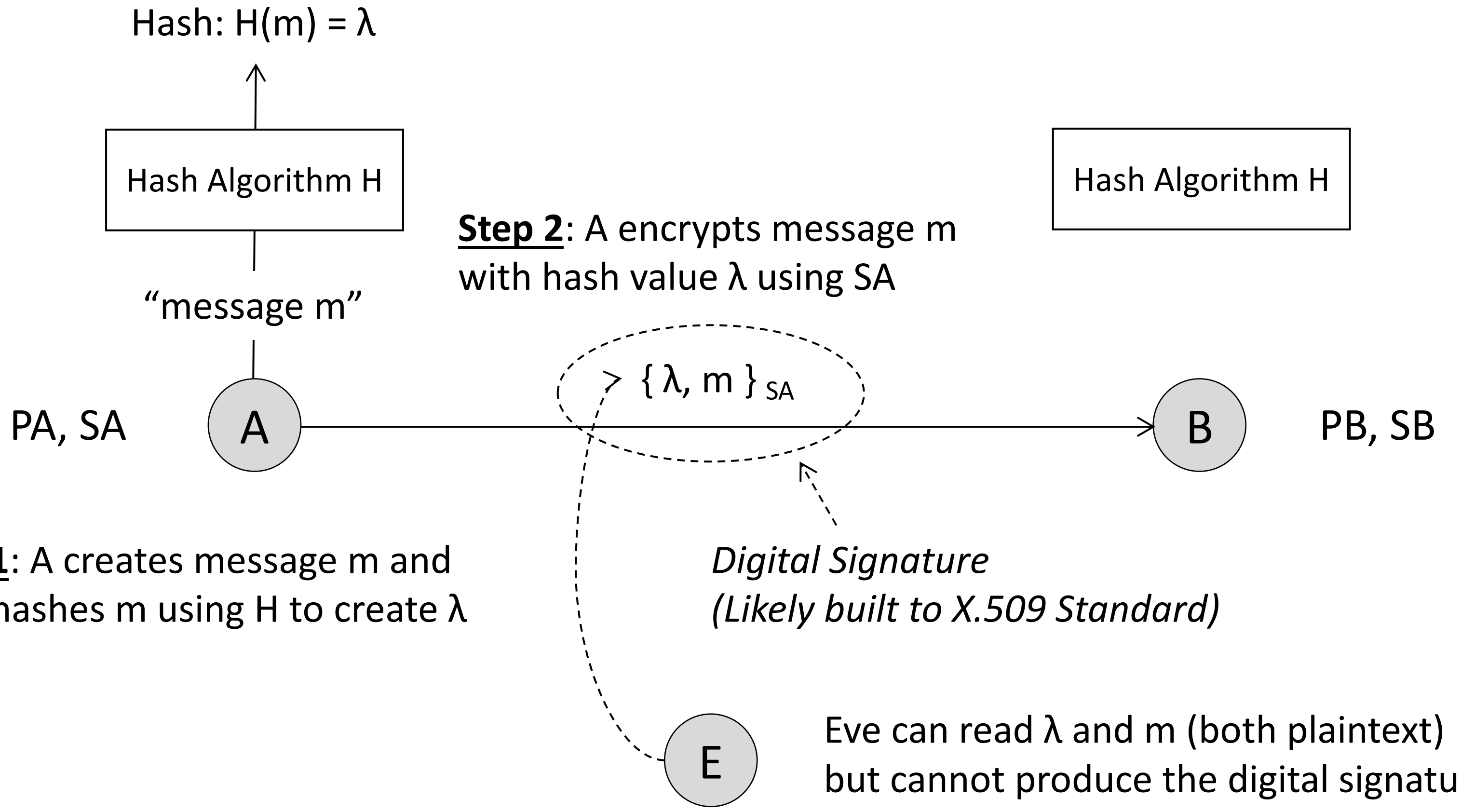


λ is essentially a mathematical "marker" for message m

Step 1: A creates message m and then hashes m using H to create λ

Week 8

Hashing for Digital Signature



Step 2: A encrypts message m with hash value λ using SA

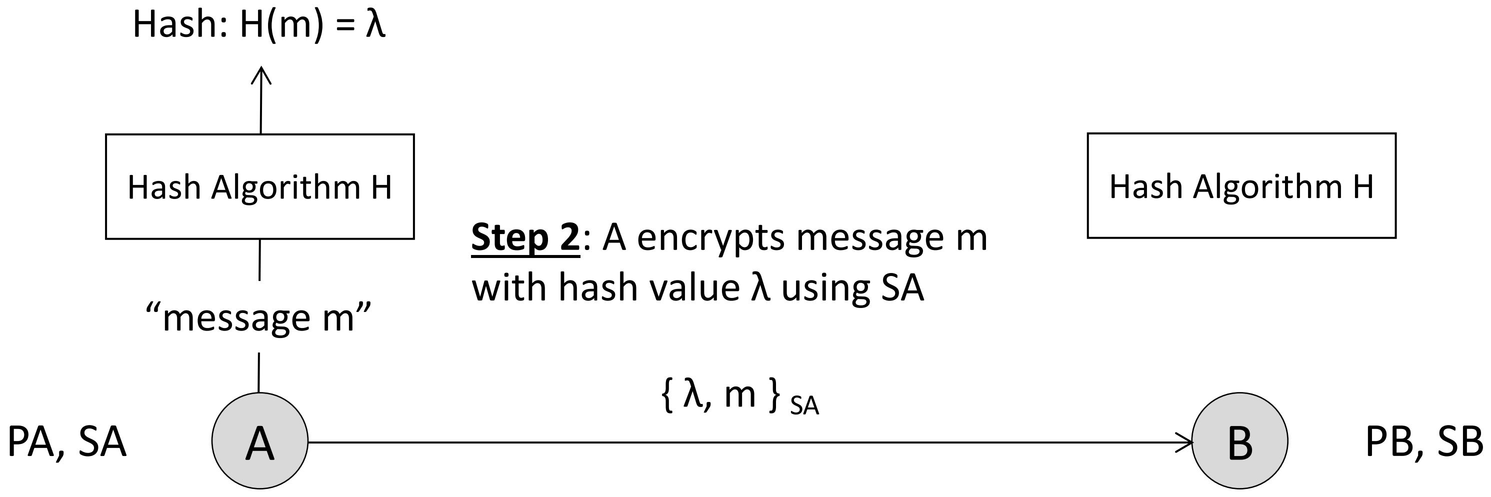
Step 1: A creates message m and then hashes m using H to create λ

Digital Signature (Likely built to X.509 Standard)

Eve can read λ and m (both plaintext) but cannot produce the digital signature (doesn't have SA)

Week 8

Hashing for Digital Signature



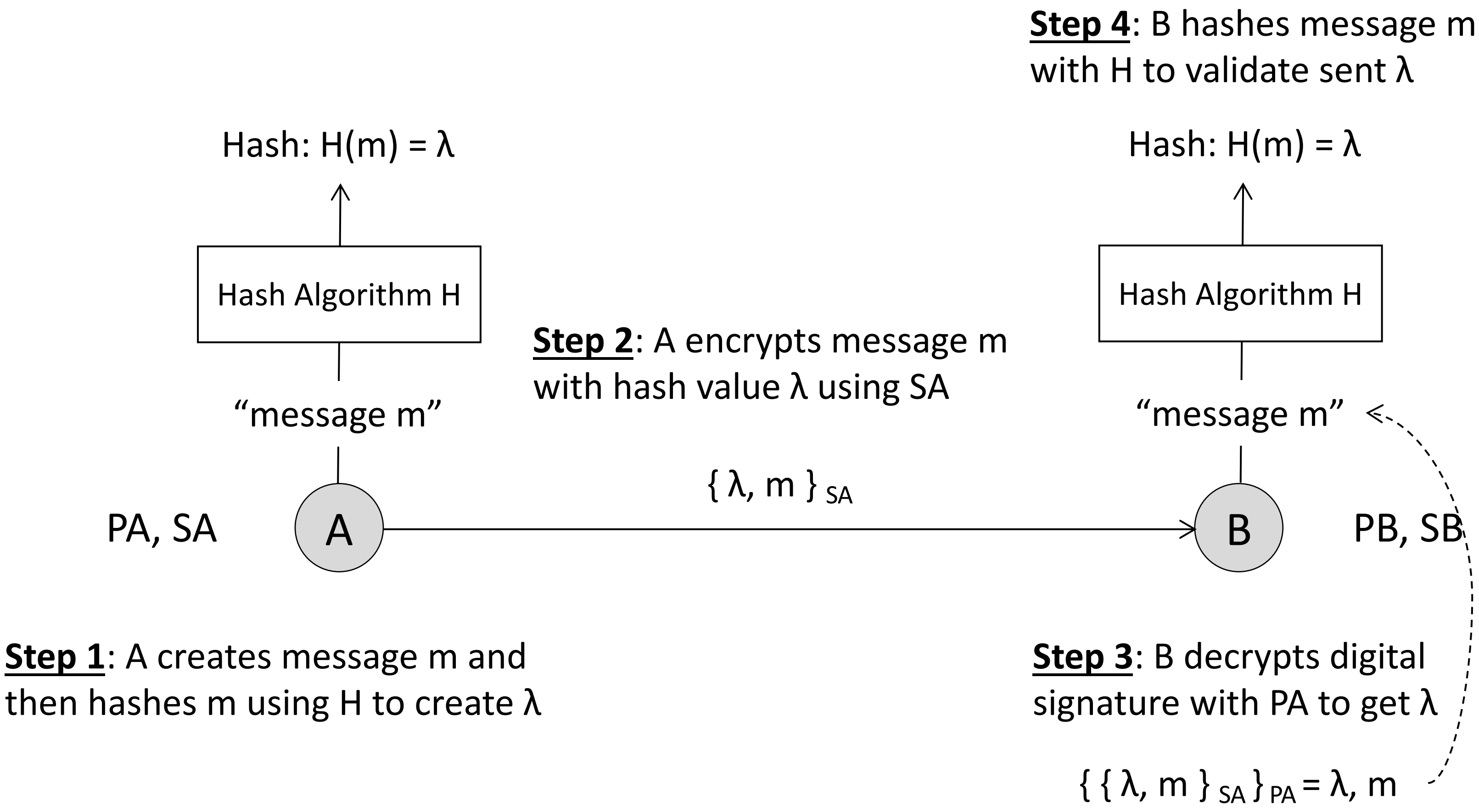
Step 2: A encrypts message m with hash value λ using SA

Step 1: A creates message m and then hashes m using H to create λ

Step 3: B decrypts digital signature with PA to get λ

$$\{ \{ \lambda, m \}_{SA} \}_{PA} = \lambda, m$$

Hashing for Digital Signature



Step 1: A creates message m and then hashes m using H to create λ

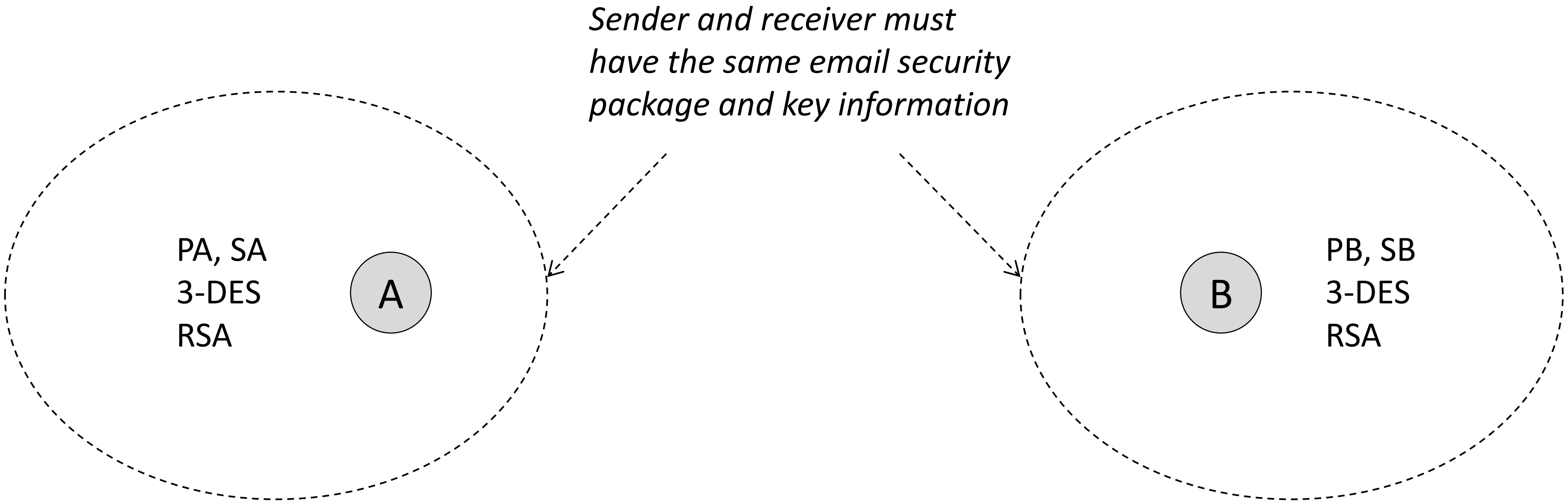
Step 2: A encrypts message m with hash value λ using SA

Step 3: B decrypts digital signature with PA to get λ

Step 4: B hashes message m with H to validate sent λ

How is Email Secured?

Secret Email



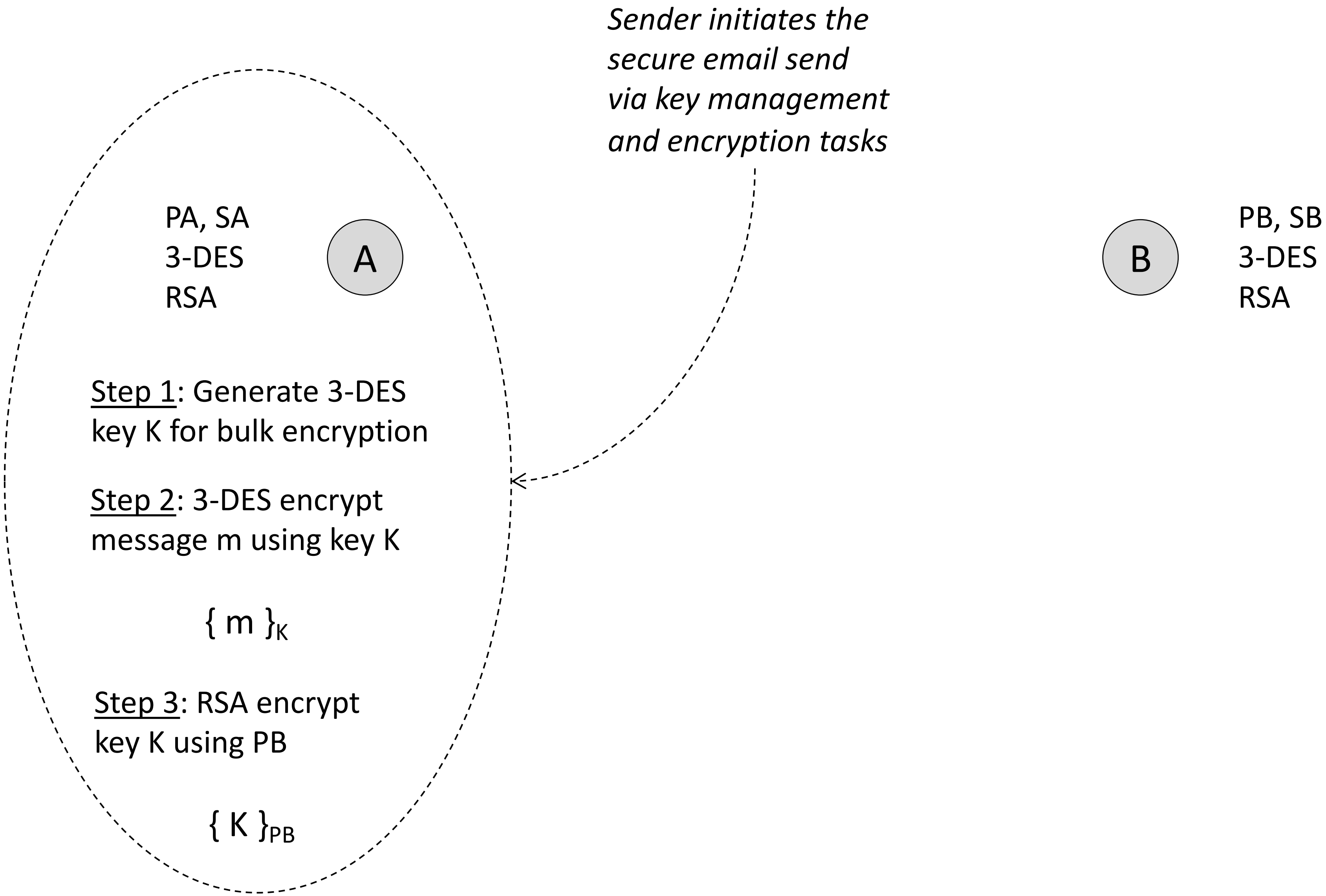
Sender and receiver must have the same email security package and key information

PA, SA
3-DES
RSA

PB, SB
3-DES
RSA

Week 8

Secret Email



Sender initiates the secure email send via key management and encryption tasks

PA, SA
3-DES
RSA

PB, SB
3-DES
RSA

Step 1: Generate 3-DES key K for bulk encryption

Step 2: 3-DES encrypt message m using key K

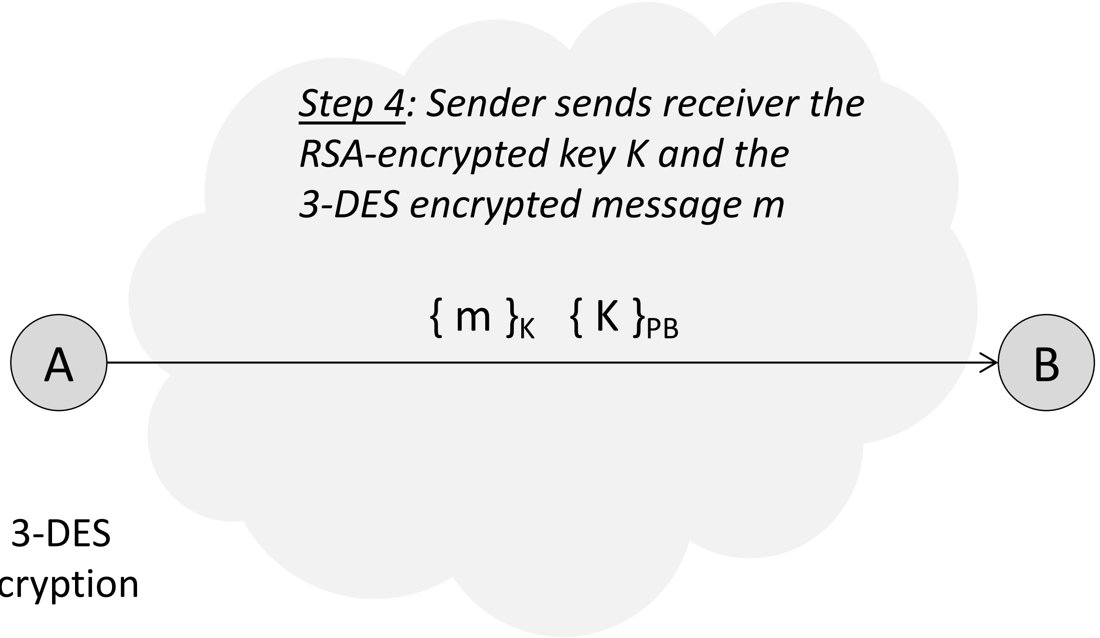
$\{ m \}_K$

Step 3: RSA encrypt key K using PB

$\{ K \}_{PB}$

Secret Email

PA, SA
3-DES
RSA



Step 4: Sender sends receiver the RSA-encrypted key K and the 3-DES encrypted message m

$\{ m \}_K \quad \{ K \}_{PB}$

PB, SB
3-DES
RSA

Step 1: Generate 3-DES key K for bulk encryption

Step 2: 3-DES encrypt message m using key K

$\{ m \}_K$

Step 3: RSA encrypt key K using PB

$\{ K \}_{PB}$

Secret Email

PA, SA
3-DES
RSA

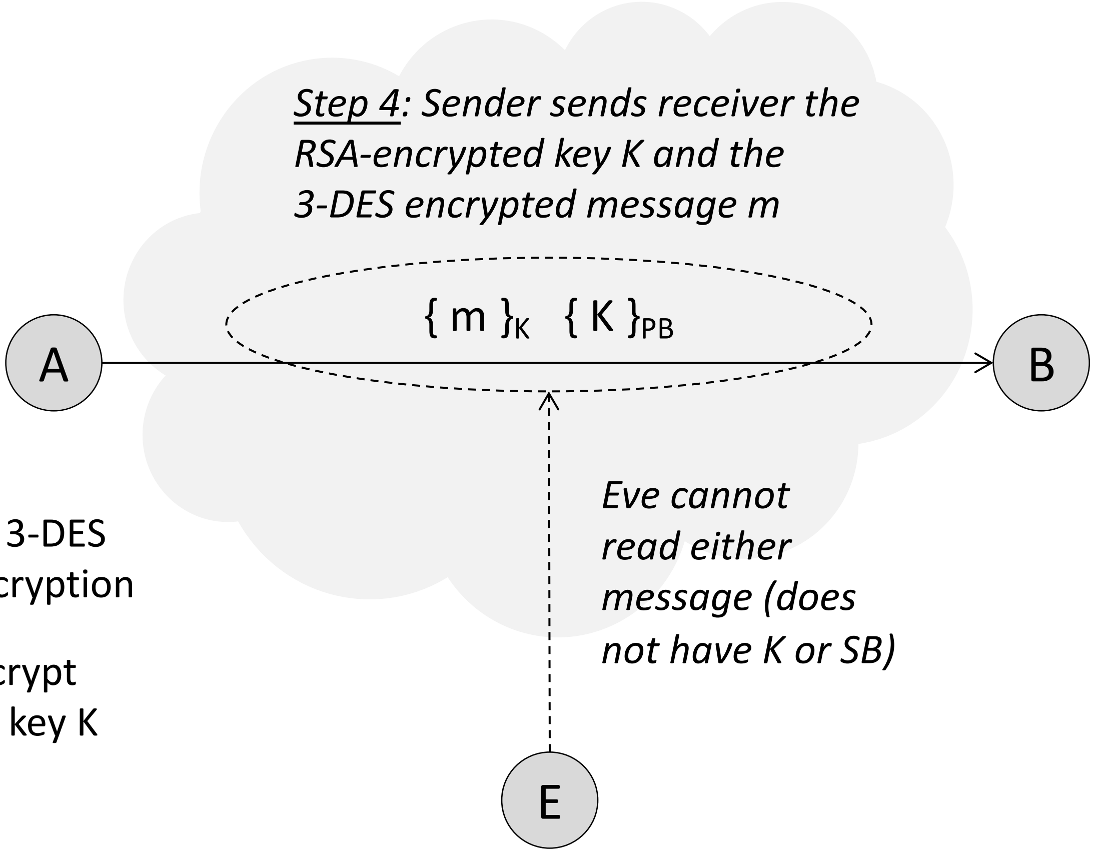
Step 1: Generate 3-DES key K for bulk encryption

Step 2: 3-DES encrypt message m using key K

$\{ m \}_K$

Step 3: RSA encrypt key K using PB

$\{ K \}_{PB}$



Step 4: Sender sends receiver the RSA-encrypted key K and the 3-DES encrypted message m

$\{ m \}_K \quad \{ K \}_{PB}$

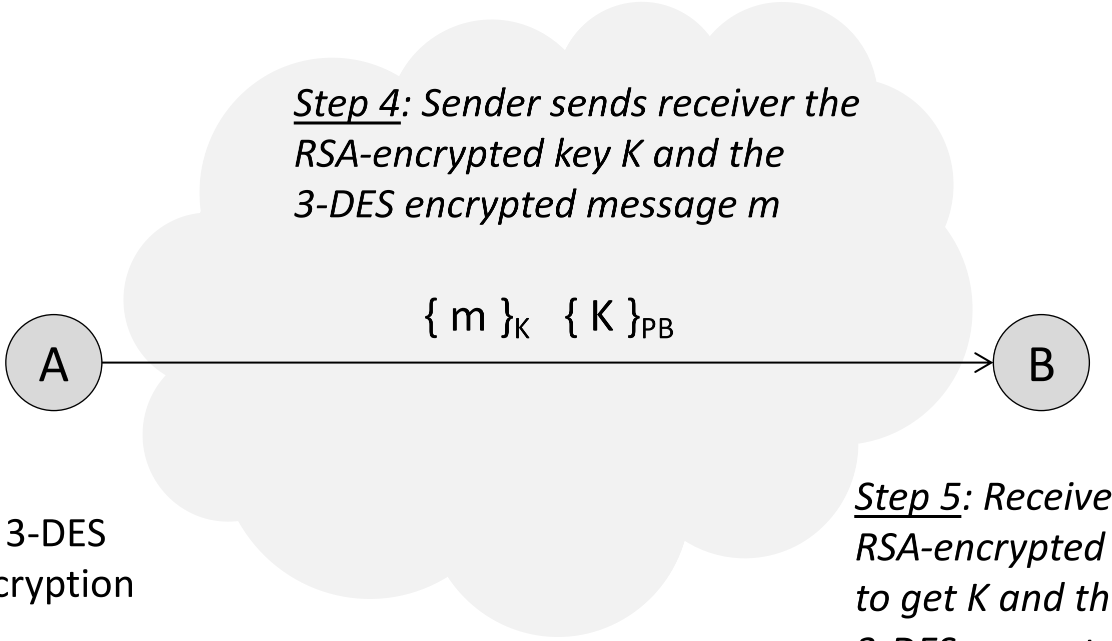
Eve cannot read either message (does not have K or SB)

PB, SB
3-DES
RSA

PA, PB

Secret Email

PA, SA
3-DES
RSA



<u>Step 4</u>: Sender sends receiver the RSA-encrypted key K and the 3-DES encrypted message m

$\{ m \}_K \quad \{ K \}_{PB}$

PB, SB
3-DES
RSA

<u>Step 1</u>: Generate 3-DES key K for bulk encryption

<u>Step 2</u>: 3-DES encrypt message m using key K

$\{ m \}_K$

<u>Step 3</u>: RSA encrypt key K using PB

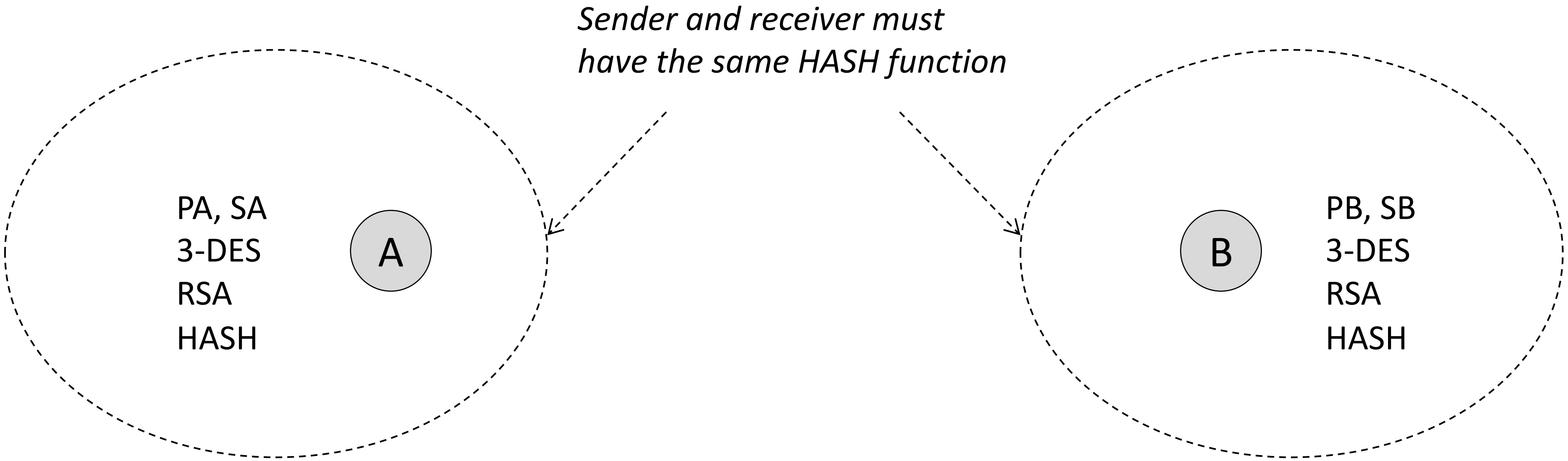
$\{ K \}_{PB}$

<u>Step 5</u>: Receiver decrypts the RSA-encrypted key with SB to get K and then decrypts the 3-DES encrypted message to get m

$\{ \{ K \}_{PB} \}_{SB} = K$

$\{ \{ m \}_K \}_K = m$

Digitally Signed Email



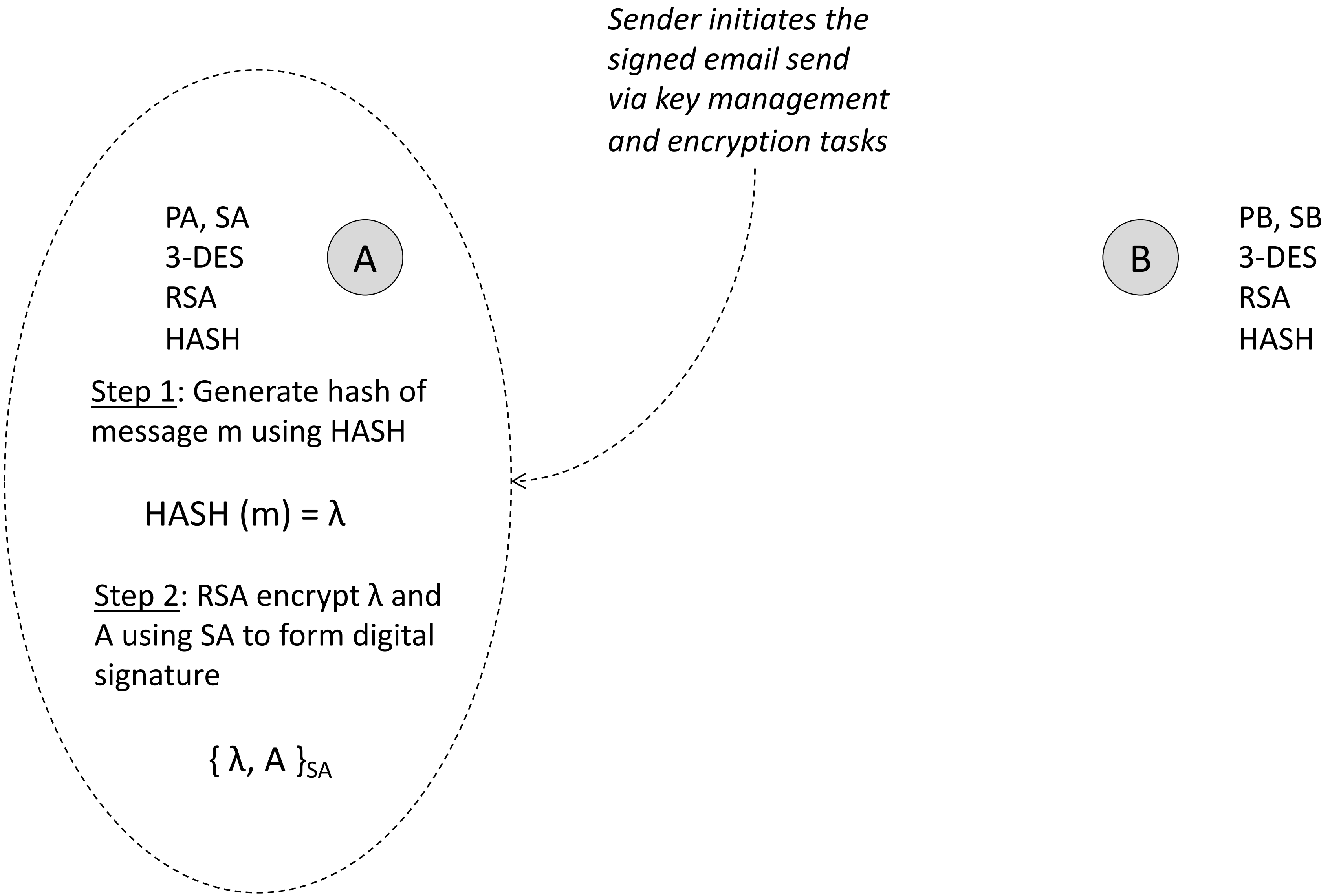
Sender and receiver must have the same HASH function

PA, SA
3-DES
RSA
HASH

PB, SB
3-DES
RSA
HASH

Week 8

Digitally Signed Email



Sender initiates the signed email send via key management and encryption tasks

PA, SA
3-DES
RSA
HASH

Step 1: Generate hash of message m using HASH

$$\text{HASH (m)} = \lambda$$

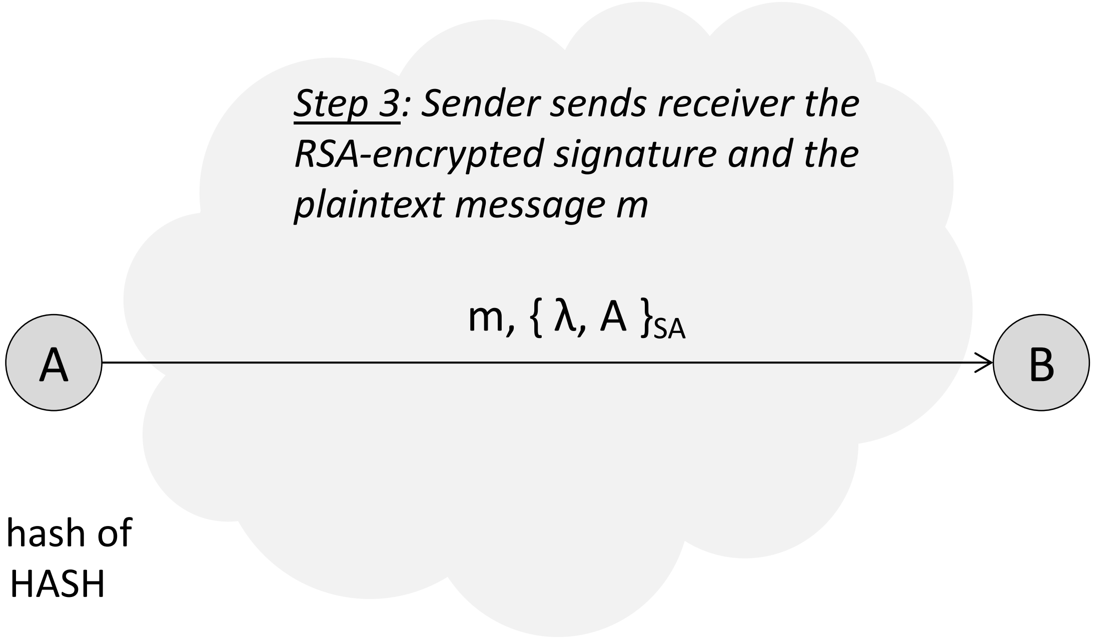
Step 2: RSA encrypt λ and A using SA to form digital signature

$$\{\ \lambda,\ A\ \}_{SA}$$

PB, SB
3-DES
RSA
HASH

Digitally Signed Email

PA, SA
3-DES
RSA
HASH



Step 3: Sender sends receiver the RSA-encrypted signature and the plaintext message m

$m, \{ \lambda, A \}_{SA}$

PB, SB
3-DES
RSA
HASH

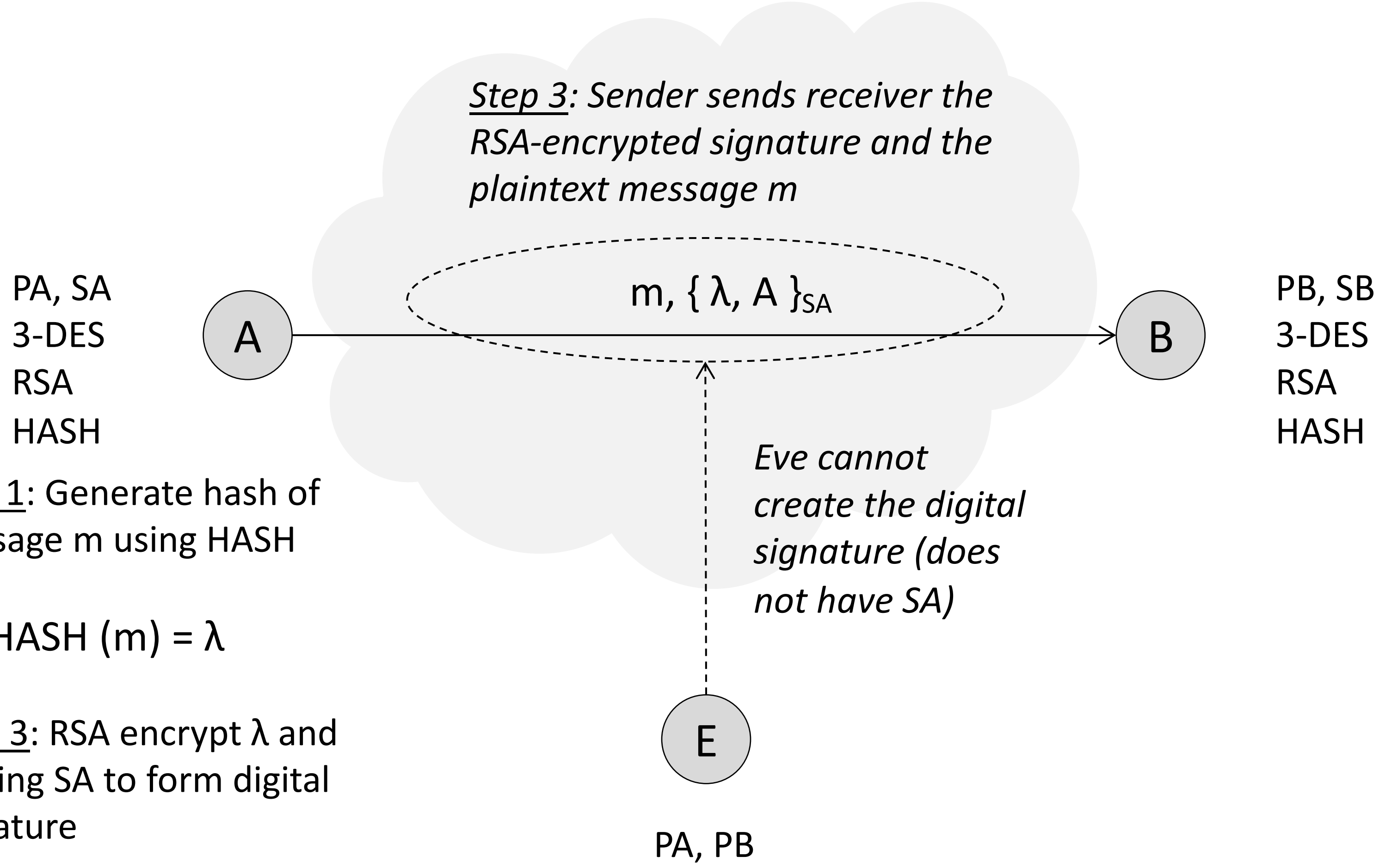
Step 1: Generate hash of message m using HASH

$$\text{HASH}(m) = \lambda$$

Step 3: RSA encrypt λ and A using SA to form digital signature

$$\{ \lambda, A \}_{SA}$$

Digitally Signed Email



Step 3: Sender sends receiver the RSA-encrypted signature and the plaintext message m

$m, \{ \lambda, A \}_{SA}$

Eve cannot create the digital signature (does not have SA)

Step 1: Generate hash of message m using HASH

$$\text{HASH}(m) = \lambda$$

Step 3: RSA encrypt λ and A using SA to form digital signature

$$\{ \lambda, A \}_{SA}$$

Digitally Signed Email

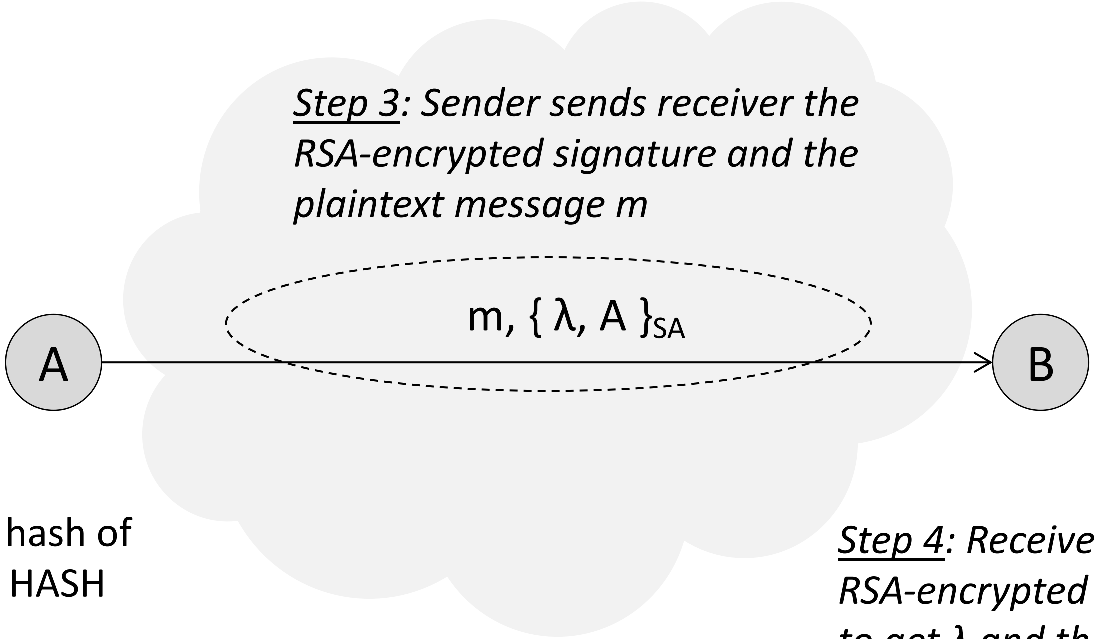
PA, SA
3-DES
RSA
HASH

Step 1: Generate hash of message m using HASH

$$\text{HASH } (m) = \lambda$$

Step 3: RSA encrypt λ and A using SA to form digital signature

$$\{ \lambda, A \}_{SA}$$



Step 3: Sender sends receiver the RSA-encrypted signature and the plaintext message m

$$m, \{ \lambda, A \}_{SA}$$

PB, SB
3-DES
RSA
HASH

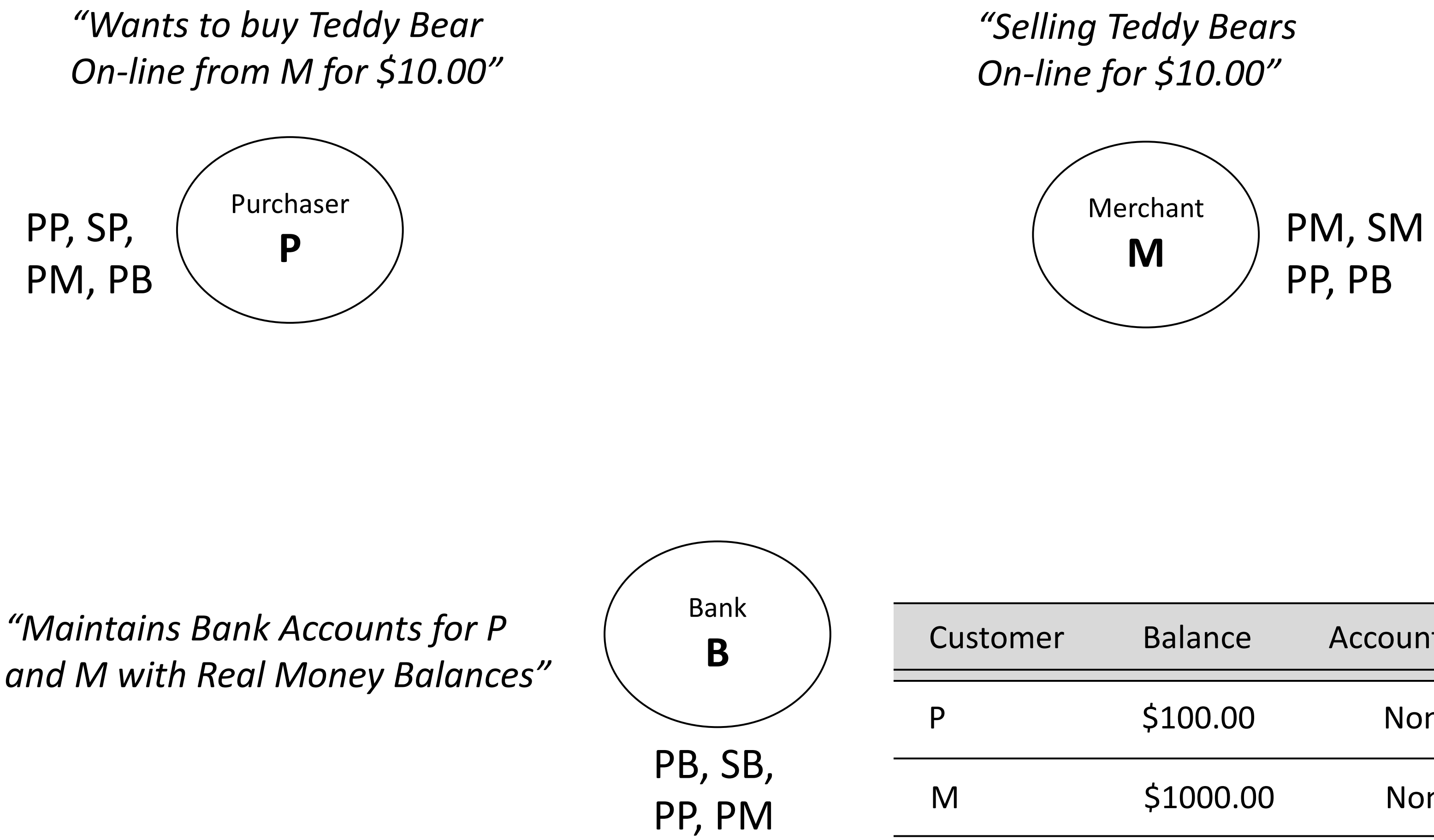
Step 4: Receiver decrypts the RSA-encrypted signature with SA to get λ and then locally computes HASH (m) to check validity

$$\{ \{ \lambda, A \}_{SA} \}_{PA} = \lambda, A$$

$$\text{HASH } (m) = \lambda$$

How Might Virtual Banking be Secured?

Banking Security



"Wants to buy Teddy Bear On-line from M for $10.00"

PP, SP, PM, PB

"Selling Teddy Bears On-line for $10.00"

PM, SM PP, PB

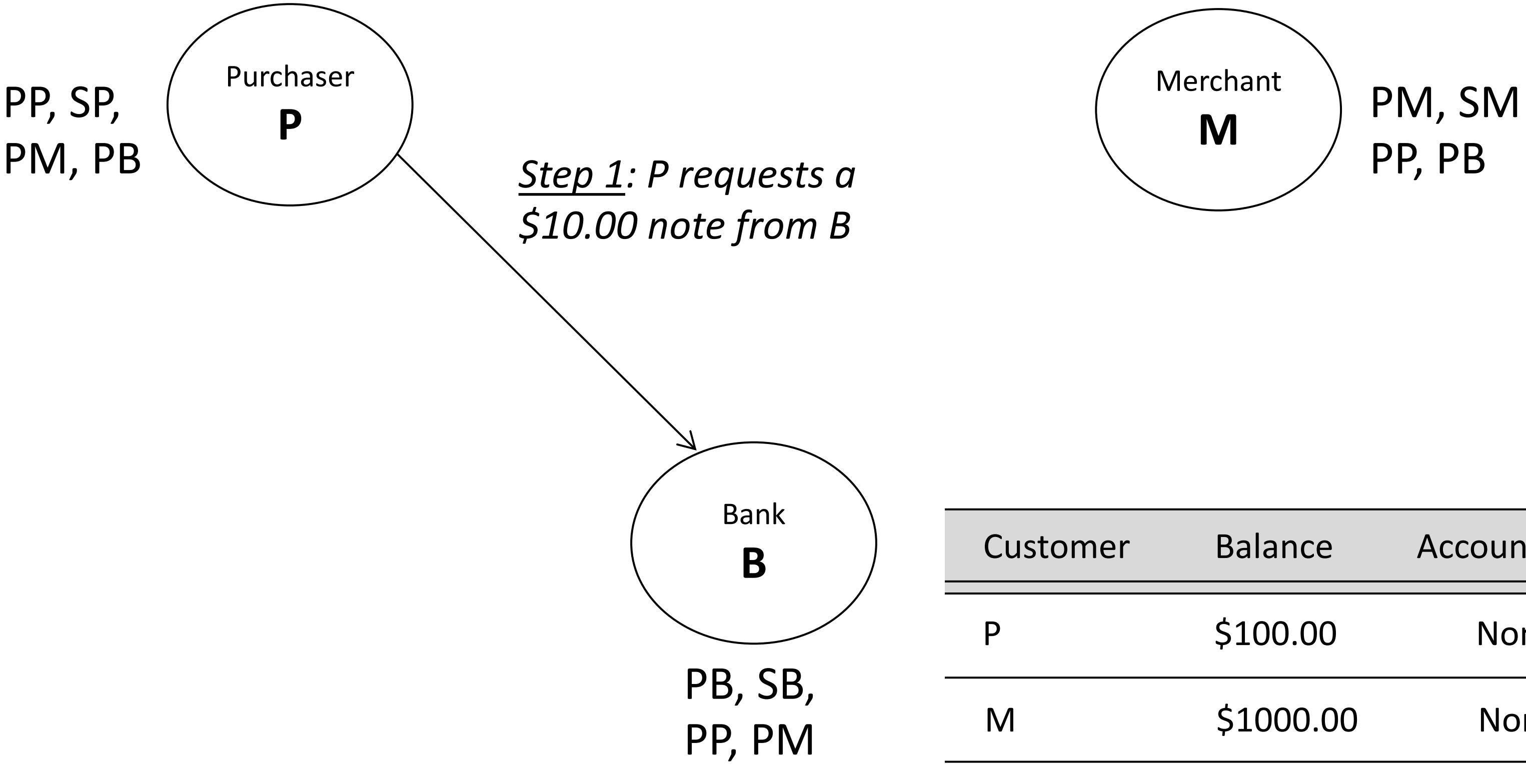
"Maintains Bank Accounts for P and M with Real Money Balances"

PB, SB, PP, PM

Customer	Balance	Account Notes
P	$100.00	None
M	$1000.00	None

Week 8

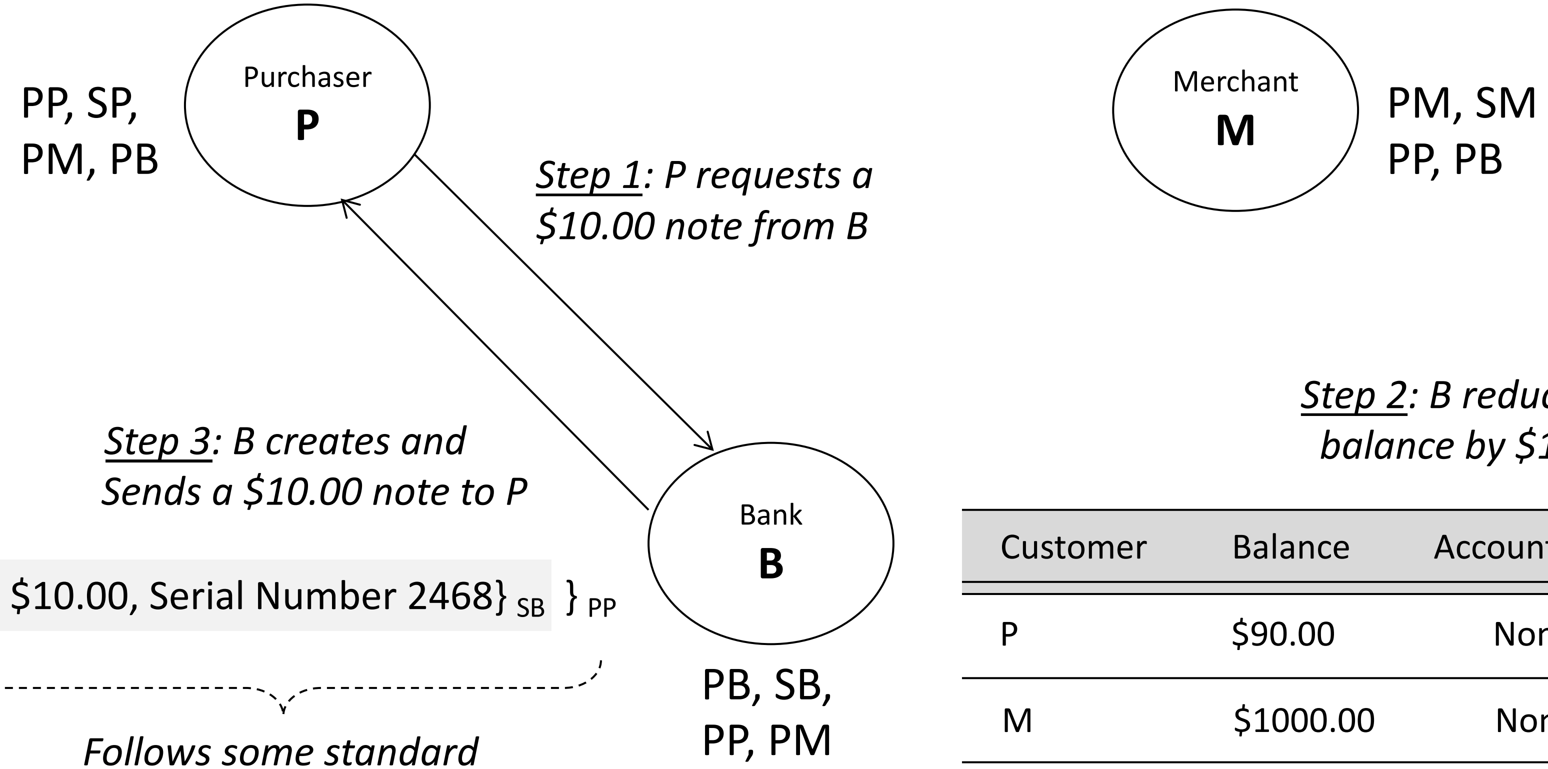
Banking Security



<u>Step 1</u>: P requests a $10.00 note from B

Customer	Balance	Account Notes
P	$100.00	None
M	$1000.00	None

Banking Security



<u>Step 1</u>: P requests a $10.00 note from B

<u>Step 2</u>: B reduces P's balance by $10.00

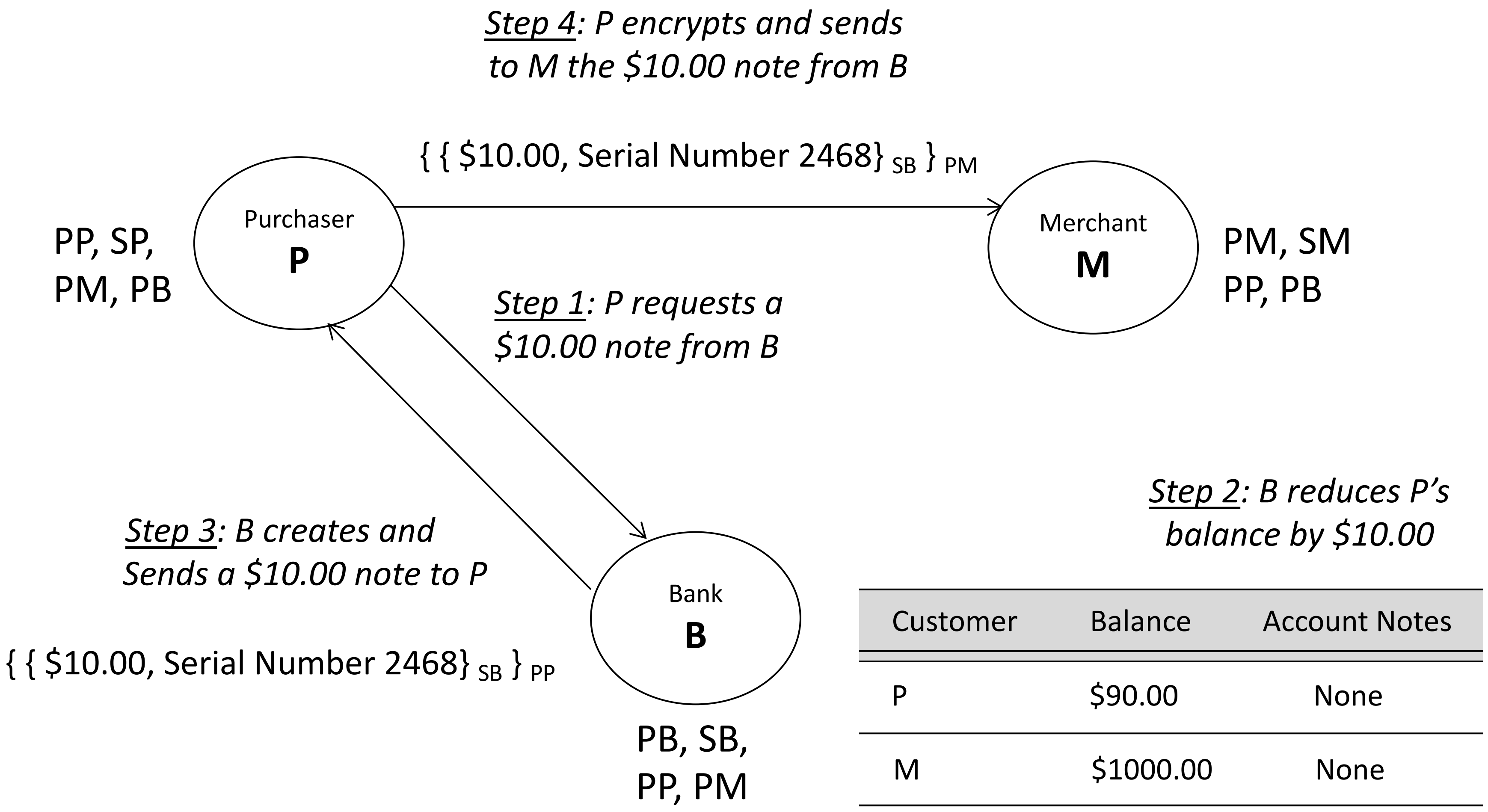
Customer	Balance	Account Notes
P	$90.00	None
M	$1000.00	None

<u>Step 3</u>: B creates and Sends a $10.00 note to P

$$\underbrace{\{ \{ \$10.00, \text{Serial Number } 2468\}_{SB} \}_{PP}}$$

Follows some standard bank note format with a random, unique serial number

Banking Security



Step 4: P encrypts and sends to M the $10.00 note from B

$\{ \{ \$10.00, \text{Serial Number } 2468\}_{SB} \}_{PM}$

Step 1: P requests a $10.00 note from B

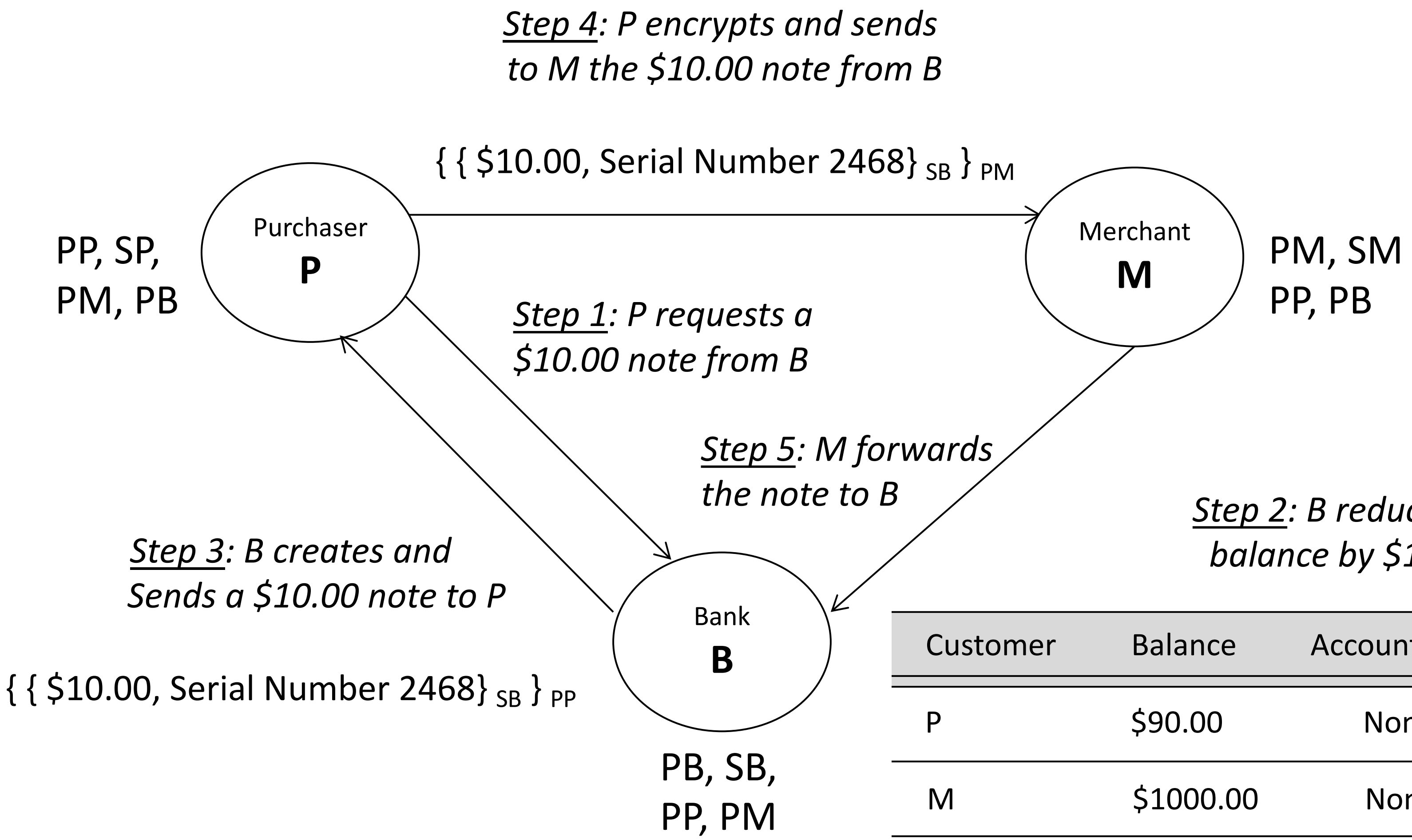
Step 2: B reduces P's balance by $10.00

Step 3: B creates and Sends a $10.00 note to P

$\{ \{ \$10.00, \text{Serial Number } 2468\}_{SB} \}_{PP}$

Customer	Balance	Account Notes
P	$90.00	None
M	$1000.00	None

Banking Security



<u>Step 1</u>: P requests a $10.00 note from B

<u>Step 2</u>: B reduces P’s balance by $10.00

<u>Step 3</u>: B creates and Sends a $10.00 note to P

$\{\{\$10.00, \text{Serial Number } 2468\}_{SB}\}_{PP}$

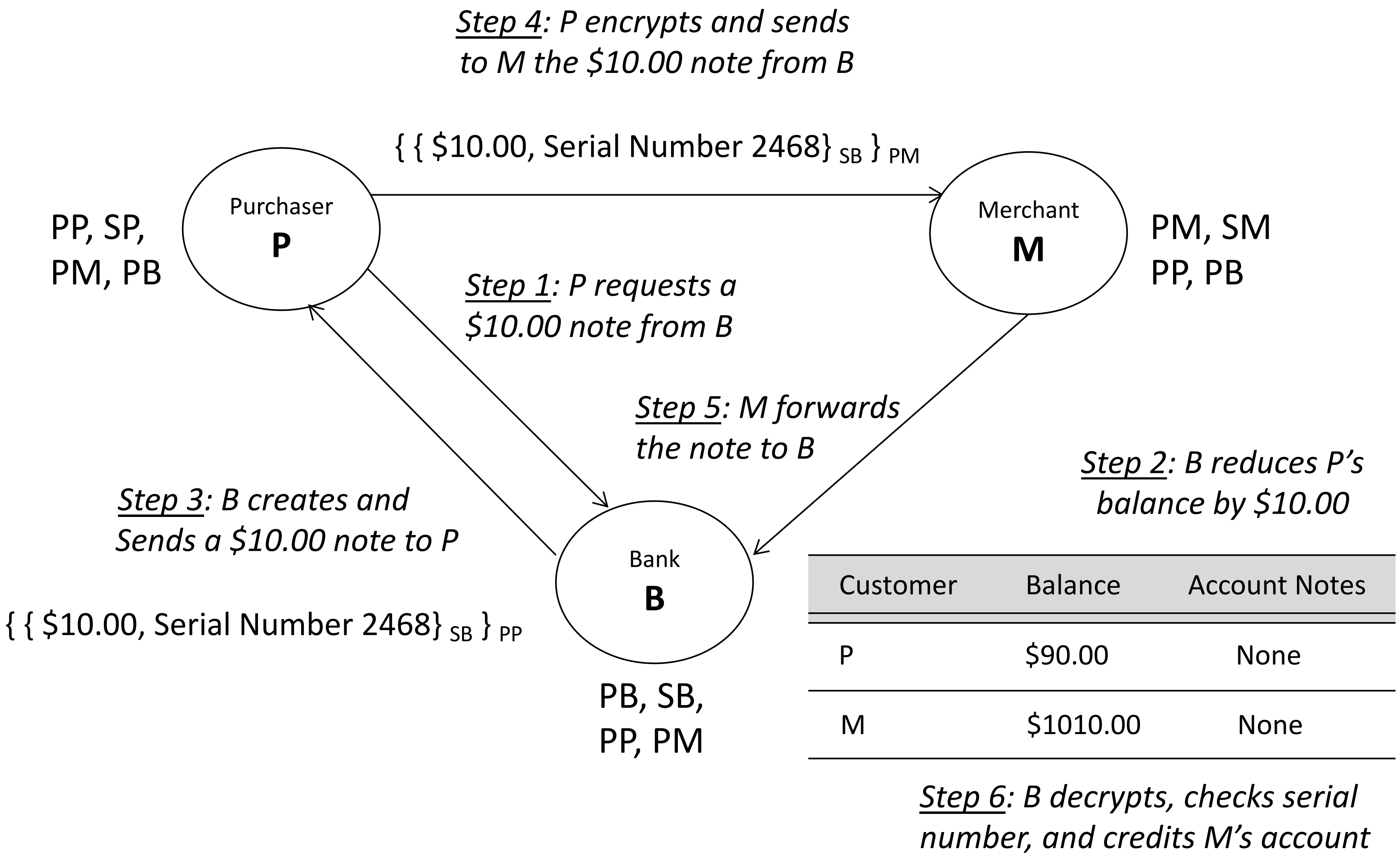
<u>Step 4</u>: P encrypts and sends to M the $10.00 note from B

$\{\{\$10.00, \text{Serial Number } 2468\}_{SB}\}_{PM}$

<u>Step 5</u>: M forwards the note to B

Customer	Balance	Account Notes
P	$90.00	None
M	$1000.00	None

Banking Security



<u>Step 4</u>: P encrypts and sends to M the $10.00 note from B

$\{ \{ \$10.00, \text{Serial Number } 2468\}_{SB} \}_{PM}$

<u>Step 1</u>: P requests a $10.00 note from B

<u>Step 5</u>: M forwards the note to B

<u>Step 2</u>: B reduces P's balance by $10.00

<u>Step 3</u>: B creates and Sends a $10.00 note to P

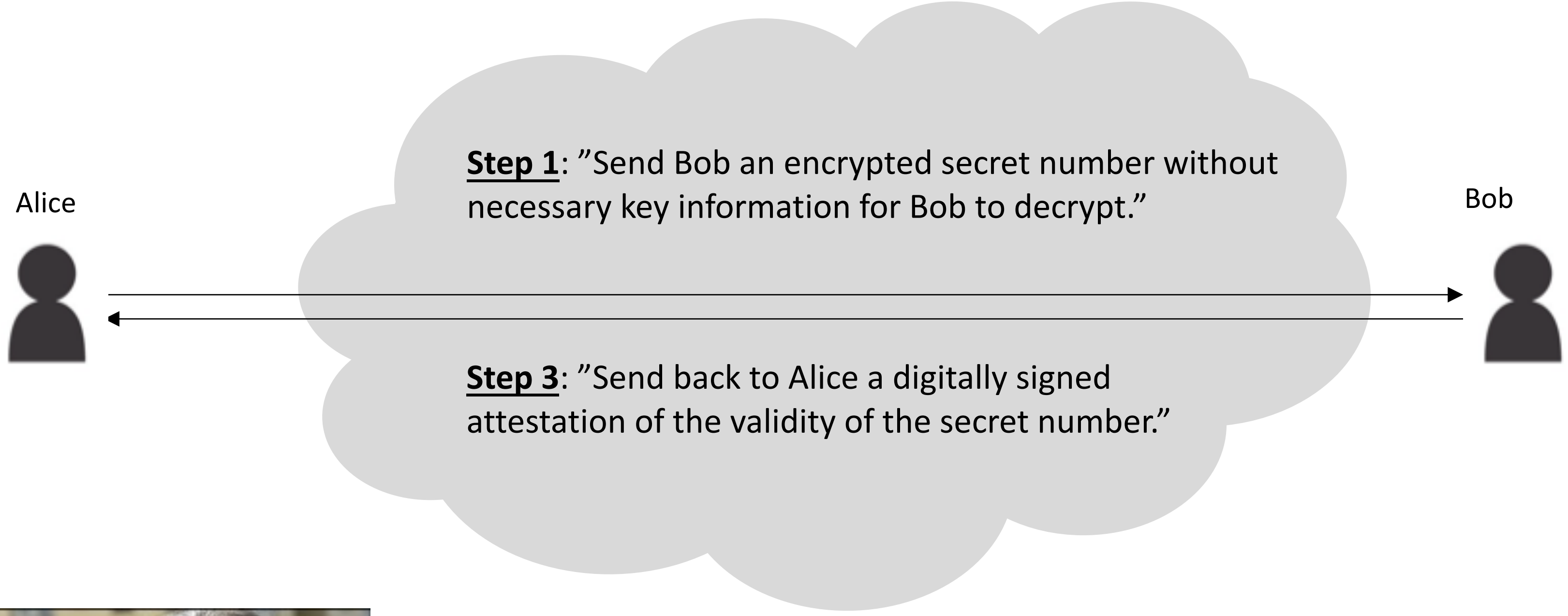
$\{ \{ \$10.00, \text{Serial Number } 2468\}_{SB} \}_{PP}$

Customer	Balance	Account Notes
P	$90.00	None
M	$1010.00	None

<u>Step 6</u>: B decrypts, checks serial number, and credits M's account

What is a Blinding Protocol?

Chaum’s Blinding Protocol: Goal



Step 1: ”Send Bob an encrypted secret number without necessary key information for Bob to decrypt.”

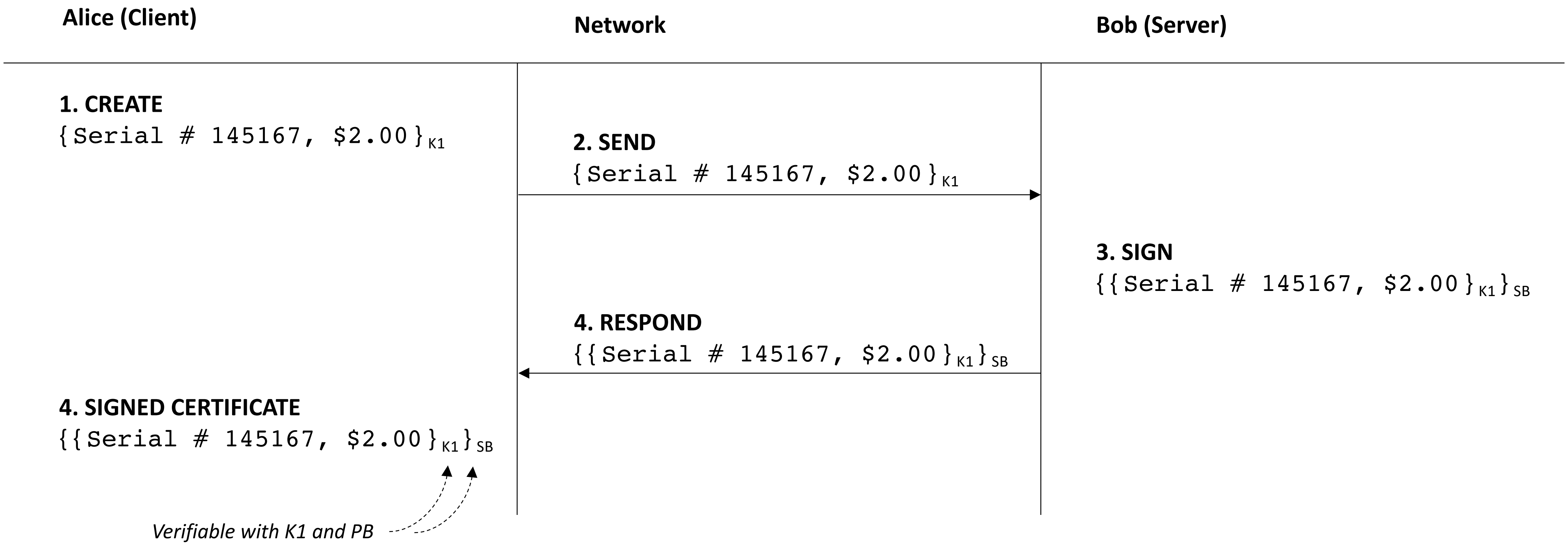
Step 2: ”Attest to the validity of the encrypted secret number without decrypting or reading it (i.e., fully blind attestation)”

Step 3: ”Send back to Alice a digitally signed attestation of the validity of the secret number.”

David Chaum
University of California at Berkeley
Founder DigiCash (defunct)

Week 8

Chaum’s Blinding Protocol: Goal



Alice (Client)

Network

Bob (Server)

1. CREATE

$\{\texttt{Serial \# 145167, \$2.00}\}_{K1}$

2. SEND

$\{\texttt{Serial \# 145167, \$2.00}\}_{K1}$

3. SIGN

$\{\{\texttt{Serial \# 145167, \$2.00}\}_{K1}\}_{SB}$

4. RESPOND

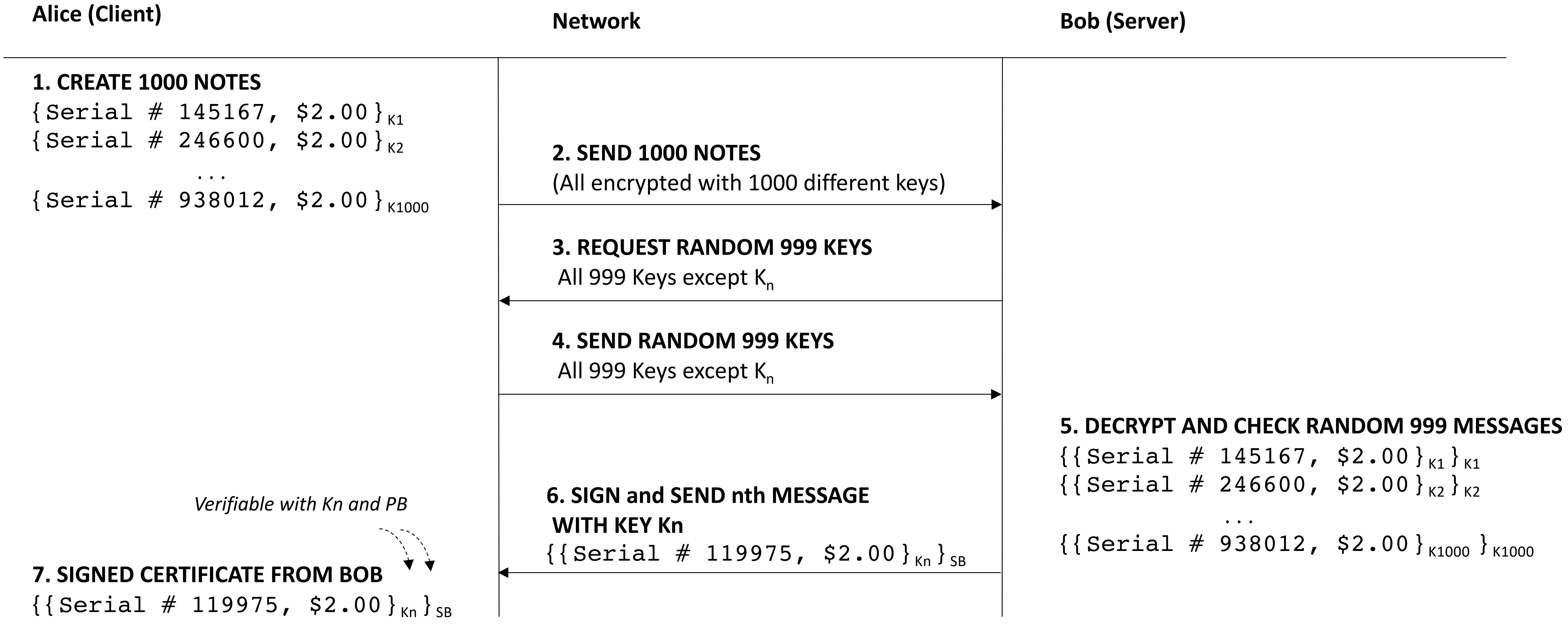
$\{\{\texttt{Serial \# 145167, \$2.00}\}_{K1}\}_{SB}$

4. SIGNED CERTIFICATE

$\{\{\texttt{Serial \# 145167, \$2.00}\}_{K1}\}_{SB}$

Verifiable with K1 and PB

Chaum's Blinding Protocol: Implementation



Alice (Client)

Network

Bob (Server)

1. CREATE 1000 NOTES

$\{\texttt{Serial \# 145167, \$2.00}\}_{K1}$

$\{\texttt{Serial \# 246600, \$2.00}\}_{K2}$

...

$\{\texttt{Serial \# 938012, \$2.00}\}_{K1000}$

2. SEND 1000 NOTES
(All encrypted with 1000 different keys)

3. REQUEST RANDOM 999 KEYS
All 999 Keys except K_n

4. SEND RANDOM 999 KEYS
All 999 Keys except K_n

5. DECRYPT AND CHECK RANDOM 999 MESSAGES

$\{\{\texttt{Serial \# 145167, \$2.00}\}_{K1}\}_{K1}$

$\{\{\texttt{Serial \# 246600, \$2.00}\}_{K2}\}_{K2}$

...

$\{\{\texttt{Serial \# 938012, \$2.00}\}_{K1000}\}_{K1000}$

6. SIGN and SEND nth MESSAGE
WITH KEY Kn

$\{\{\texttt{Serial \# 119975, \$2.00}\}_{Kn}\}_{SB}$

Verifiable with Kn and PB

7. SIGNED CERTIFICATE FROM BOB

$\{\{\texttt{Serial \# 119975, \$2.00}\}_{Kn}\}_{SB}$

What is Zero Trust?

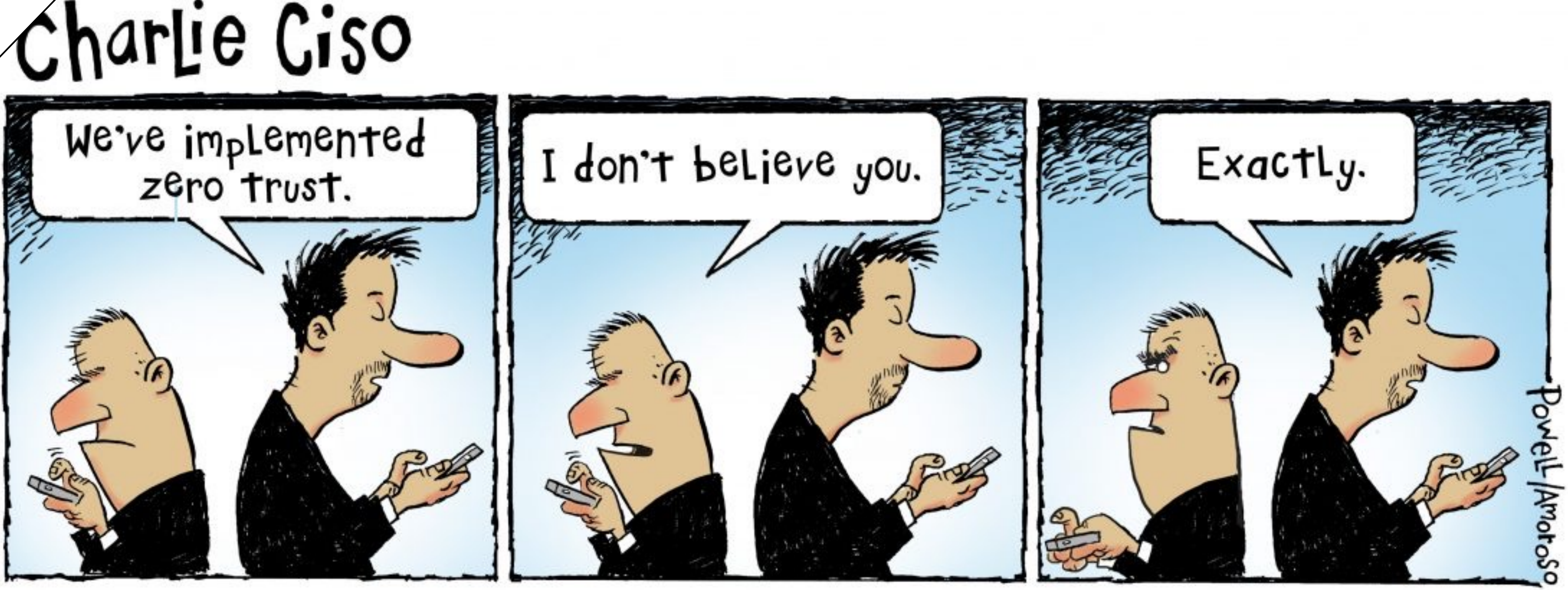
Week 8

Charlie Ciso
We've implemented zero trust.

Week 8

Charlie Ciso
We've implemented zero trust.
I don't believe you.

Week 8


Charlie Ciso
We've implemented zero trust.
I don't believe you.
Exactly.
Powell/Amoroso

What is Zero Trust?

- **Conceptual cyber security model for protection of apps and data**
    - Introduced by Forrester in 2010 (possibly earlier by Jericho Forum)

What is Zero Trust?

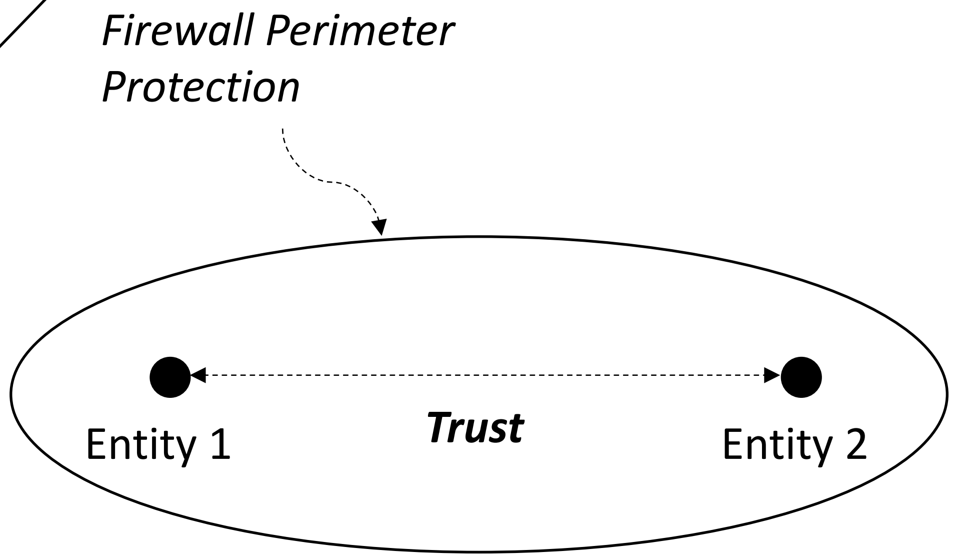
- **Conceptual cyber security model for protection of apps and data**
  - Introduced by Forrester in 2010 (possibly earlier by Jericho Forum)
- **Identity verification versus perimeter protection**
  - Endpoint workloads are authenticated and authorized based on identity

What is Zero Trust?

- **Conceptual cyber security model for protection of apps and data**
  - Introduced by Forrester in 2010 (possibly earlier by Jericho Forum)
- **Identity verification versus perimeter protection**
  - Endpoint workloads are authenticated and authorized based on identity
- **Trust no longer established by enterprise perimeter**
  - Firewall perimeters no longer a primary control in Zero Trust

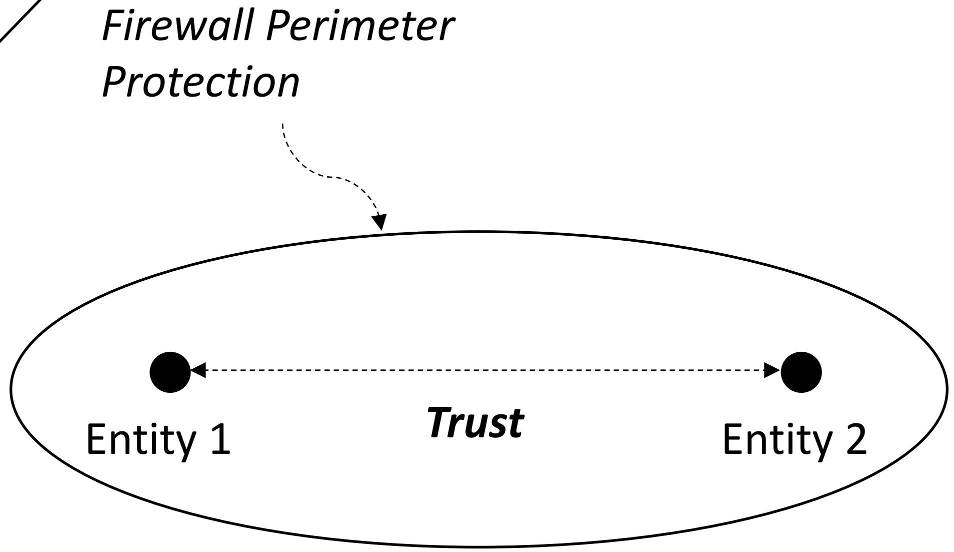
Week 8

Zero Trust: Entities Must Self-Protect



1. Entity 1 and 2 can share freely (bidirectional)
2. No mutual authentication (no 1FA, 2FA, etc.)
3. Shared boundary protection (perimeter)
4. Malware can traverse laterally from 1 to 2

Firewall Perimeter Protection (Opposite of Zero Trust)



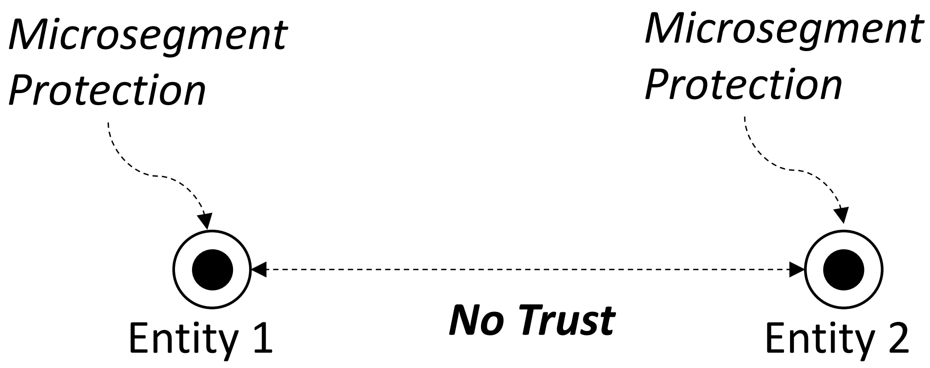
Firewall Perimeter Protection

Entity 1

Trust

Entity 2

1. Entity 1 and 2 can share freely (bidirectional)
2. No mutual authentication (no 1FA, 2FA, etc.)
3. Shared boundary protection (perimeter)
4. Malware can traverse from 1 to 2 freely



Microsegment Protection

Microsegment Protection

Entity 1

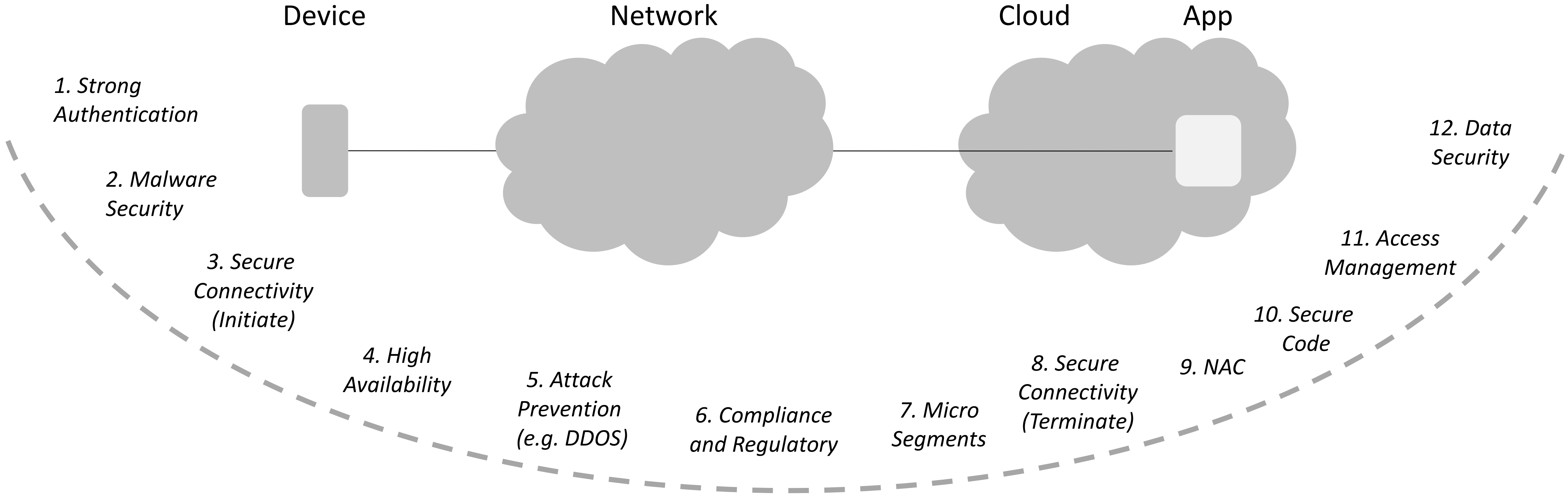
No Trust

Entity 2

1. Entity 1 and 2 will only share if necessary
2. Mutual authentication (1FA, 2FA, etc.)
3. Local boundary protections (no perimeter)
4. Malware cannot traverse from 1 to 2 freely

Comparison to Zero Trust with No Perimeter

Week 8



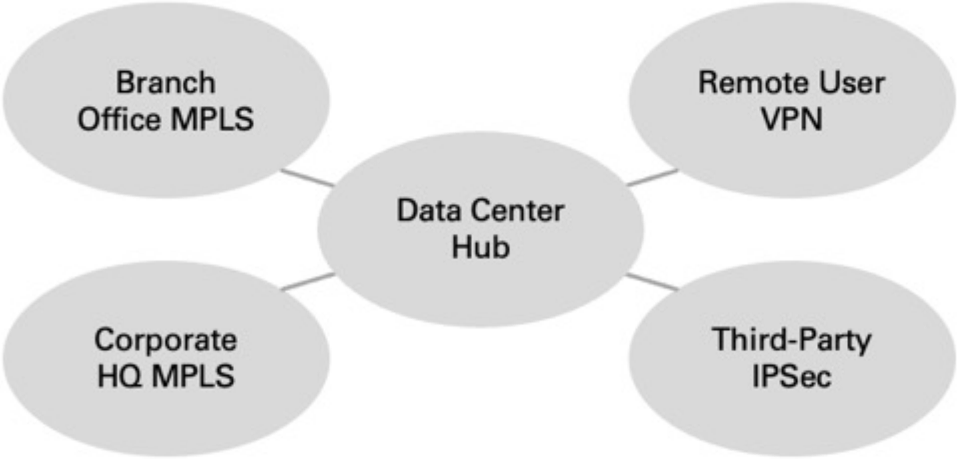
Components of Zero Trust Network Access (ZTNA)

What is Secure Business Networking 2.0?
(Hint: SASE)

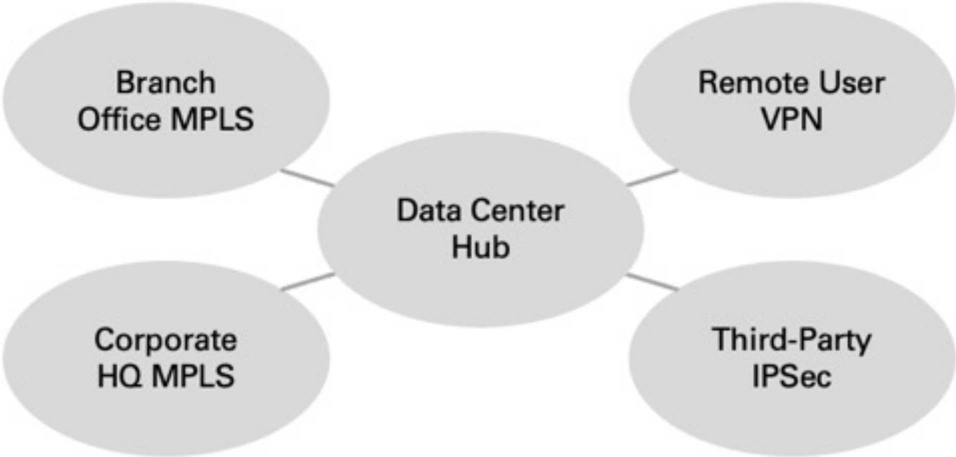
Week 8

"I just didn't have the stomach to tell him that SASE is not self-addressed stamped envelope."

Week 8



➡ SECURE BUSINESS NETWORKING 1.0 (MPLS, VPN, IPSEC)



Branch Office MPLS

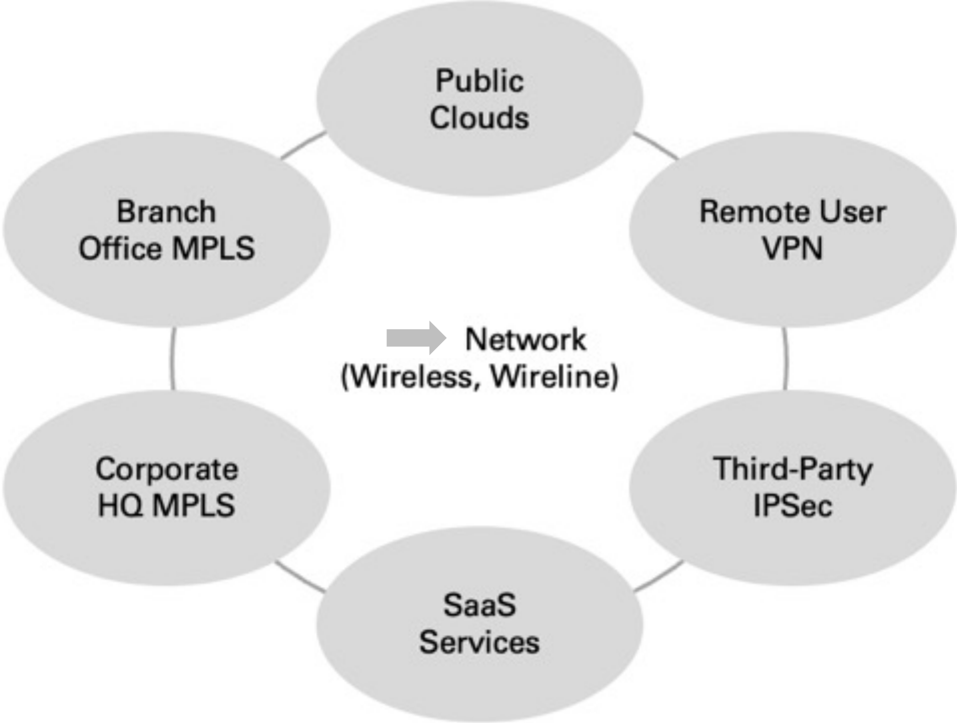
Remote User VPN

Data Center Hub

Corporate HQ MPLS

Third-Party IPSec

SECURE BUSINESS NETWORKING 1.0 (MPLS, VPN, IPSEC)



Public Clouds

Branch Office MPLS

Remote User VPN

Network (Wireless, Wireline)

Corporate HQ MPLS

Third-Party IPSec

SaaS Services

SECURE BUSINESS NETWORKING 2.0 (5G, FIBER, CLOUD, SAAS, SECURITY)