Today

- Security (Lesson 11)
  - ✓ * Terminologies
  - ⇒ * Security implementation in AFS

Wednesday

- Wrap up

Friday

- Townhall

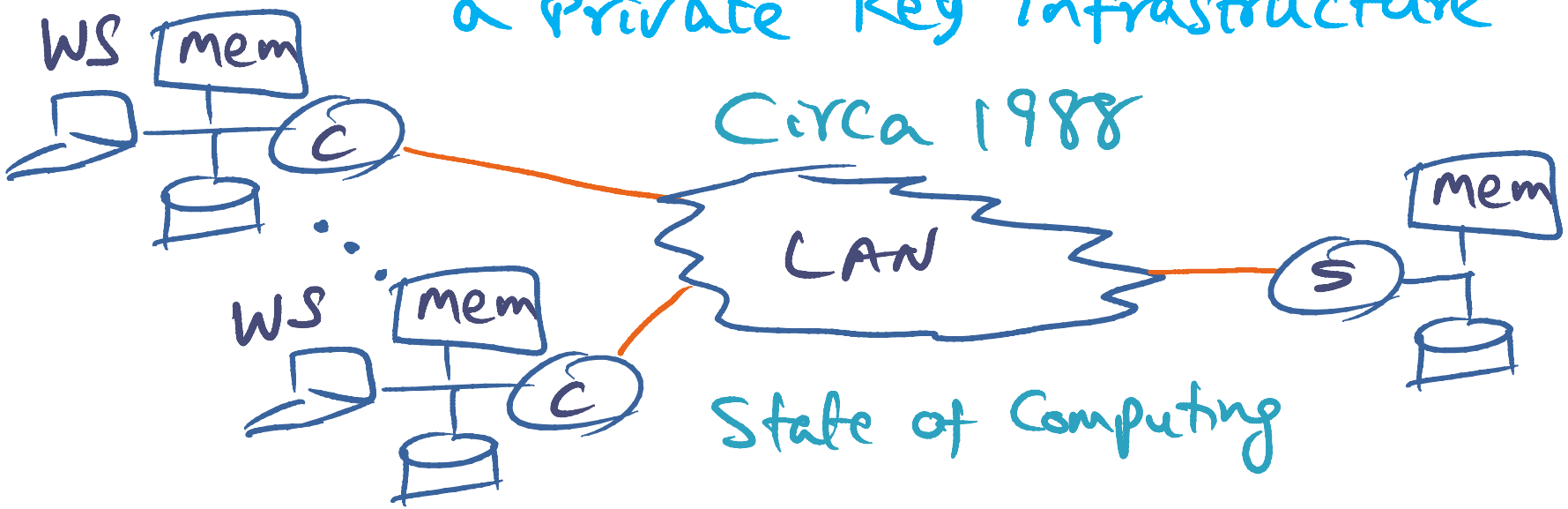
Final Exam

- Wed @ 8 AM (12/9)

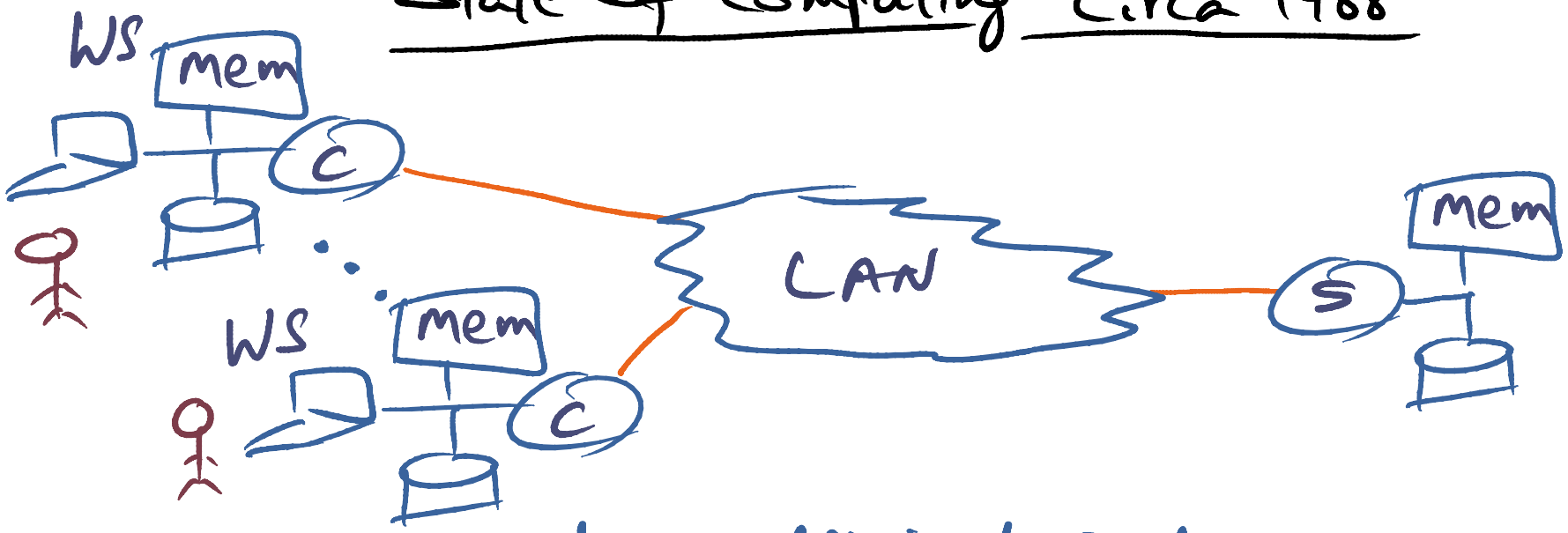
Lesson outline

✓ Terminologies

→ Security and authentication using a Private Key infrastructure

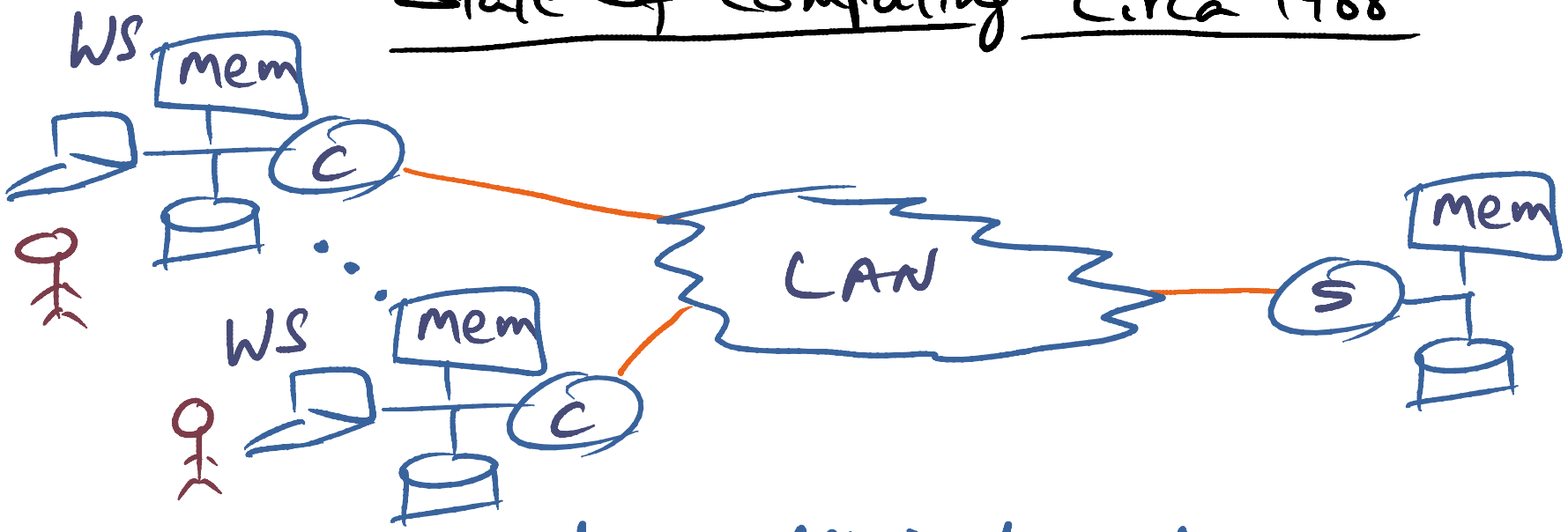


State of Computing Circa 1988



Local Disks served as efficient caches

State of Computing Circa 1988

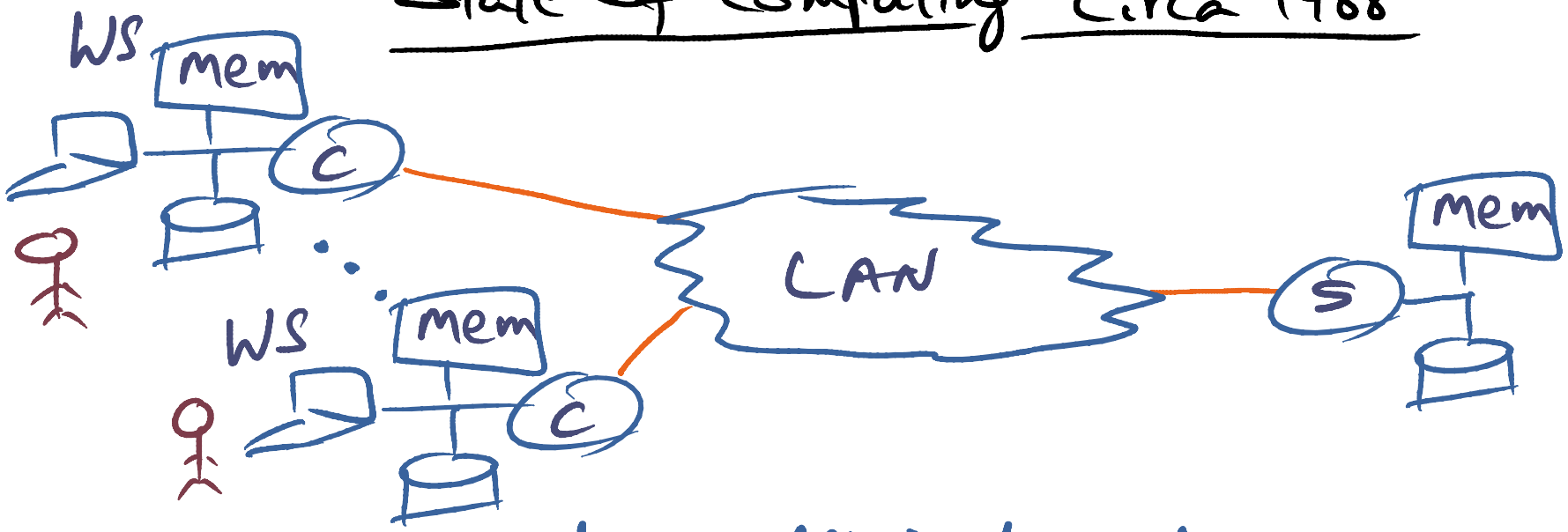


Local Disks served as efficient caches

Vision of AFS + Coda (both from CMU)

- Walk up to any WS + login

State of Computing Circa 1988

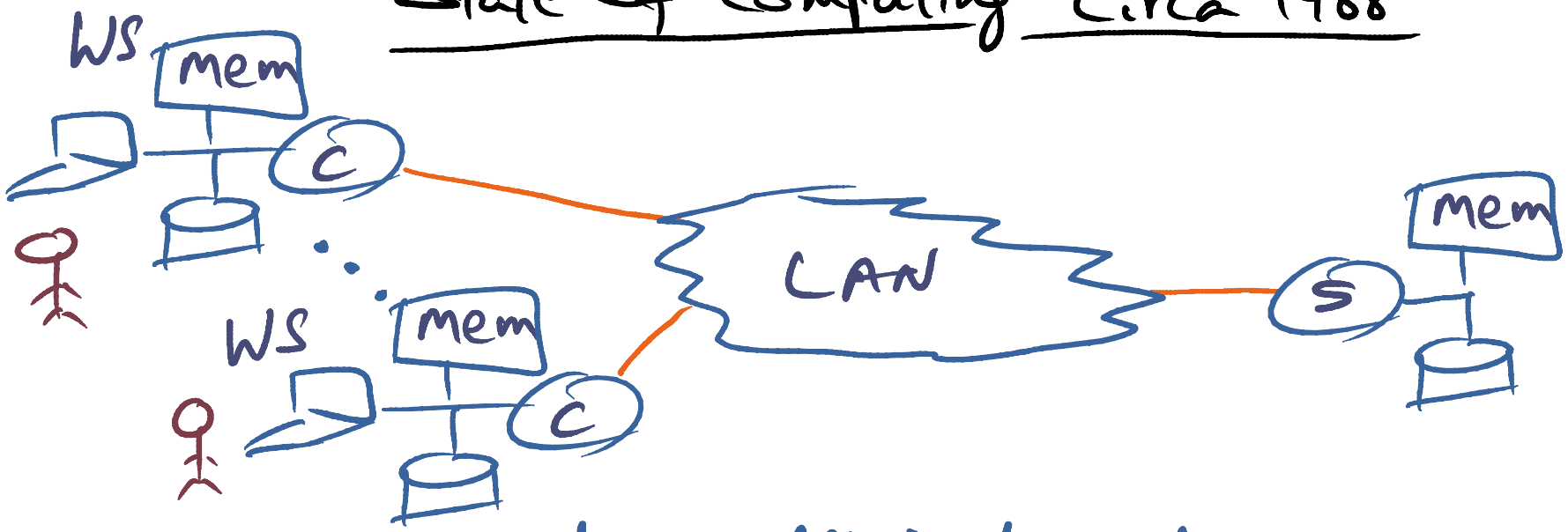


Local Disks served as efficient caches

Vision of AFS + Coda (both from CMU)

- Walk up to any WS + login
- your content magically appears

State of Computing Circa 1988



Local Disks served as efficient caches

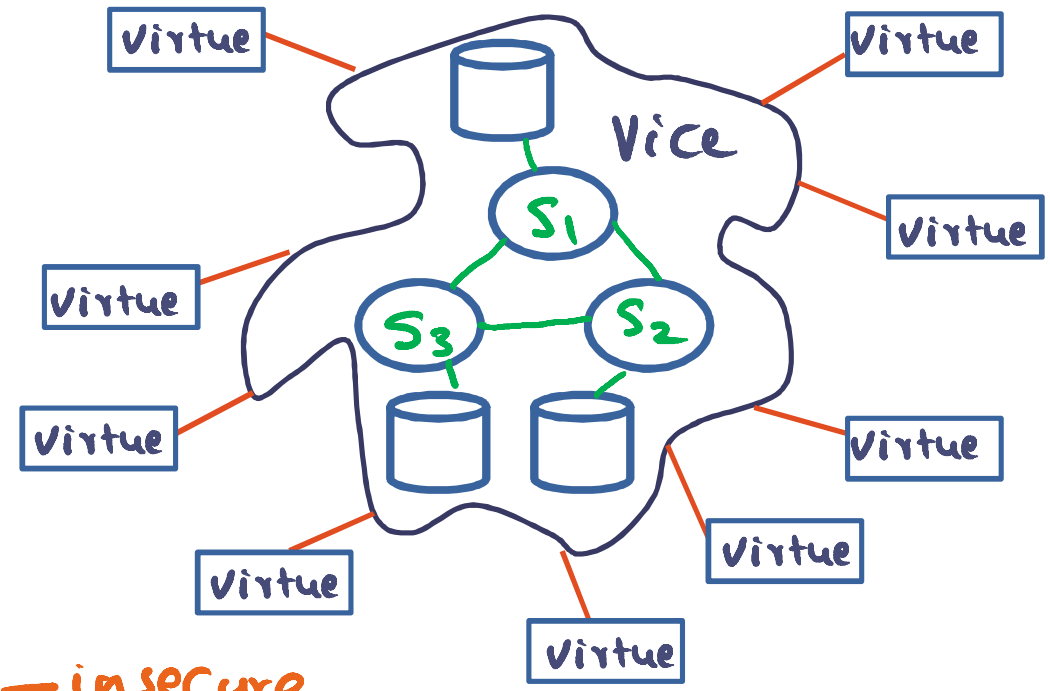
Vision of AFS + Coda (both from CMU)

- Walk up to any WS + login
- your content magically appears

Similar to the cloud + mobile devices today!!

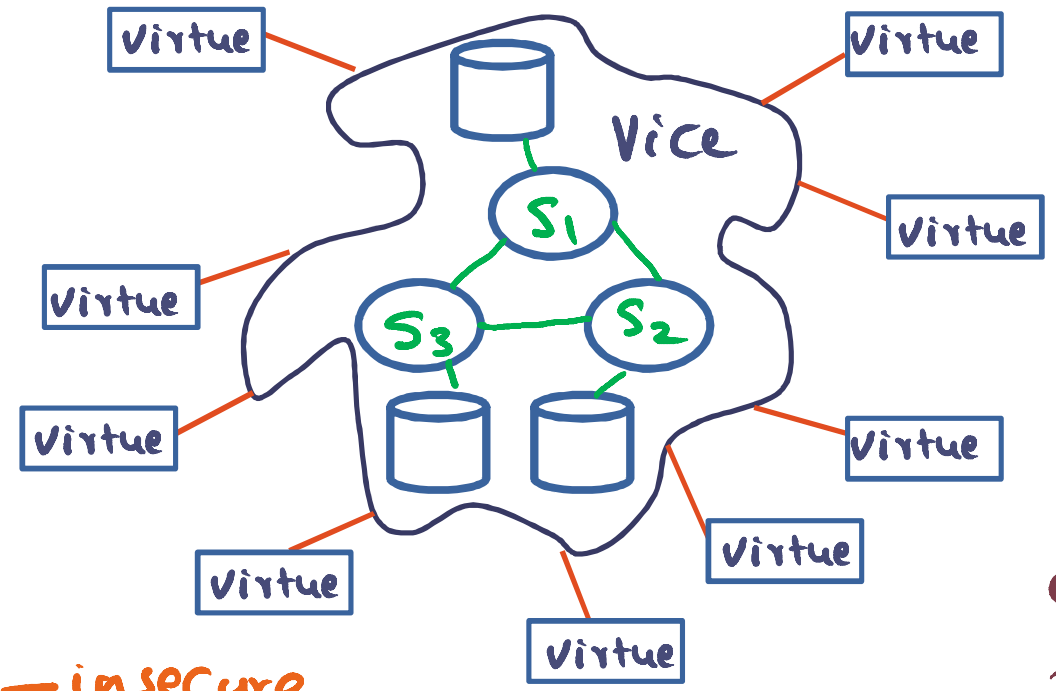
Andrew Architecture

client workstation
- unix



- insecure
- secure

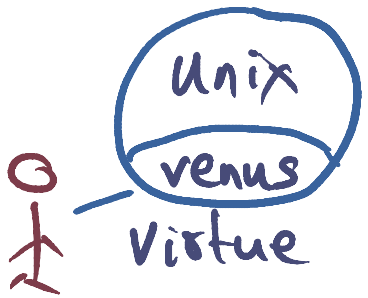
Andrew Architecture



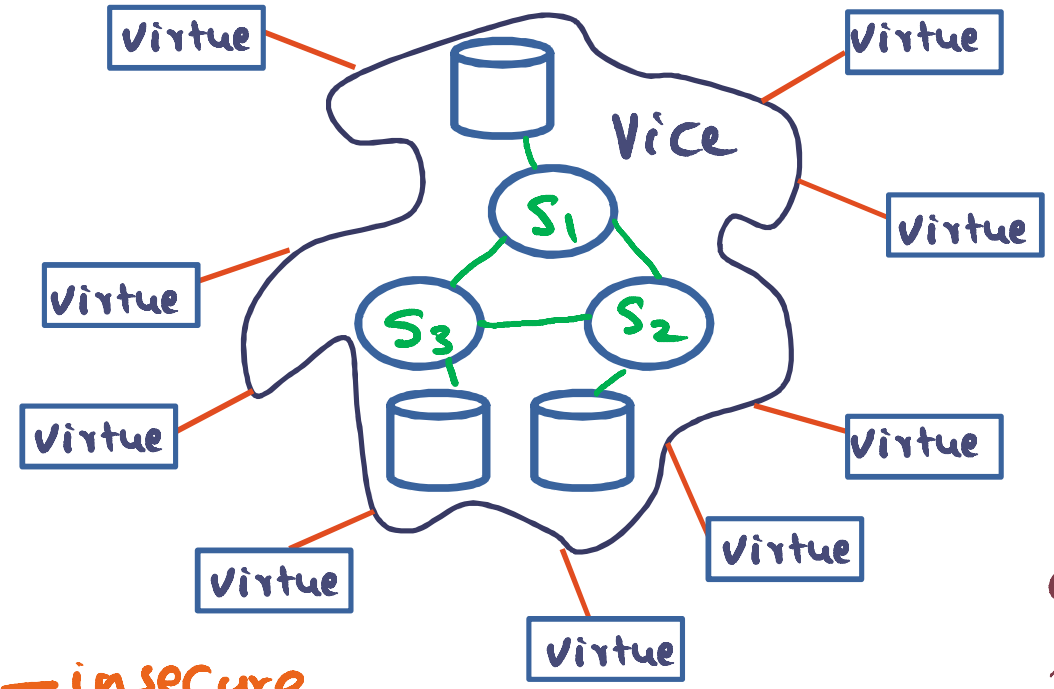
- insecure
- secure

Client Workstation
- unix

Venus
- Process on Virtue for user authentication and client caching



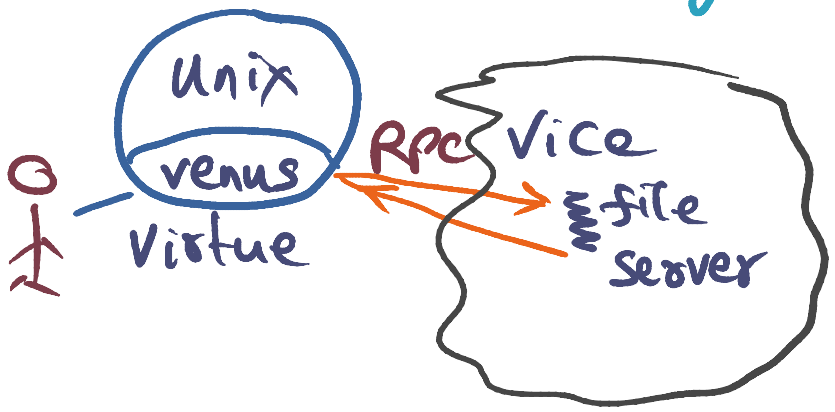
Andrew Architecture



- insecure
- secure

Client Workstation
- unix

Venus
- Process on Virtue for user authentication and client caching



Encryption Primer

Private Key System

- Symmetric Keys (e.g. passwords used in login)

S: Data → Enc(Data, Key) → Ciphertext

R: Data ← Dec(Ciphertext, Key) ←

Encryption Primer

Private Key System

- Symmetric Keys (e.g. passwords used in login)

S: Data → Enc(Data, Key) → Ciphertext

R: Data ← Dec(Ciphertext, Key) ←

Public Key system

- Asymmetric Keys (Pair of keys)
  - * Public Key Published ⇒ encrypt
  - * Private Key ⇒ decrypt

Encryption Primer

Private Key System

- Symmetric Keys (e.g. passwords used in login)

S: Data → Enc(Data, Key) → Ciphertext

R: Data ← Dec(Ciphertext, Key) ←

Public Key system

- Asymmetric Keys (Pair of Keys)
  - * Public Key Published ⇒ encrypt } one way functions
  - * Private Key ⇒ decrypt

Encryption Primer

Private Key System

- Symmetric Keys (e.g. passwords used in login)

S: Data → Enc(Data, Key) → Ciphertext

R: Data ← Dec(Ciphertext, Key)

Public Key system

- Asymmetric Keys (Pair of Keys)
  - * Public Key Published ⇒ encrypt } one way functions
  - * Private Key ⇒ decrypt

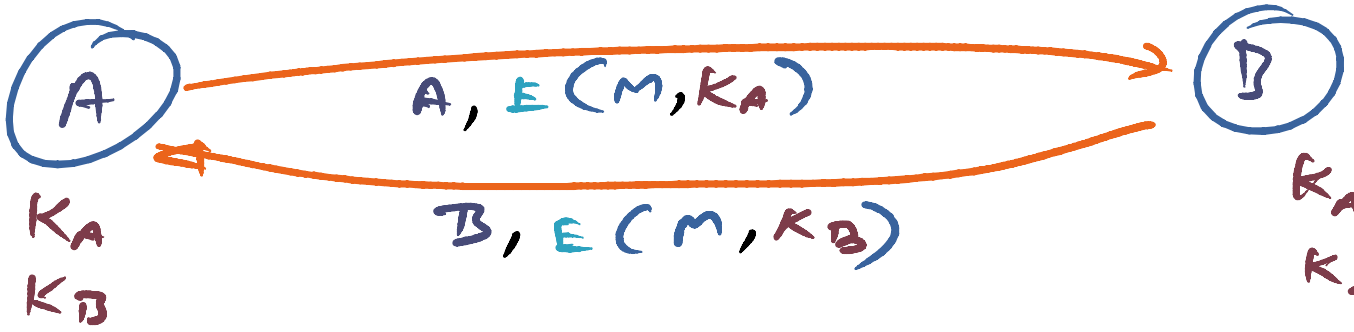
S: Data → Enc(Data, public key) → Ciphertext

R: Data ← Dec(Ciphertext, Private Key)

Private Key Encryption System in Action

With private key encryption

A + B have exchanged keys



Identity of sender

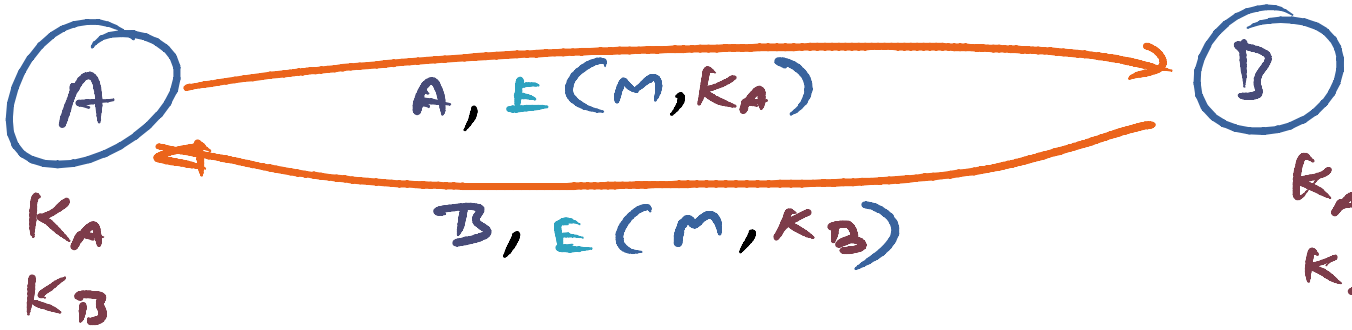
cleartext for receiver to know

what key to use for decryption

Private Key Encryption System in Action

With private key encryption

A + B have exchanged keys



Identity of sender

cleartext for receiver to know

what key to use for decryption

(K_A + K_B can be the same)

Challenges for Andrew System

Authenticate User

Authenticate Server

Prevent replay attacks

isolate users

Challenges for Andrew System

Authenticate User

Authenticate Server

Prevent replay attacks

isolate Users

Choice in Andrew: private key crypto system

Challenges for Andrew System

Authenticate User

Authenticate Server

Prevent replay attacks

isolate Users

Choice in Andrew: private key crypto system
⇒ identity of sender in cleartext

Challenges for Andrew System

Authenticate User

Authenticate Server

Prevent replay attacks

isolate users

Choice in Andrew: private key crypto system
⇒ identity of sender in cleartext

Traditional Unix: username, password
— overuse of Lotk security hole

Challenges for Andrew System

Authenticate User

Authenticate Server

Prevent replay attacks

isolate users

Choice in Andrew: private key crypto system
⇒ identity of sender in cleartext

Traditional Unix: username, password
— overuse of Lotk security hole

Dilemma: What to use as identity and Private Key?

Andrew solution

- username and password only for login
- use ephemeral id and keys for subsequent venus-vize communication

Andrew solution

- username and password only for login
- use ephemeral id and keys for subsequent Venus-Vice communication

Three classes of client-server interaction

- login
- RPC session establishment
- file system access during session

Andrew solution

- username and password only for login
- use ephemeral id and keys for subsequent Venus-Vice communication

Three classes of client-server interaction

- login ⇒ username, password
- RPC session establishment
- file system access during session

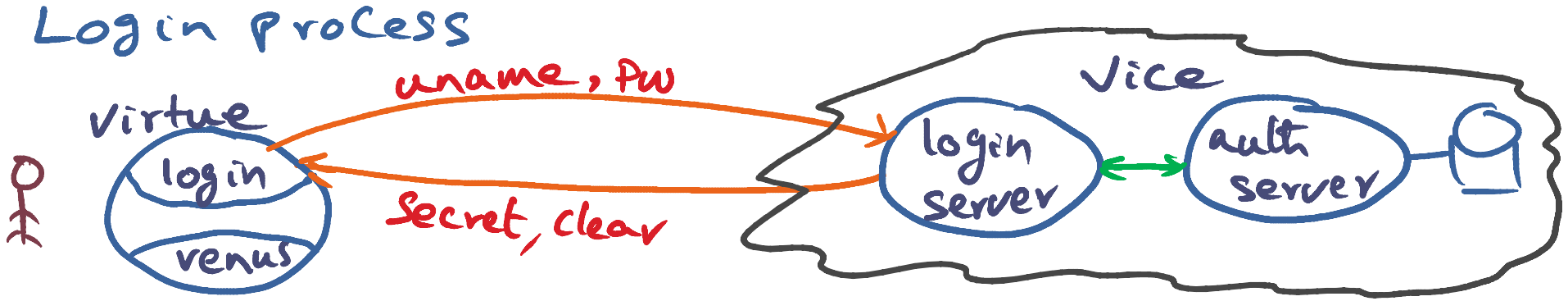
Andrew solution

- username and password only for login
- use ephemeral id and keys for subsequent Venus-Vice communication

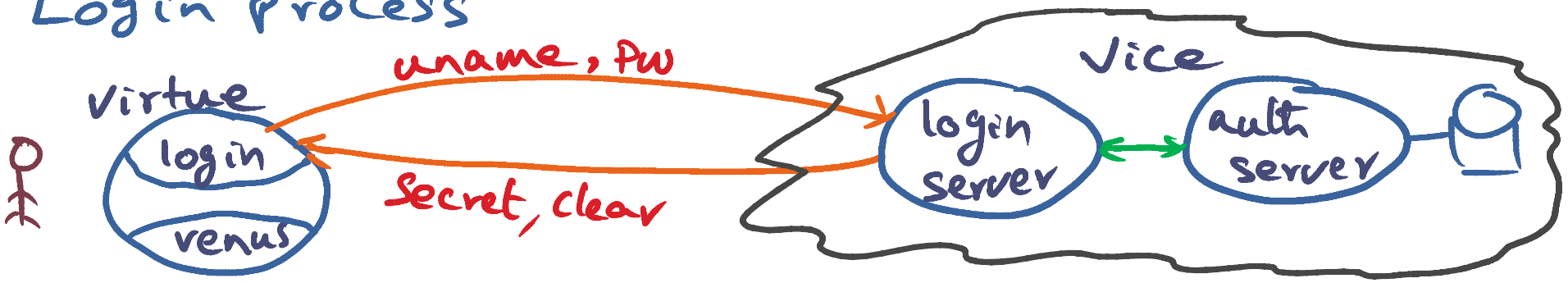
Three classes of client-server interaction

- login ⇒ username, password
- RPC session establishment
- file system access during session

} both use ephemeral id and keys


Login process
Virtue
login
venus
uname, PW
Secret, clear
Vice
login server
auth server

Login process



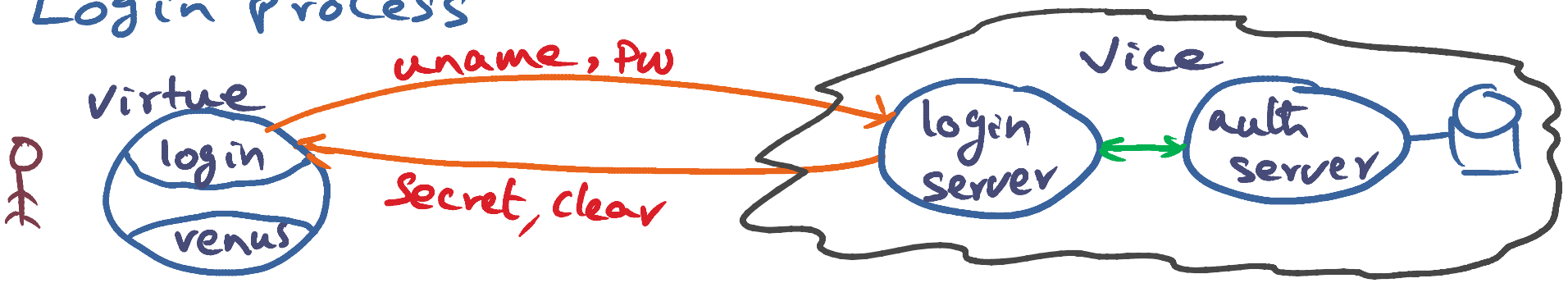
Cleartoken: Data structure

⇒ Extract handshake key client (HKC)

Secrettoken: cleartoken encrypted with key

known only to vice

Login process



Cleartoken: Data structure

⇒ Extract handshake key client (HKC)

Secrettoken: cleartoken encrypted with key

known only to vice

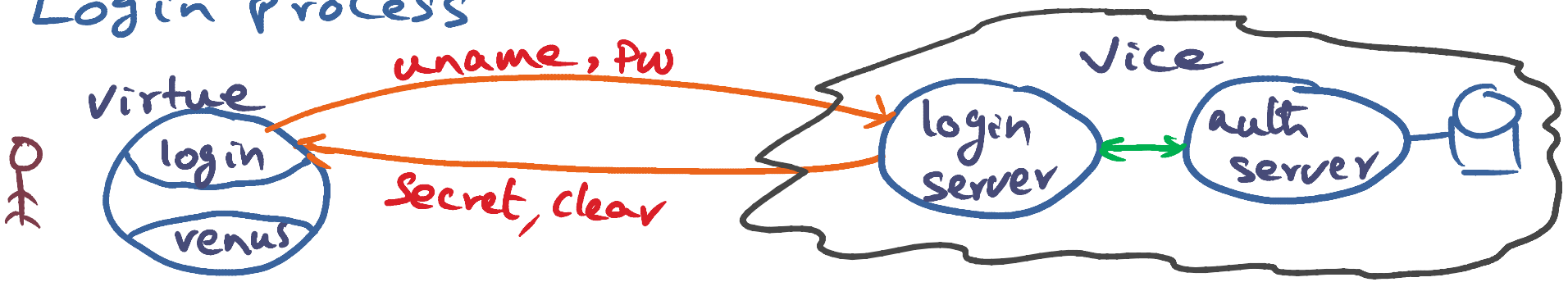
⇒ unique for this login session

⇓

use as ephemeral client-id

for this login session

Login process



Cleartoken: Data structure

⇒ Extract handshake key client (HKC)

Secrettoken: cleartoken encrypted with key known only to vice

⇒ unique for this login session

⇓

use as ephemeral client-id for this login session

use HKC as private key for establishing a new RPC session