Authentication in real world: Kerberos, SSH and SSL

Yinghua Wu

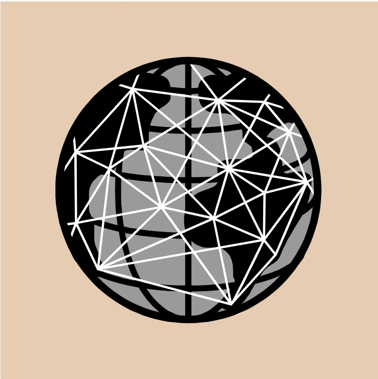
Dec 05, 2006

Where are we?

- After learning all the foundation of modern cryptography, we are ready to see some real world applications based on them.

  - What happened when you use your Yale netid and password? How does our system authenticate you?

  - Internet is a tough environment, security protocols need to deal with many different scenarios of attacks.



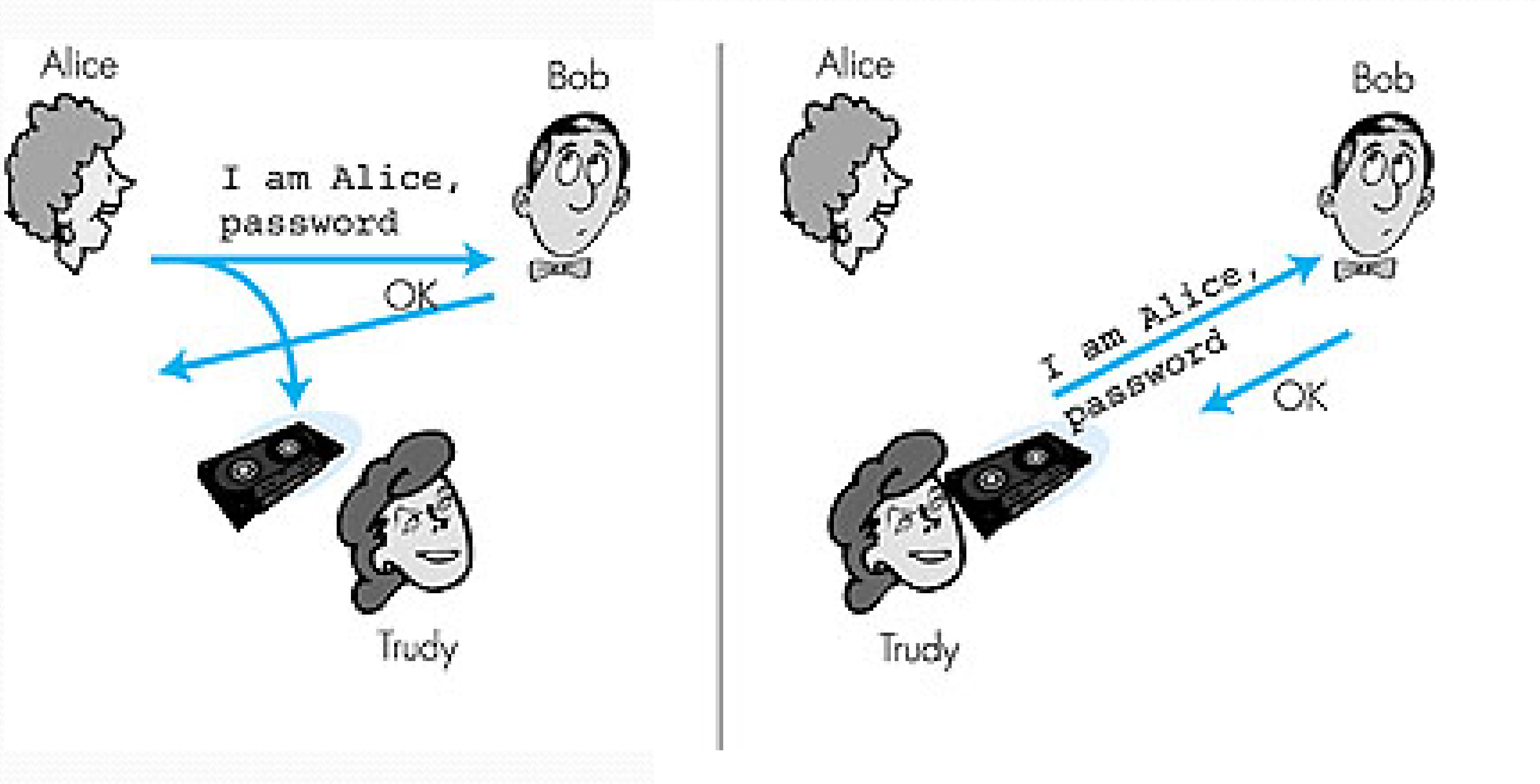
Think about Authentication

- Authentication provides a means to identify a client that requires access to some system.
  - Network services, such as telnet and pop3, need to authenticate individual users, by using their passwords.
- Note that firewalls can not replace authentication
  - For public computers with multiple users, blocking traffic based on IP addresses and port numbers is definitely insufficient.
- Usually, each user identity is associated with a secure password, which is used to authenticate the identity.

How can we send passwords through insecure network?

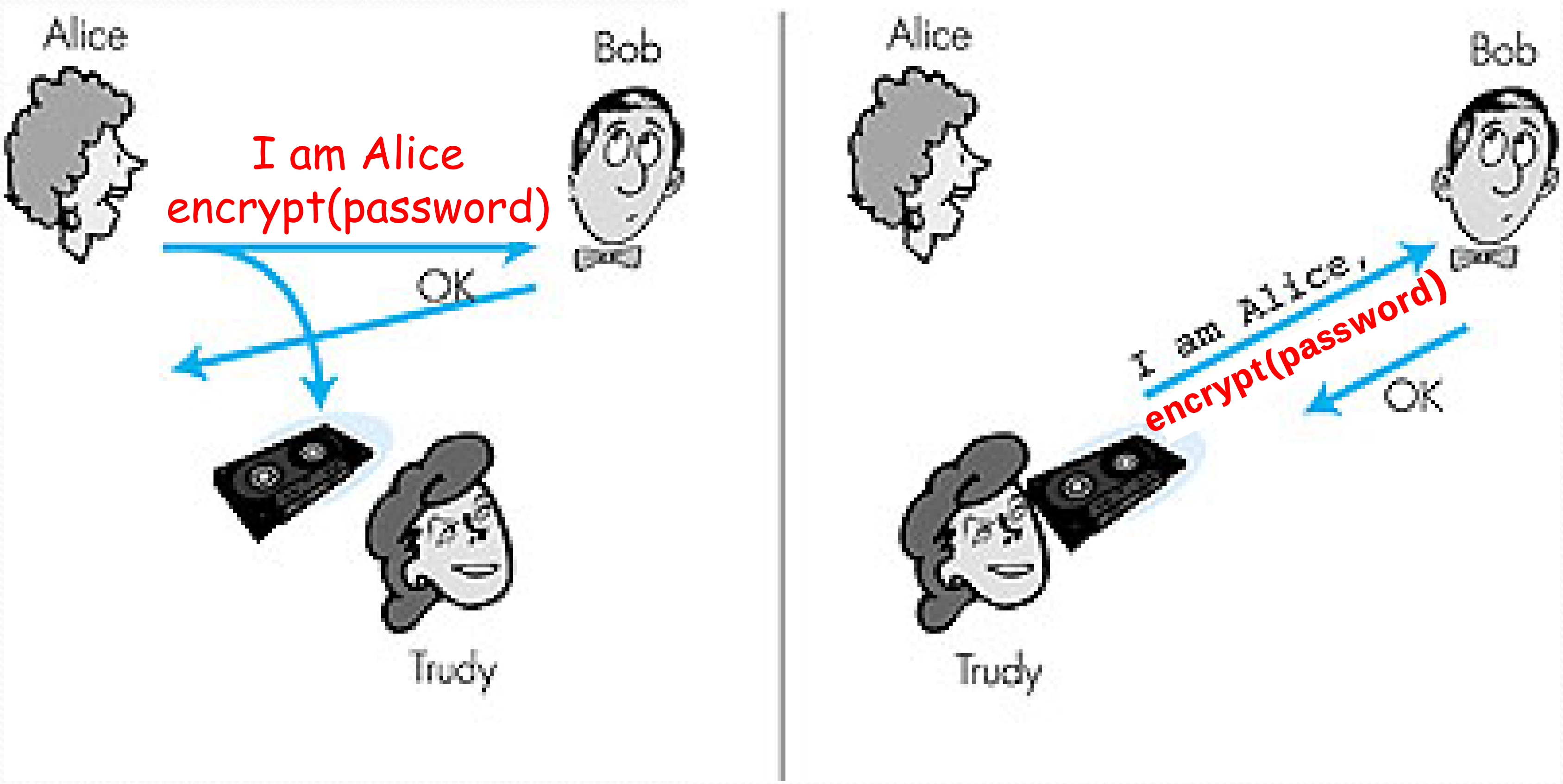
Authentication: First Try

Alice says "I am Alice" and sends her secret password to "prove" it.



Authentication: Yet Another Try

Alice says "I am Alice" and sends her *encrypted* secret password to "prove" it.



Use of Passwords with Challenges

Alice

Bob

Password

Offline
Operation

Client's Name

Challenge
(time-dependent
value, a randomly
select value, or both)

- Enter password
- Compute a hash value using challenge and password
- Send hash value

Verify received
hash value

The "O(N^2) Password Management Problem"

- Each of the N servers authenticates each of the N users
- Every server keeps track of the password of every user
- Thus a total of O(N^2) pieces of information items to manage

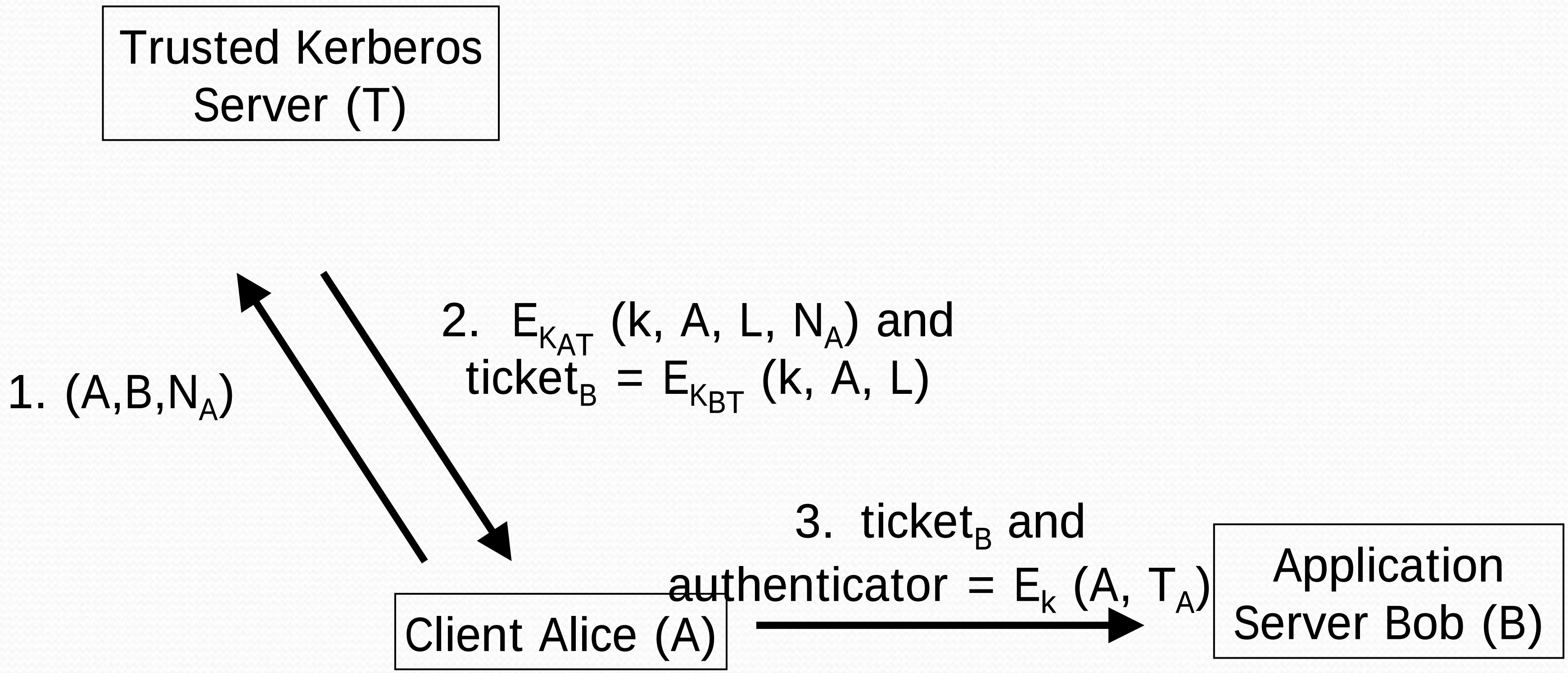
Kerberos' Objective: Provide an O(N) Solution

- Use a single authentication server that has trusted relationship with N clients and N servers. Thus, only O(N) keys to worry about.
- The authentication server will generate session keys (aka "tickets") for each client-server session

What is Kerberos?

- Part of project Athena (MIT).
- Trusted Kerberos Authentication Server (AS).
- Assumes that clients are not trustworthy.
- Each client has a secret *Kerberos key* used to authenticate itself to AS.
- The basic idea is that clients use their long-lived *Kerberos keys* to get short-lived session keys.

Kerberos Authentication



1. N_A: Nonce (random string) chosen by A
2. k: session key; L: lifetime
3. T_A: timestamp on A's local clock

Practical Considerations

- The dilemma of security and efficiency:
  - Tickets have a relatively long lifetime and can be used many times.
  - Authenticators have a relatively short lifetime and can be used only once.
- Still not scalable. So to scale Kerberos:
  - The trusted server is split into two parts, an *authentication server(AS)* and a *ticket-granting server(TGS).*
  - The nodes are partitioned into several groups, each with its own server.
  - For Alice to contact Bob, she first goes to AS to get a ticket that lets her talk to Bob’s TGS from who she gets a ticket to talk to Bob.

Advantages of Kerberos

- Passwords aren’t exposed to eavesdropping
- Password is only typed to the local workstation
  - It never travels over the network
  - It is never transmitted to a remote server
- Password guessing more difficult
- Single Sign-on
  - More convenient: only one password, entered once
  - Users may be less likely to store passwords
- Stolen tickets hard to reuse
  - Need authenticator as well, which can’t be reused
- Much easier to effectively secure a small set of limited access machines (the AS’s)
- Easier to recover from host compromises
- Centralized user account administration

Kerberos caveats

- Kerberos server can impersonate anyone
- AS is a single point of failure
  - Can have replicated AS's
- AS could be a performance bottleneck
  - Everyone needs to communicate with it frequently
  - Not a practical concern these days
  - Having multiple AS's alleviates the problem
- If local workstation is compromised, user's password could be stolen by a trojan horse
  - Only use a desktop machine or laptop that you trust
  - Use hardware token pre-authentication
- Kerberos vulnerable to password guessing attacks
  - Choose good passwords!
  - Use hardware pre-authentication
  - Hardware tokens, Smart cards etc

Secure Shell (SSH)

- To build up a secure channel between a local computer and a remote computer.
- Uses public key cryptography to authenticate the remote computer and exchange encryption keys.

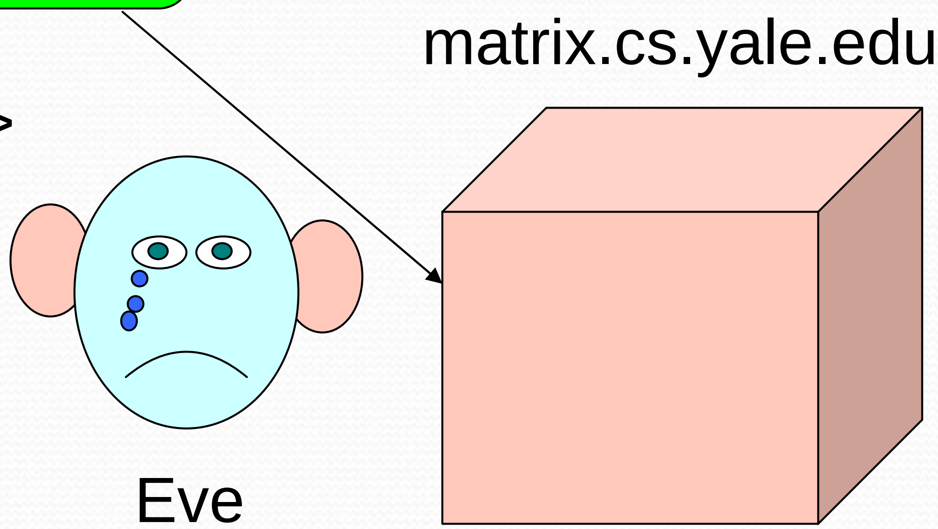
Simplified SSH Protocol

Terminal

```
Login: zm25
Password: ************
```

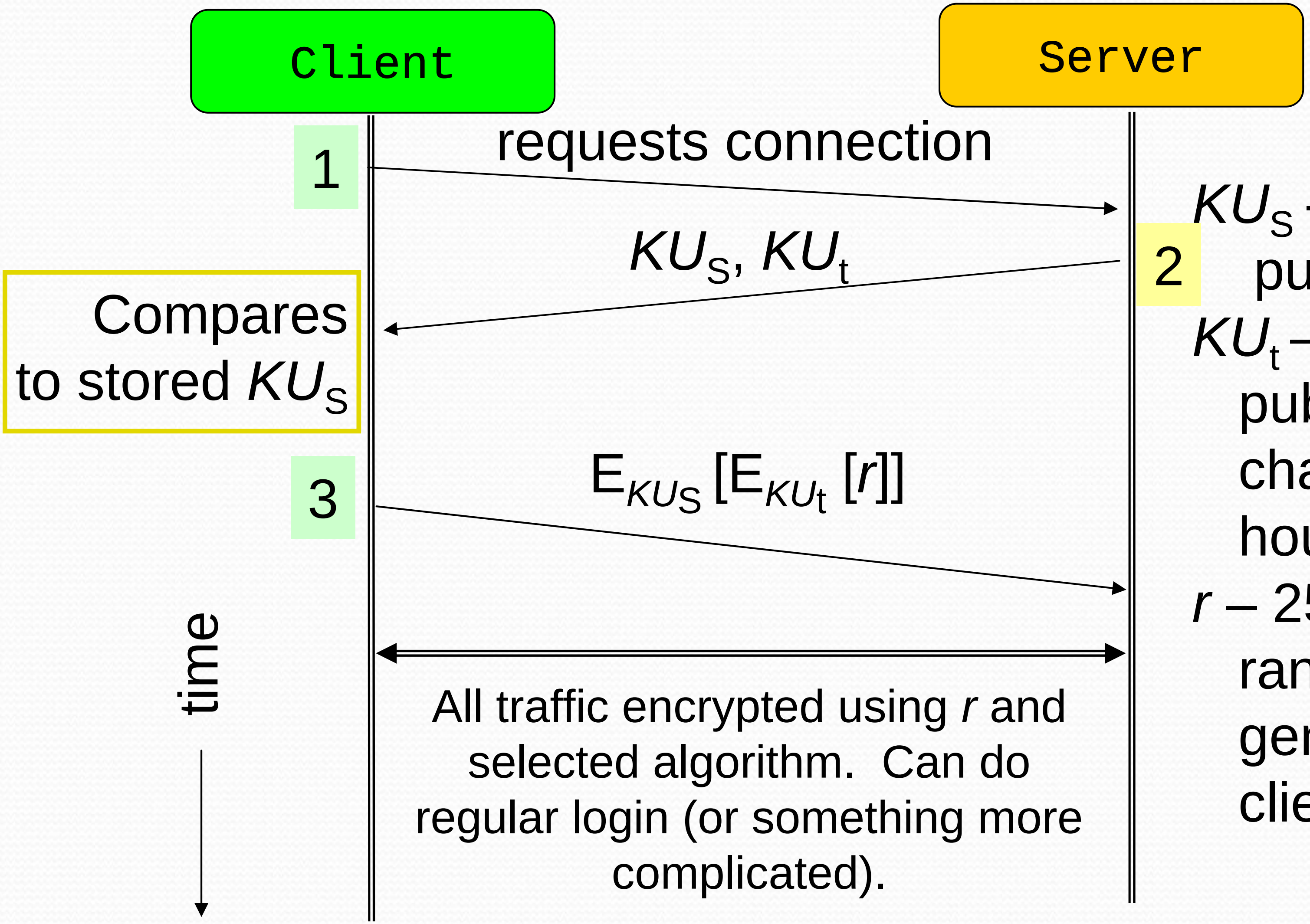
login sends

$E_{KU_{matrix}}$<“zm25”, *password*>



Can’t decrypt without KR_{matrix}

Actual SSH Protocol



KU_S - server's public host key

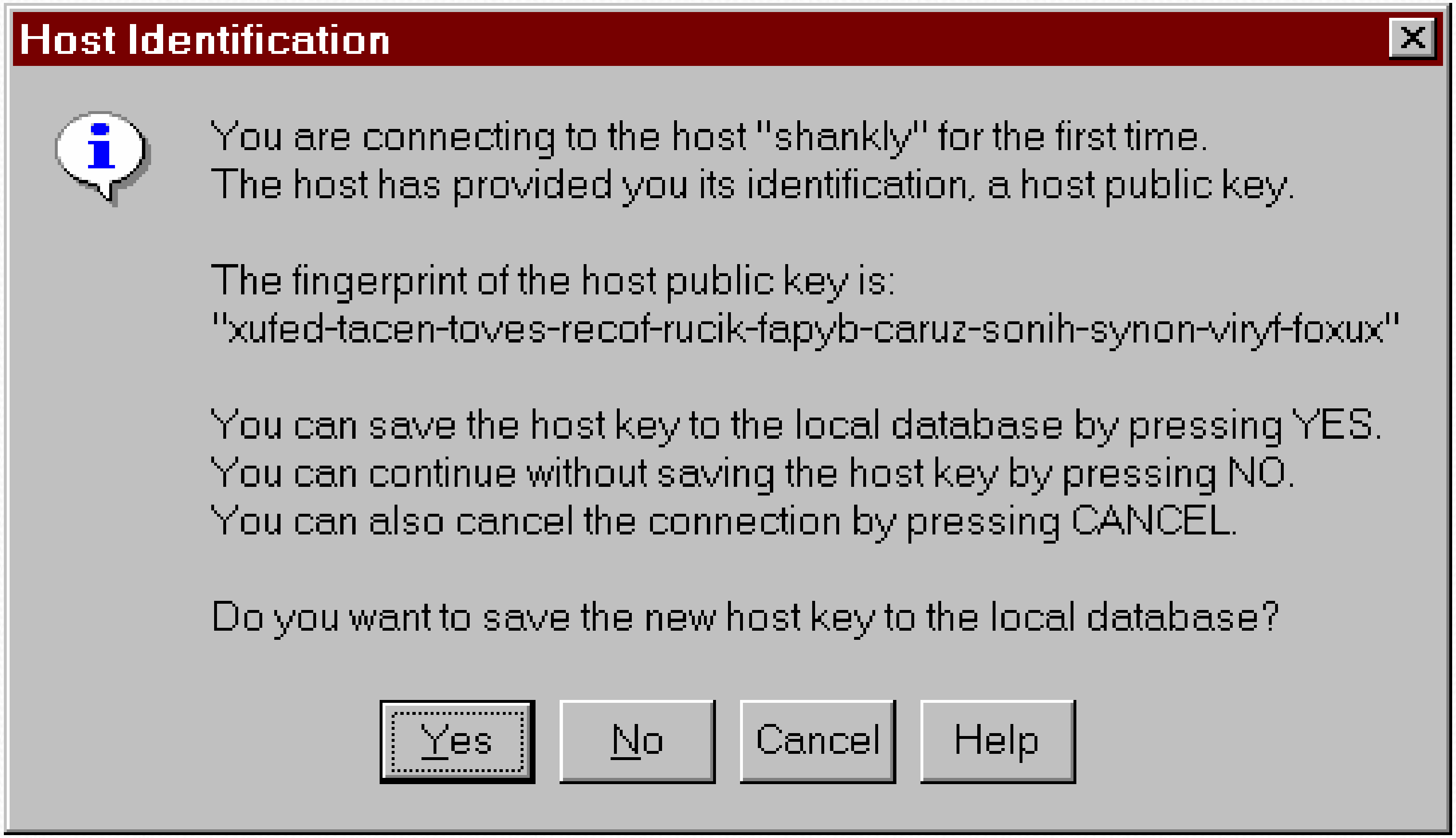
KU_t – server's public key, changes every hour

r – 256-bit random number generated by client

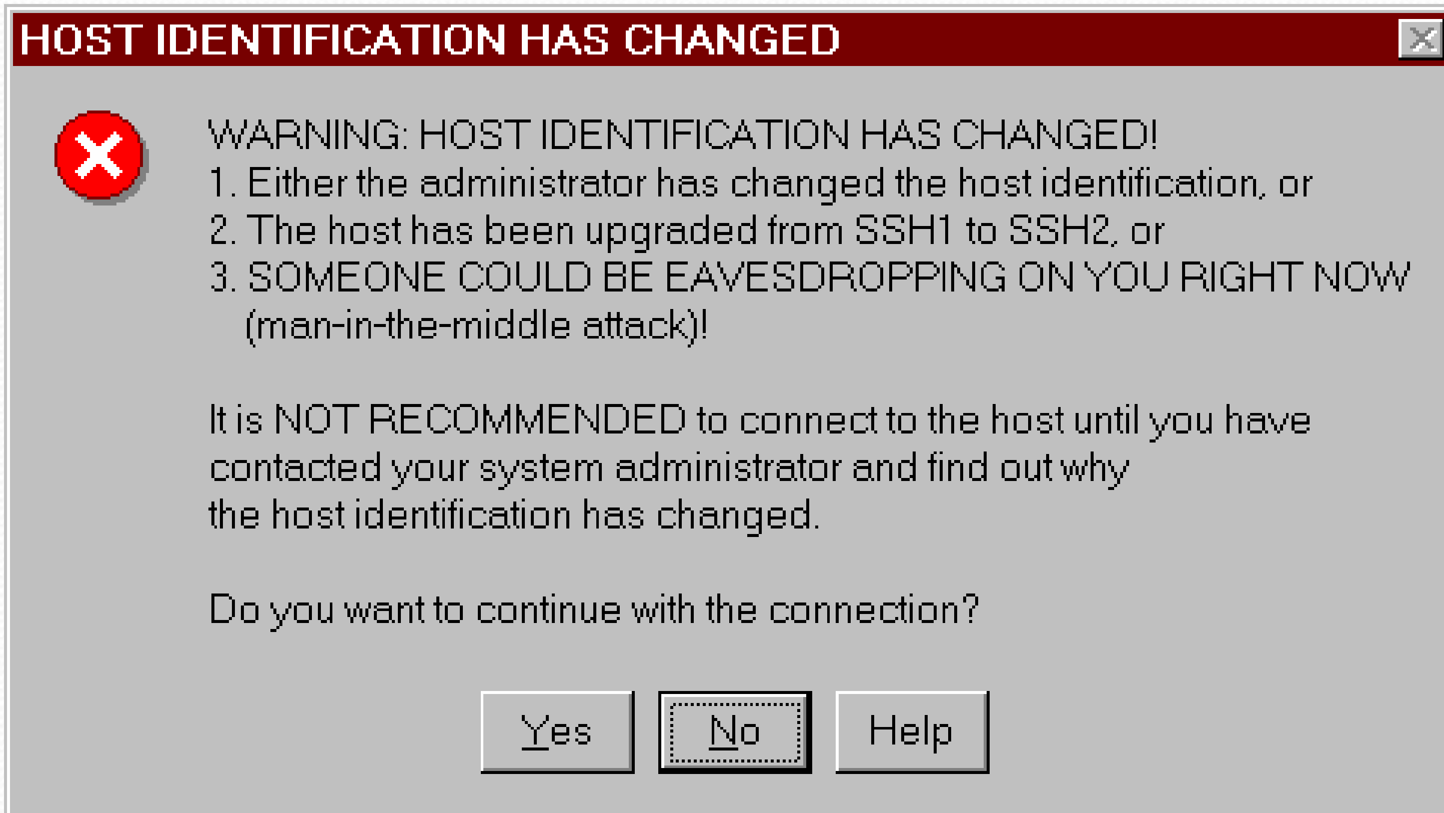
Comparing to stored KU_S

- It better be stored securely
  - PuTTY stores it in windows registry (HKEY_CURRENT_USER\Software\SimonTatham\PuTTY\SshHostKeys)

ssh.com's SSH


Host Identification
You are connecting to the host "shankly" for the first time.
The host has provided you its identification, a host public key.
The fingerprint of the host public key is:
"xufed-tacen-toves-recof-rucik-fapyb-caruz-sonih-synon-viryf-foxux"
You can save the host key to the local database by pressing YES.
You can continue without saving the host key by pressing NO.
You can also cancel the connection by pressing CANCEL.
Do you want to save the new host key to the local database?
Yes
No
Cancel
Help

ssh Error


HOST IDENTIFICATION HAS CHANGED
WARNING: HOST IDENTIFICATION HAS CHANGED!
1. Either the administrator has changed the host identification, or
2. The host has been upgraded from SSH1 to SSH2, or
3. SOMEONE COULD BE EAVESDROPPING ON YOU RIGHT NOW (man-in-the-middle attack)!
It is NOT RECOMMENDED to connect to the host until you have contacted your system administrator and find out why the host identification has changed.
Do you want to continue with the connection?
Yes
No
Help

Secure Socket Layer (SSL)

- Security at the Transport Layer
- Developed by Netscape to provide security in WWW browsers and servers
- SSL is the basis for the Internet standard protocol – Transport Layer Security (TLS) protocol (compatible with SSLv3)
- Designed for communications between computers with no previous knowledge of each other’s capabilities.

Secure Socket Layer (SSL), cont

SSL consists of two main components:

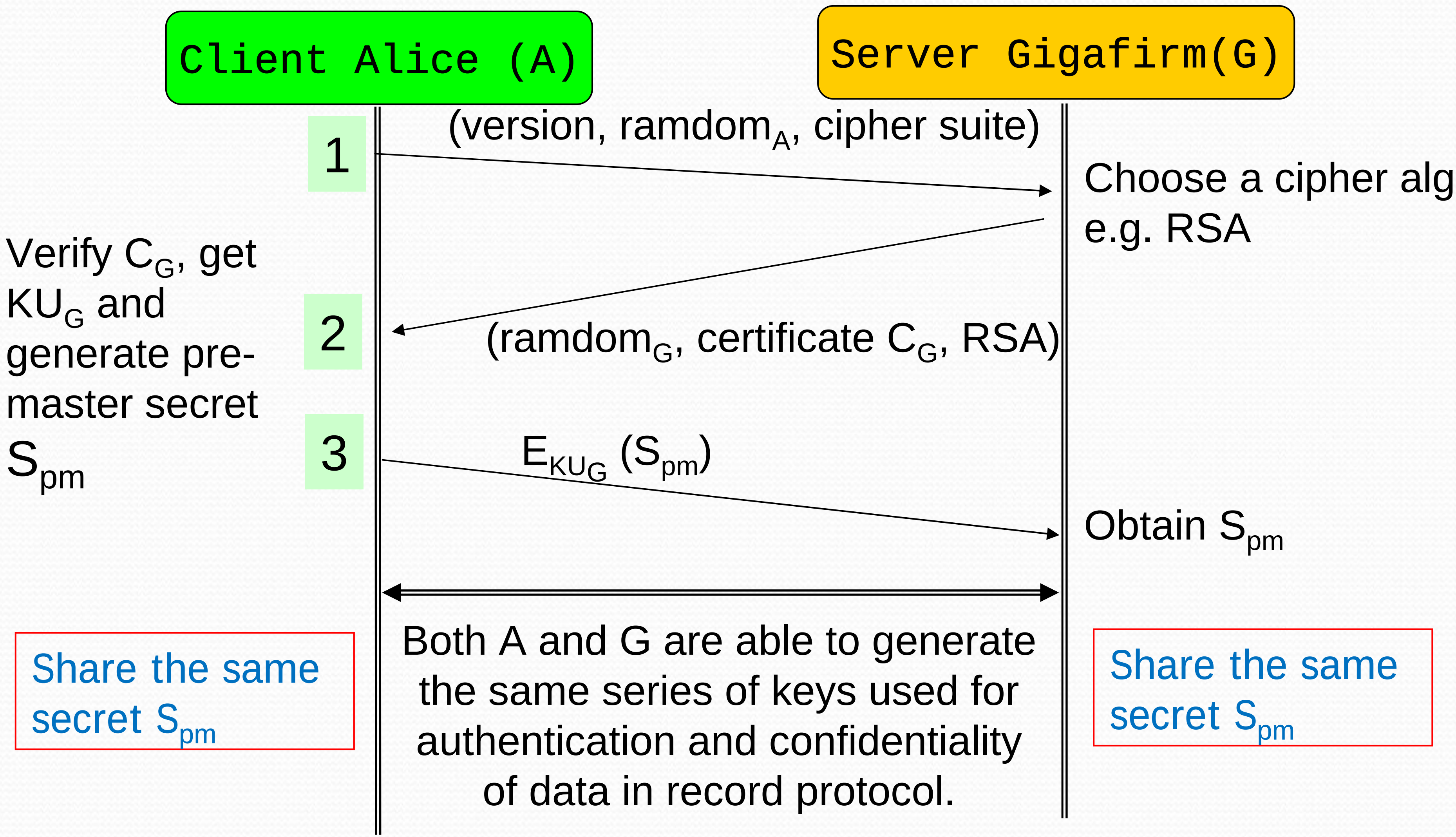
- Record protocol
  - Responsible for compressing and encrypting the bulk of the data sent between two entities;

- Handshake protocol
  - Responsible for setting up and maintaining the parameters used by the record protocol.

Preliminary: What is certificate?

- A *certificate* is a quantity of information that has been signed by its publisher, commonly referred to as the *certificate authority(CA)*.
- The data are encrypted using the CA's private key.
  - e.g. $C = S_{KR_{CA}}(A, P_A)$, in which A is the identity and P_A is A's public key and S is a signature function.
- Verifying the certificate by $V_{KU_{CA}}(C, A, P_A)$ to verify A's public key, in which V is a verification predicate.

An example of SSL protocols



Acknowledgements

- Credits of some slides and images:
  - http://www.upenn.edu/computing/pennkey/docs/kerbpres/200207Kerberos.htm
  - http://www.eecs.harvard.edu/cs143/
  - http://www.cs.virginia.edu/~evans/cs551/
  - http://zoo.cs.yale.edu/classes/cs433/
- Thanks to Zheng Ma for his slides in the previous course.