

---

---

# Kerberos, S/Key

8 June 2010  
Lecture 11

Slide Credits: Steve Zdancewic (UPenn)

---

# Topics for Today

---

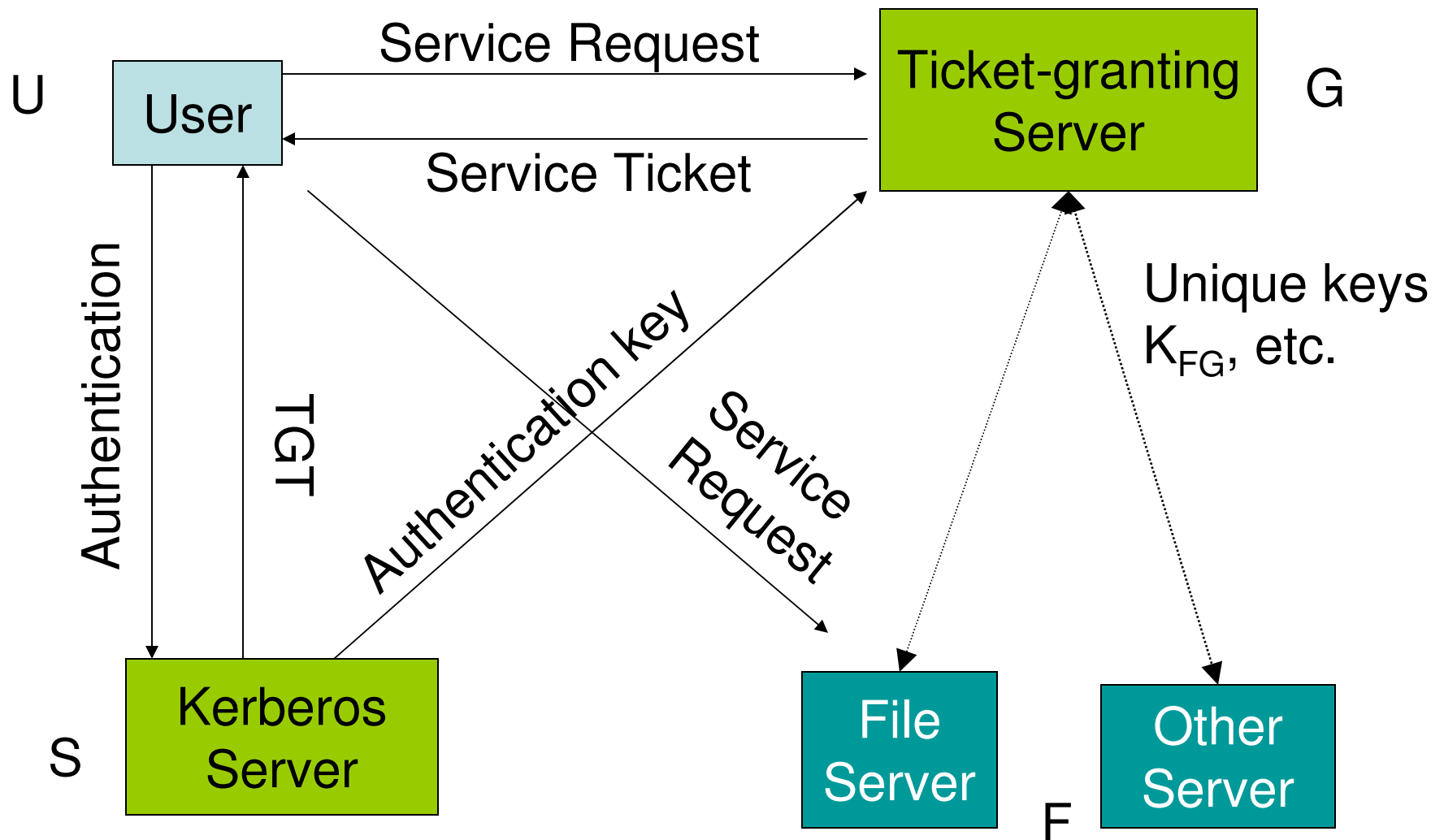
- Kerberos
- S/Key: One Time Passwords

# Kerberos

---

- Key exchange protocol developed at MIT in the late 1980s
- Central server provides “tickets”
- *Tickets* – (also known as *capabilities*):
  - Unforgeable
  - Nonreplayable
  - Authenticated
  - Represents authority
- Designed to work with NFS (network file system)
- Also saves on authenticating for each service
  - E.g. with ssh

# Kerberos



# Kerberos Login

---

- U = User's machine
- S = Kerberos Server
  - Has a database of user “passwords”:  $\text{userID} \rightarrow k_{\text{pwd}}$
- G = Ticket granting server
- $U \rightarrow S: \text{userID}, G, n_U$
- $S \rightarrow U: k_{\text{pwd}}\{n_U, K_{UG}\}, K_{SG}\{T(U,G)\}$
- $S \rightarrow G: K_{SG}\{K_{UG}, \text{userID}\}$
- $T(X,Y) = X, Y, L, K_{XY}$

Kerberos ticket granting ticket

Session Key

Ticket lifetime

# Kerberos Service Request

---

- Requesting a service from server F
- $U \rightarrow G: K_{UG}\{\text{userID, timestamp}\}, K_{SG}\{T(U,G)\}, \text{req}(F), n'_U$
- $G \rightarrow U: K_{UG}\{K_{UF}, n'_U\}, K_{FG}\{T(U,F)\}$
- $U \rightarrow F: K_{UF}\{\text{userID, timestamp}\}, K_{FG}\{T(U,F)\}$

# Kerberos Benefits

---

- Distributed access control
  - No passwords communicated over the network
- Cryptographic protection against spoofing
  - All accesses mediated by G (ticket granting server)
- Limited period of validity
  - Servers check timestamps against ticket validity
  - Limits window of vulnerability
- Timestamps prevent replay attacks
  - Servers check timestamps against their own clocks to ensure “fresh” requests
- Mutual authentication
  - User sends nonce challenges

# Kerberos Drawbacks

---

- Requires available ticket granting server
  - Could become a bottleneck
  - Must be reliable
- All servers must trust G, G must trust servers
  - They share unique keys
- Kerberos requires synchronized clocks
  - Replay can occur during validity period
  - Not easy to synchronize clocks
- User's machine could save and replay passwords
  - Password is a weak spot
- Kerberos does not scale well
  - Hard to replicate authentication server and ticket granting server
  - Duplicating keys is bad, extra keys means more management

# So Far

---

- Kerberos
- S/Key: One Time Passwords

# One Time Passwords

---

- Shared lists.
- Sequentially updated.
- One-time password sequences based on a one-way (hash) function.
  
- "Dongles"
  - Small devices that generate a sequence of random numbers from a secret seed.
  - Synchronized with the remote location when the dongle is assigned to a user
  - Often requires a pin or other password for local authentication
  - Can be stolen or lost!

# Hash-based 1-time Passwords

---

- Alice identifies herself to verifier Bart using a well-known one-way hash function  $H$ .
- One-time setup.
  - Alice chooses a secret  $w$ .
  - Fixes a constant  $t$  for the number of times the authentication can be done.
  - Alice securely transfers  $H^t(w)$  to Bart

$$\underbrace{H(H(H\dots(H(w))\dots))}_{t \text{ times}}$$

# Protocol actions

---

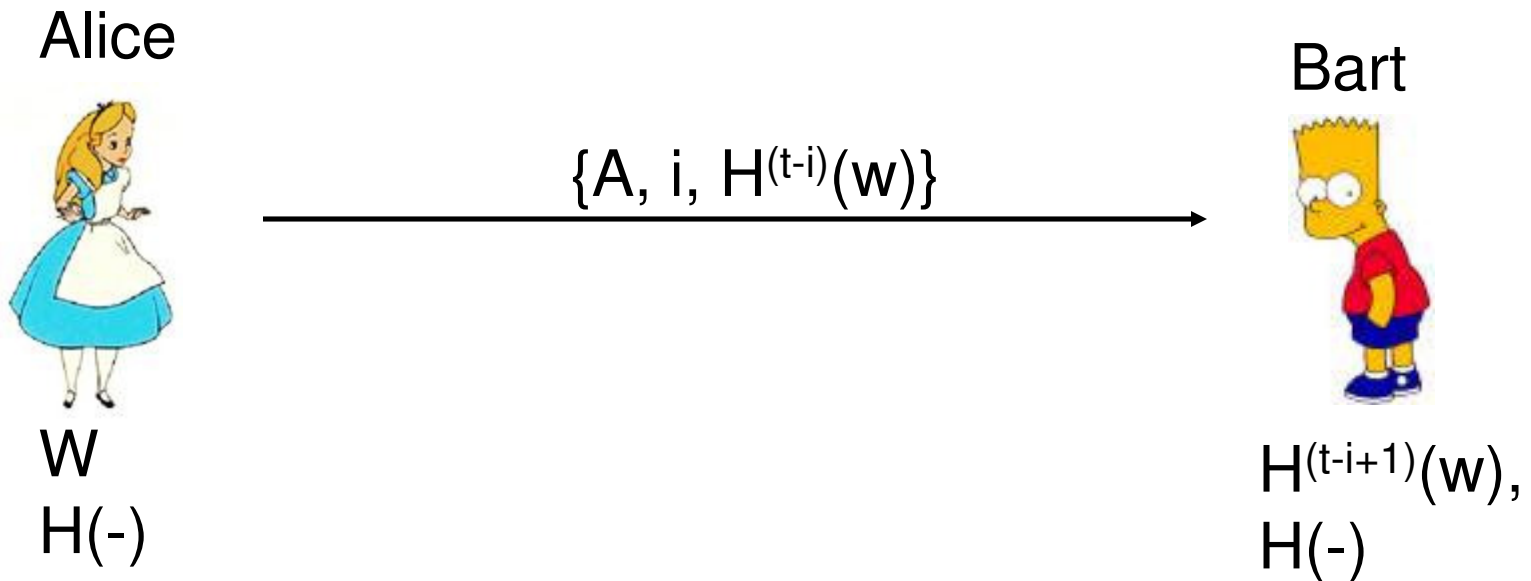
For session  $i$ , Alice does the following to identify herself:

- A computes  $w' = H^{(t-i)}(w)$  and sends the value to B.
  - B checks that  $i$  is the correct session (i.e. that the previous session was  $i-1$ ) and checks to see if  $H(w') = v$  where  $v$  was the last value provided by A (as part of session  $i-1$ ).
  - B saves  $w'$  and  $i$  for use in the next session.
- It's hard to compute  $x$  from  $H(x)$ .
    - Even though attacker sees  $H^{(t-i)}(x)$ , can't guess the next message  $H^{(t-(i+1))}(x)$ .
- 

**Example:**  $t = 10$ .

- For session 1: A sends  $H^9(w) \rightarrow p1$
- For session 2: A sends  $H^8(w) \rightarrow p2$ 
  - $H(p2) = p1$
- For session 3: A sends  $H^7(w) \rightarrow p3$ 
  - $H(p3) = p2$
- ...
- For session 10: A sends  $w \rightarrow p10$ 
  - $H(w) = p9$

# One-time passwords: $i^{\text{th}}$ authentication



- Alice does the following to identify herself:
  - A computes  $w' = H^{(t-i)}(w)$  and transmits the value to B.
  - B checks that  $i$  is the correct session (i.e., that the previous session was  $i-1$ ) and checks to see if  $H(w') = v$  where  $v$  was the last value provided by A (as part of session  $i-1$ ).
  - B saves  $w'$  and  $i$  for use in the next session.

# S/Key Passwords

---

- Hash-based one-time authentication used in practice
  - RFC 1760 / 2289
- Internally, S/Key uses 64 bit numbers
- For human use, each 64 bit number is mapped to 6 short words:
  - Example: "ROY HURT SKI FAIL GRIM KNEE"
- Should be used in conjunction with other encryption to prevent man-in-the-middle attacks

# Conclusion

---

- Kerberos
- S/Key: One Time Passwords