

Introduction to Cryptography

Lecture 10

SSL, secret sharing

Benny Pinkas

SSL/TLS

- General structure of secure HTTP connections
 - To connect to a secure web site using SSL or TLS, we send an `https://` command
 - The web site sends back a public key⁽¹⁾, and a certificate.
 - Our browser
 - Checks that the certificate belongs to the url we're visiting
 - Checks the expiration date
 - Checks that the certificate is signed by a CA whose public key is known to the browser
 - Checks the signature
 - If everything is fine, it chooses a session key and sends it to the server encrypted with RSA using the server's public key

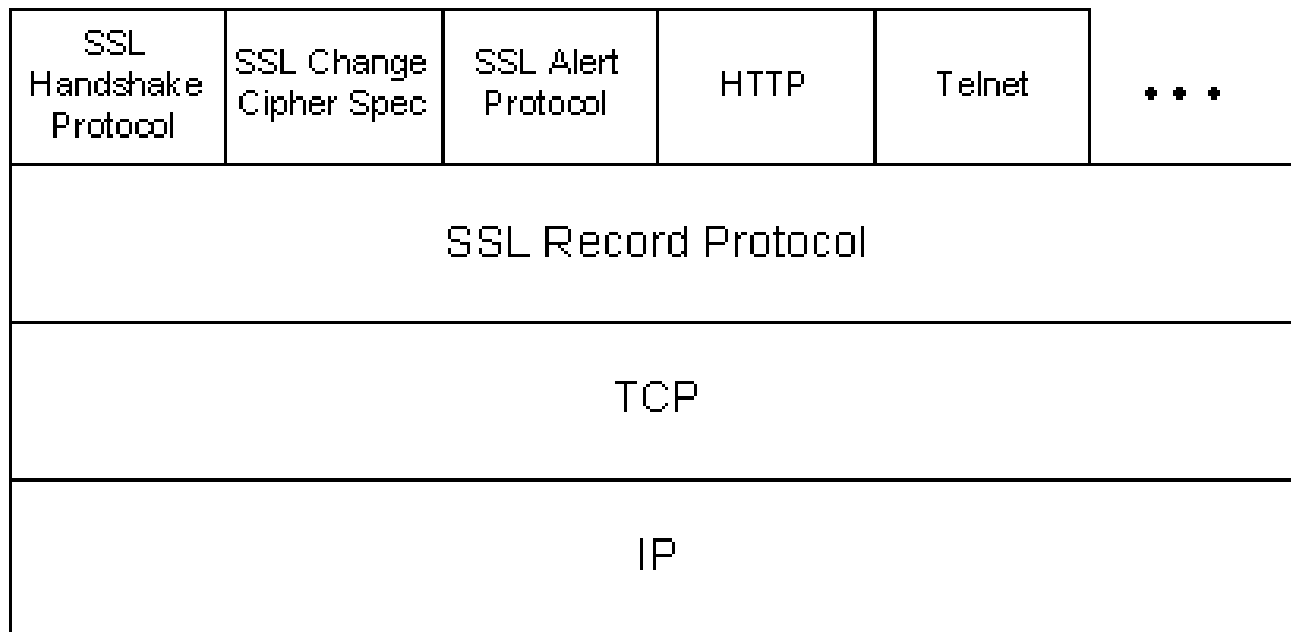
⁽¹⁾ This is a very simplified version of the actual protocol.

SSL/TLS

- SSL (Secure Sockets Layer)
 - SSL v2
 - Released in 1995 with Netscape 1.1
 - A flaw found in the key generation algorithm
 - SSL v3
 - Improved, released in 1996
 - Public design process
- TLS (Transport Layer Security)
 - IETF standard, RFC 2246
- Common browsers support all these protocols

SSL Protocol Stack

- SSL/TLS operates over TCP, which ensures reliable transport.
- Supports any application protocol (usually used with http).



SSL/TLS Overview

- Handshake Protocol - establishes a session
 - Agreement on algorithms and security parameters
 - Identity authentication
 - Agreement on a key
 - Report error conditions to each other
- Record Protocol - Secures the transferred data
 - Message encryption and authentication
- Alert Protocol – Error notification (including “fatal” errors).
- Change Cipher Protocol – Activates the pending crypto suite

Simplified SSL Handshake

Client

Server

I want to talk, ciphers I support, R_C

Certificate (PK_{Server}), cipher I choose, R_S

$\{S\}_{PK_{server}}$, {keyed hash of handshake message}

compute
 $K = f(S, R_C, R_S)$

{keyed hash of handshake message}

compute
 $K = f(S, R_C, R_S)$

Data protected by keys derived from K

A typical run of a TLS protocol

- C -> S
 - ClientHello.protocol.version = “TLS version 1.0”
 - ClientHello.random = T_C, N_C
 - ClientHello.session_id = “NULL”
 - ClientHello.crypto_suite = “RSA: encryption.SHA-1:HMAC”
 - ClientHello.compression_method = “NULL”
- S -> C
 - ServerHello.protocol.version = “TLS version 1.0”
 - ServerHello.random = T_S, N_S
 - ServerHello.session_id = “1234”
 - ServerHello.crypto_suite = “RSA: encryption.SHA-1:HMAC”
 - ServerHello.compression_method = “NULL”
 - ServerCertificate = pointer to server’s certificate
 - ServerHelloDone

Some additional issues

- More on S -> C
 - The ServerHello message can also contain Certificate Request Message
 - I.e., server may request client to send its certificate
 - Two fields: certificate type and acceptable CAs
- Negotiating crypto suites
 - The crypto suite defines the encryption and authentication algorithms and the key lengths to be used.
 - ~30 predefined standard crypto suites
 - Selection (SSL v3): Client proposes a set of suites. Server selects one.

Key generation

- Key computation:
 - The key is generated in two steps:
 - *pre-master secret* S is exchanged during handshake
 - *master secret* K is a 48 byte value calculated using pre-master secret and the random nonces
- Session vs. Connection: a *session* is relatively long lived. Multiple TCP *connections* can be supported under the same SSL/TSL connection.
- For each connection: 6 keys are generated from the master secret K and from the nonces. (For each direction: encryption key, authentication key, IV.)

TLS Record Protocol

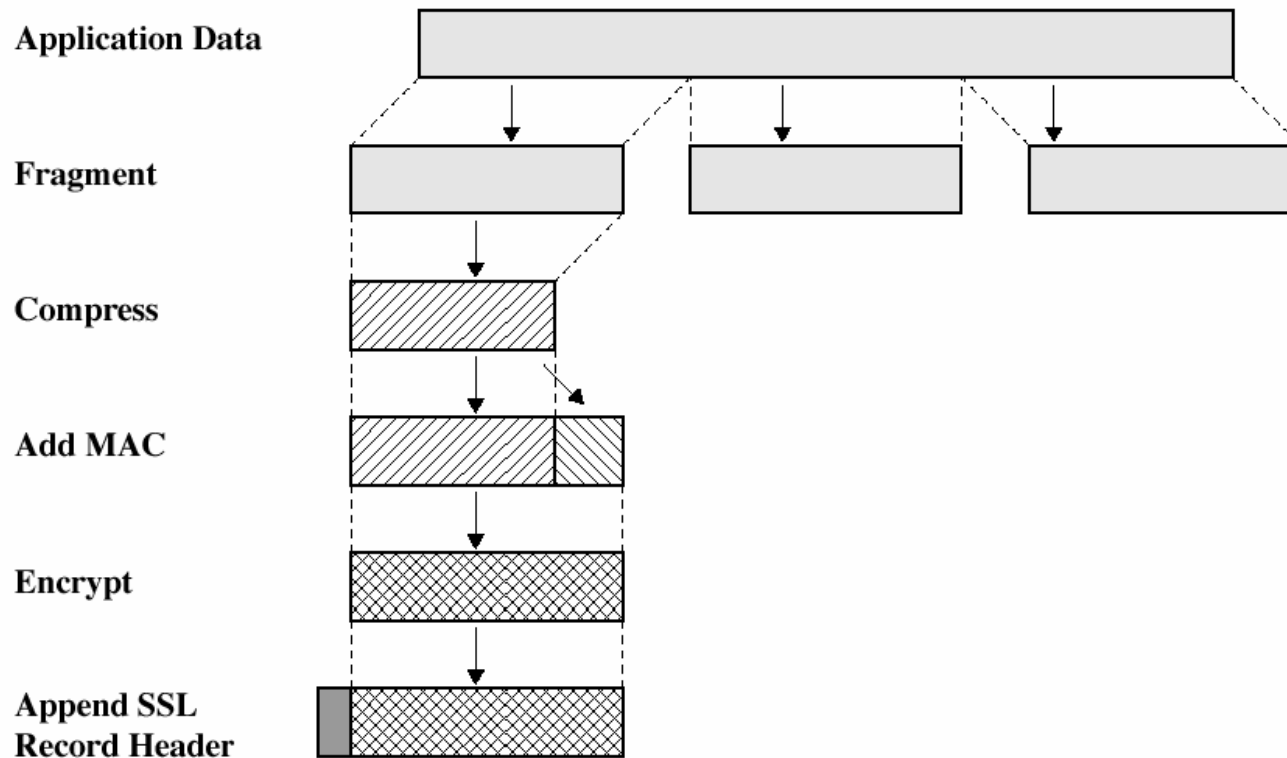


Figure 17.3 SSL Record Protocol Operation

Secret Sharing

Secret Sharing

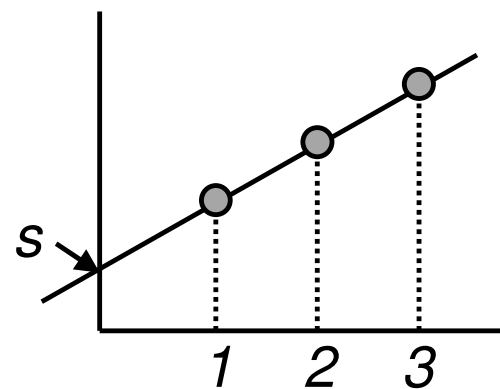
- 3-out-of-3 secret sharing:
 - Three parties, A, B and C.
 - Secret K .
 - No two parties should know anything about K , but all three together should be able to retrieve it.
- How about the following scheme:
 - Let $K = k_1 k_2 \dots k_m$ be the bit representation of K . m is a power of 3.
 - Party A receives $k_1, \dots, k_{m/3}$. Party B receives $k_{m/3+1}, \dots, k_{2m/3}$. Party C receives $k_{2m/3+1}, \dots, k_m$.
 - All three parties can recover K .
 - Why doesn't this scheme satisfy the definition of secret sharing?

Secret Sharing

- Define *shares* for A,B,C in the following way:
 - (S_A, S_B, S_C) is a random triple, subject to the constraint that $S_A \oplus S_B \oplus S_C = K$; or
 - S_A and S_B are random, and $S_C = S_A \oplus S_B \oplus K$.
- What if it is required that any one of the parties should be able to compute K ?
 - Set $S_A = S_B = S_C = K$

t -out-of- n secret sharing

- Provide shares to n parties, satisfying
 - Recoverability: any t shares enable the reconstruction of the secret.
 - Secrecy: any $t-1$ shares reveal nothing about the secret.
- We saw 1 -out-of- n and n -out-of- n secret sharing.
- Consider 2 -out-of- n secret sharing.
 - Define a line which intersects the Y axis at S
 - The shares are points on the line
 - Any two shares define S
 - A single share reveals nothing



t -out-of- n secret sharing

- Fact: Let F be a field. Any $d+1$ pairs (a_i, b_i) define a unique polynomial P of degree $\leq d$, s.t. $P(a_i)=b_i$. (assuming $d < |F|$).
- Shamir's secret sharing scheme:
 - Choose a large prime and work in the field Z_p .
 - The secret S is an element in the field.
 - Define a polynomial P of degree $t-1$ by choosing random coefficients a_1, \dots, a_{t-1} and defining
$$P(x) = a_{t-1}x^{t-1} + \dots + a_1x + \underline{S}.$$
 - The share of party j is $(j, P(j))$.

t-out-of-n secret sharing

- Reconstruction of the secret:
 - Assume we have $P(x_1), \dots, P(x_t)$.
 - Use Lagrange interpolation to compute the unique polynomial of degree $\leq t-1$ which agrees with these points.
 - Output the free coefficient of this polynomial.
- Lagrange interpolation
 - $P(x) = \sum_{i=1..t} P(x_i) \cdot L_i(x)$
 - where $L_i(x) = \prod_{j \neq i} (x - x_j) / \prod_{j \neq i} (x_i - x_j)$
 - (Note that $L_i(x_i) = 1$, $L_i(x_j) = 0$ for $j \neq i$.)
 - I.e., $S = \sum_{i=1..t} P(x_i) \cdot \prod_{j \neq i} x_j / \prod_{j \neq i} (x_i - x_j)$

Properties of Shamir's secret sharing

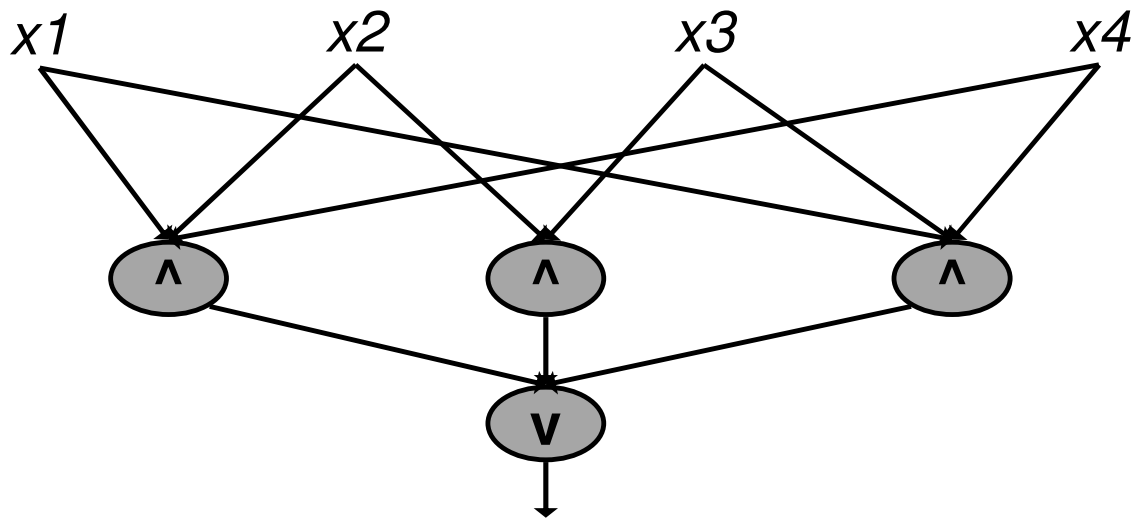
- Perfect secrecy: Any $t-1$ shares give no information about the secret: $\Pr(\text{secret}=s \mid P(1), \dots, P(t-1)) = \Pr(\text{secret}=s)$. (Security is not based on any assumptions.)
- Ideal size: Each share is the same size as the secret.
- Extendable: Additional shares can be easily added.
- Flexible: different weights can be given to different parties by giving them more shares.
- Homomorphic property: Suppose $P(1), \dots, P(n)$ are shares for S , and $P'(1), \dots, P'(n)$ are shares for S' , then $P(1)+P'(1), \dots, P(n)+P'(n)$ are shares for $S+S'$.

General secret sharing

- P is the set of users (say, n users).
- $A \in \{1,2,\dots,n\}$ is an authorized subset if it is authorized to access the secret.
- Γ is the set of authorized subsets.
- For example,
 - $P = \{1,2,3,4\}$
 - $\Gamma = \text{Any set containing one of } \{ \{1,2,4\}, \{1,3,4\}, \{2,3\} \}$
 - Not supported by threshold secret sharing
- If $A \in \Gamma$ and $A \subseteq B$, then $B \in \Gamma$.
- $A \in \Gamma$ is a minimal authorized set if there is no $C \subseteq A$ such that $C \in \Gamma$.
- The set of minimal subsets Γ_0 is called the basis of Γ .

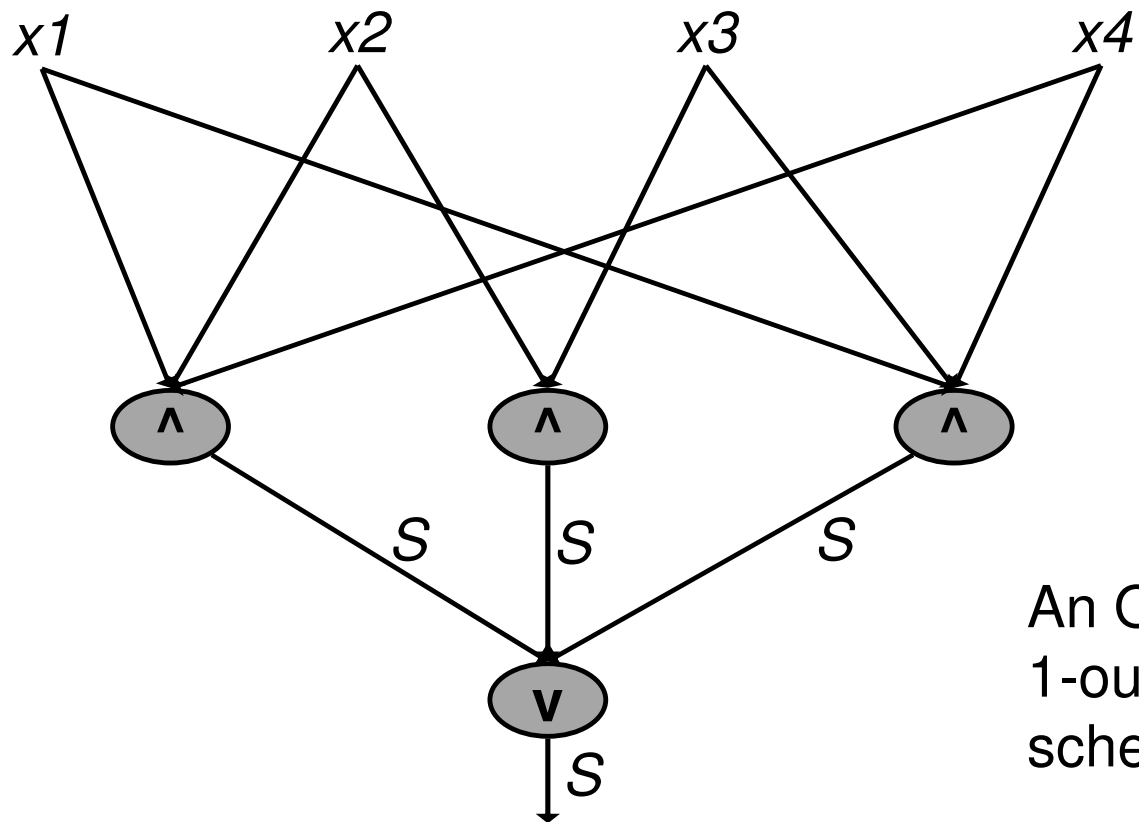
The monotone circuit construction (Benaloh-Leichter)

- A Boolean circuit C with OR and AND gates, is *monotone*. Namely, if $C(x)=1$, then changing bits of x from 0 to 1 does not change the result to 0.
- Given Γ construct a circuit C s.t. $C(A)=1$ iff $A \in \Gamma$.
 - $\Gamma_0 = \{ \{1,2,4\}, \{1,3,4\}, \{2,3\} \}$



Handling OR gates

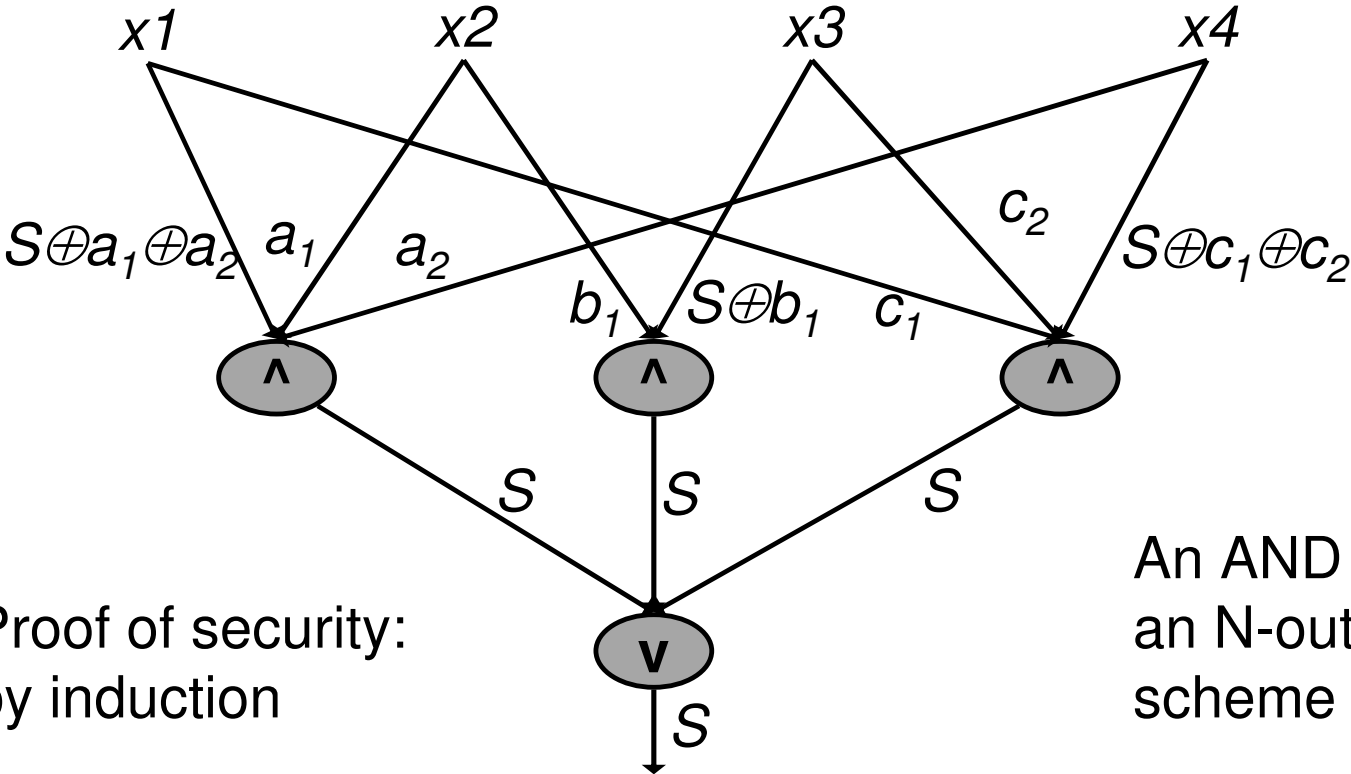
Starting from the output gate and going backwards



An OR gate is a 1-out-of-N scheme

Handling AND gates

Final step: each user gets the keys of the wires going out from its variable

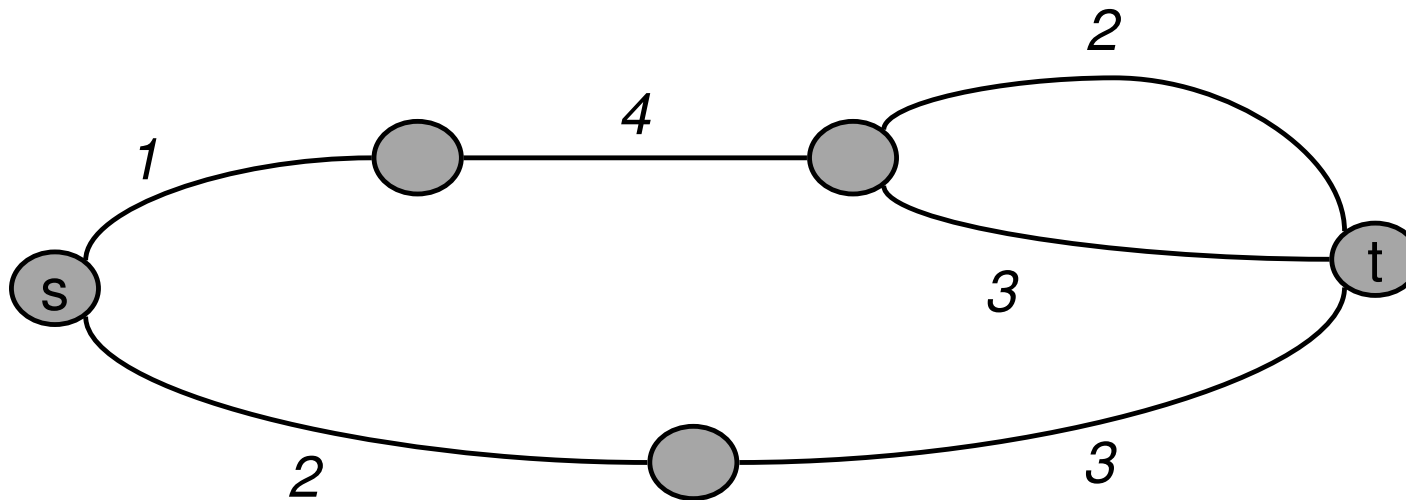


Proof of security:
by induction

An AND gate is an N-out-of-N scheme

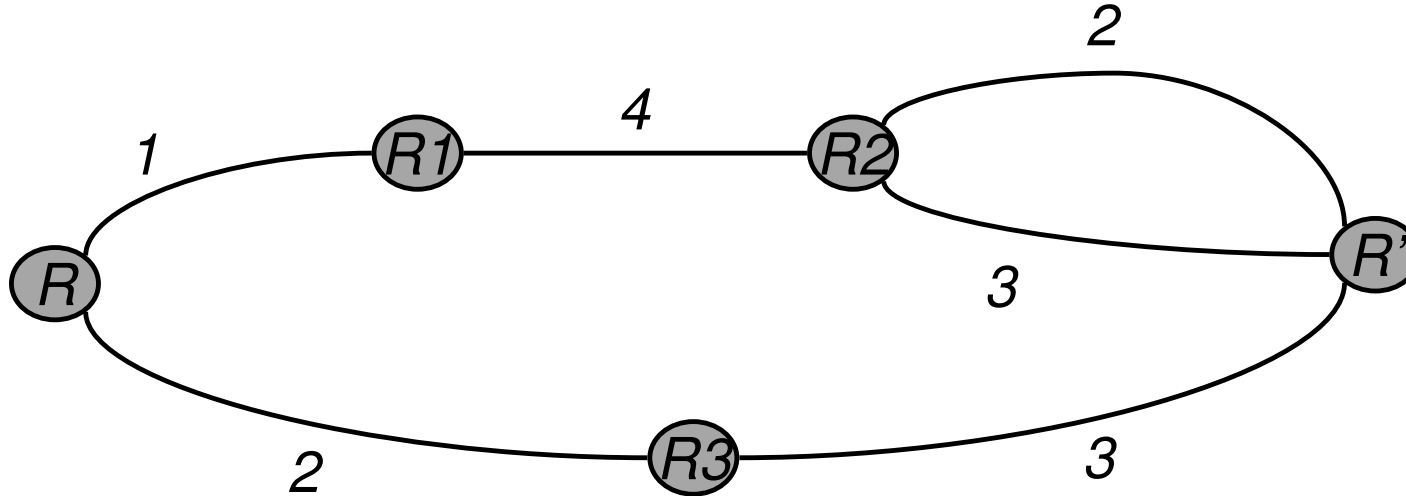
A graph based construction

- Represent the access structure by an undirected graph.
- An authorized set corresponds to a path from s to t in an undirected graph.
- $\Gamma_0 = \{ \{1,2,4\}, \{1,3,4\}, \{2,3\} \}$

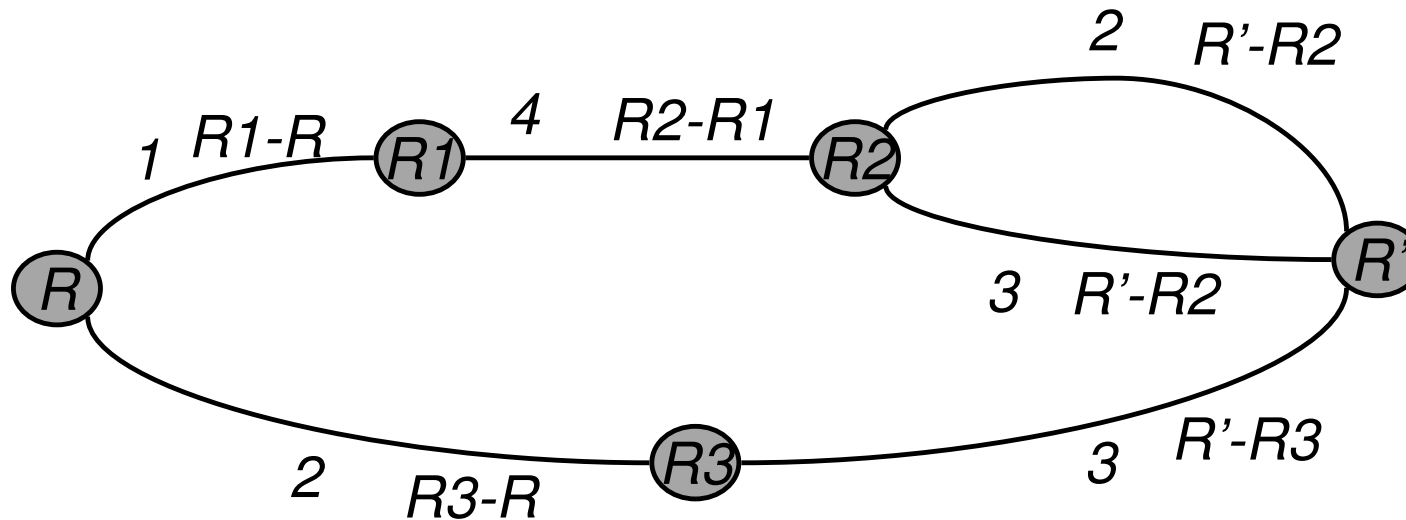


A graph based construction

Assign random values to nodes, s.t. $R' - R = \text{shared secret}$
($R' = R + \text{shared secret}$)



A graph based construction



- Assign to edge $R1 \rightarrow R2$ the value $R2 - R1$
- Give to each user the values associated with its edges

A graph based construction

- Consider the set $\{1,2,4\}$
- why can an authorized set reconstruct the secret? Why can't a unauthorized set do that?

