

Un modello per la Sicurezza

Introduzione (1)

- Per tutti i protocolli sono prescritte **sequenze di interazioni** tra agenti per raggiungere un determinato obiettivo
- I protocolli di comunicazione hanno l'obiettivo di stabilire una **comunicazione** tra gli agenti
 - stabilire il link
 - accordarsi sulla sintassi
 - ...

Introduzione (2)

- I protocolli di sicurezza hanno lo scopo di stabilire le **opportune sequenze di interazioni** che permettono di offrire determinati **servizi di sicurezza** nella comunicazione tra **agenti distribuiti**
- Richiedono scambio di informazioni tra i nodi comunicanti
 - talvolta è richiesta la presenza di una terza parte **fidata (trusted)**

Introduzione (3)

- Sono adatti per tecniche di analisi rigorosa e formale
- Rappresentano componenti critiche per sistemi distribuiti in cui è desiderato un alto livello di sicurezza
- Sono facilmente descrivibili a parole
- Sono difficili da valutare manualmente
- Sono compatti e facili da manipolare

Caratteristiche di Sicurezza (1)

- **Segretezza:**
 - nessun intruso può dedurre le attività che gli utenti legittimi stanno svolgendo
 - nessun intruso può leggere i messaggi che gli utenti legittimi si stanno scambiando
- **Autenticazione:**
 - dell'origine dei messaggi
 - delle entità coinvolte nella comunicazione

Caratteristiche di Sicurezza (2)

- **Integrità dei dati:**
 - Garanzia che i messaggi ricevuti non sono stati corrotti
 - È un corollario dell'autenticazione precedente
- **Non ripudio:**
 - Non è possibile negare di aver partecipato a una comunicazione

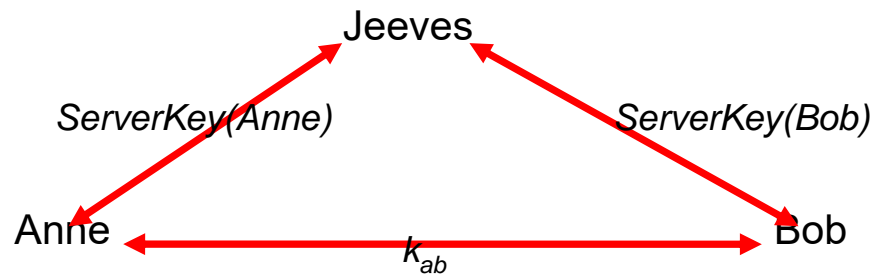
Caratteristiche di Sicurezza (3)

- **Equità (Fairness):**
 - Evitare che un comunicante possa avvantaggiarsi durante la comunicazione rispetto ad un altro
 - Ad es. interrompendo la comunicazione dopo aver ricevuto dati sensibili E prima di aver trasmesso i propri
- **Anonimia:**
 - Poter utilizzare determinati servizi senza dover comunicare la propria identità

Caratteristiche di Sicurezza (4)

- **Disponibilità**
 - Garanzia che il servizio per il quale si è attivata la comunicazione è effettivamente offerto

Schema di Riferimento Classico per Canale Sicuro (1)



Schema di Riferimento Classico per Canale Sicuro (2)

- Protocollo di sicurezza basato su un algoritmo simmetrico di crittografia
- I due agenti Anne (a) e Bob (b) devono comunicare attraverso un canale sicuro con l'aiuto di un server fidato (Jeeves)
- In generale, tutti i gli agenti registrati condividono con Jeeves chiavi di crittografia private e a lunga vita

Schema di Riferimento Classico per Canale Sicuro (3)

- Queste chiavi permettono ad ogni agente di comunicare in modo sicuro con Jeeves
- Non permettono la comunicazione tra Anne e Bob

Possibili comunicazioni sicure(1)

- Una soluzione che permetta a due agenti di comunicare tra di loro è data da:
 - Anne manda a Jeeves il msg destinato a Bob, codificato con il codice ServerKey(Anne)
 - Jeeves decifra il msg da Anne, lo ricodifica secondo ServerKey(Bob)
 - Jeeves invia il ms a Bob
- La comunicazione è sicura ma è troppo onerosa: Jeeves è un collo di bottiglia

Possibili comunicazioni sicure(2)

- Soluzione: Jeeves fornisce chiavi a lunga vita per ogni possibile coppia di agenti comunicanti
 - se il numero N di agenti è elevato servono N^2 chiavi
 - in genere la dimensione della rete potrebbe essere molto dinamica

Possibili comunicazioni sicure(3)

- Soluzione: richiedere l'intervento di Jeeves solo quando necessario per fornire una chiave che permetta la comunicazione sicura tra Anne e Bob

Un Protocollo Classico (1)

Msg1 $a \rightarrow J : a.b.n_a$
Msg2 $J \rightarrow a : \{n_a.b.k_{ab}.\{k_{ab}.a\}_{ServerKey(b)}\}_{ServerKey(a)}$
Msg3 $a \rightarrow b : \{k_{ab}.a\}_{ServerKey(b)}$
Msg4 $b \rightarrow a : \{n_b\}_{k_{ab}}$
Msg5 $a \rightarrow b : \{n_b-1\}_{k_{ab}}$

Un Protocollo Classico (2)

- Nella forma
 $MsgN \ x \rightarrow y : data$
 - N indica lo step del protocollo
 - x l'agente mittente del msg
 - y il ricevente
 - $data$ il contenuto in chiaro
 - $\{data\}_k$ il contenuto criptato con la chiave k
- Il contenuto dei messaggi è costituito dalla concatenazione di varie parti
 - Il simbolo $.$ indica la concatenazione

Protocollo: step1

- Nel msg1 a comunica a J di voler comunicare con b (a.b)
 - n_a è un **nonce**, cioè un messaggio fittizio, creato da a
 - È un messaggio “in chiaro”
- Nel msg2 J
 - Crea la chiave di codifica k_{ab} che dovrà essere usata per le comunicazioni tra a e b
 - fornisce ad a la chiave

Protocollo: step2

- J crea la chiave di codifica k_{ab} che si dovrà usare per le comunicazioni tra a e b
- Il messaggio spedito da J è criptato secondo la chiave stabilita per le comunicazione tra a e J ($\{ \}$ $_{ServerKey(a)}$) e contiene:
 - Il nonce inviato da a allo step 1 e il nome di b, allo scopo di dare conferma alla richiesta del msg1
 - La chiave k_{ab}
 - Un msg ($\{k_{ab} \cdot a\}_{ServerKey(b)}$) che a **NON** è in grado di capire in quanto criptato secondo la chiave stabilita per le comunicazione tra b e J, ma che a dovrà girare a b

Protocollo: step3

- a gira a b la parte del messaggio ricevuto da J, che non è stata decriptata ($\{k_{ab} \cdot a\}_{ServerKey(b)}$)
- Tale messaggio ha lo scopo di segnalare a b che la comunicazione susseguente con a sarà sicura e autenticata da J
- b riceve il messaggio e decriptandolo con la chiave $ServerKey(b)$ è in grado di conoscere la chiave k_{ab}

Protocollo: step4

- b crea un nonce e lo spedisce ad a codificato con la chiave k_{ab} allo scopo di indicare la disponibilità ad iniziare la comunicazione

Protocollo: step5

- a
 - estrae dal messaggio ricevuto da b il nonce
 - lo modifica secondo un criterio standard (tipicamente sottraendo 1)
 - invia a b il valore del nonce modificato
- Quando riceve il messaggio verifica che il valore del nonce ricevuto sia effettivamente quanto si aspetta e in tal caso la comunicazione criptata secondo k_{ab} può avere inizio

Protocollo Yahalom

Msg1 $a \rightarrow b$: $a.n_a$
Msg2 $b \rightarrow J$: $b.\{a.n_a.n_b\}_{ServerKey(b)}$
Msg3 $J \rightarrow a$: $\{b.k_{ab}.n_a.n_b\}_{ServerKey(a)}.\{a.k_{ab}\}_{ServerKey(b)}$
Msg4 $a \rightarrow b$: $\{a.k_{ab}\}_{ServerKey(b)}.\{n_b\}_{k_{ab}}$