

Crittografia e sicurezza delle reti

Rassegna

- Autenticazione
- Protocolli di autenticazione
- X.509, Kerberos

Modello per Autenticazione

Alice vuole provare la sua identità a Bob per ottenere un servizio, avere accesso ad una risorsa ecc.

Bob può chiedere:

- Chi sei? (Dimostra che sei Alice)

Trudy cerca di impersonare Alice:

- Una sola volta
- Sempre

Scenari di autenticazione

Autenticazione

- utenti che condividono una password fra loro (symm. key)
- utenti che condividono una password con autorità fidata (authentication servers)
- utenti che hanno una chiave pubblica

Autenticazione - Symm.Key

K chiave comune fra A e B

- Autent. unilaterale con timestamps
 1. A invia a B: $K(t_A, B)$
- Autent. unilaterale con nonce
 1. B invia a A N
 2. A invia a B $K(N, B)$
- Mutua autent. con nonce
 1. B invia a A N
 2. A invia a B $K(N, N', B)$
 3. B invia a A $K(N, N')$

Server di Autenticazione

Intermediari fidati

Non è possibile autenticare a due a due fra tutti gli utenti (esplosione combinatoriale)

Server di autenticazione Key distribution center (KDC)

- Singolo punto di errore, prestazioni

Autorità di certificazione (CA)

- può essere off-line
- Singolo punto di errore
- Necessità di gestire lista revocche (CRL)

Server di Autenticazione

Protocolli per sistemi distribuiti

- Autenticazione e individuazione di chiavi di sessione in un sistema distribuito
- A e B condividono ciascuno una chiave (K_{AC} e K_{BC}) con C che è autorità fidata
- A e B non sono necessariamente utenti umani ma, in genere, entità del sistema
 - autenticazione di A
 - opzionale: B si autentica
 - opzionale: stabilire chiave di sessione

Server di Autenticazione

Trudy è un utente legittimo del sistema che può fare:

- intercettazione, sniffing e spoofing di messaggi
- può partecipare in una o più sessioni con A e B
 - eseguire le diverse esecuzioni in modo interleaved
 - indurre A e/o B a iniziare sessioni con T
- può conoscere vecchie chiavi di sessione (ma non conosce chiavi long term)

Trudy non

- può indovinare numeri casuali
- non conosce le chiavi pubbliche o private di altri utenti
- non è in grado di decodificare messaggi codificati con chiavi non note

Crittografia vs. autenticità

Gli attacchi si basano sul fatto che uno o più messaggi inviati non sono autentici (non sono stati inviati durante l'esecuzione del protocollo sotto attacco)

- o sono vecchi messaggi o sono falsi
- garantire l'autenticità del mittente e **l'integrità** dei messaggi

La codifica crittog. di un messaggio non ne garantisce l'autenticità

Autenticazione: tecniche

- uso di nonce
- uso di timestamp
- esplicitare destinatario del messaggio

grande cautela nella valutazione di un protocollo;

Autenticaz. con autorità fidata C

A e B condividono ciascuno una chiave (K_{AC} e K_{BC}) con C che è autorità fidata

1. A manda a C A,B
2. C genera K a caso, e manda a A $K_{AC}(K)$ e $K_{BC}(K)$
3. A calcola K e manda a B C, A, $K_{BC}(K)$
4. B decodif. $K_{BC}(K)$ e trova K e manda a A K(Hello A, sono B)

Attacco

Problema: B non è certo di ricevere il messaggio originato da C

1. A manda a T (e non a C) : A,B
2. T manda a C A,T
3. C genera K e manda a A $KAC(K)$ e $KTC(K)$
4. A invia a B C,A, $KTC(K)$ - T intercetta
5. T invia a A K(Hello A, sono B)

Correz: passo 1: A manda a C $\langle A, KAC(B) \rangle$ (C ha garanzia che il messaggio sia di A)

Questa e altre correzioni non funzionano

Protocollo Needham-Schroeder

Challenge-Response :

1. A genera N (nonce) e invia a C A, B, N
2. C genera K e manda a A $KAC(N, K, B, KBC(K, A))$
3. A decodifica, verifica N e B ident. e manda a B $KBC(K, A)$
4. B decodifica, verifica id. A crea nonce N' e invia a A $K(\text{sono } B, N')$
5. A invia a B $K(\text{sono } A, N'-1)$

Attacco Protocollo NS

1. A genera N (nonce) e invia a $C \langle A, B, N \rangle$
2. C genera K e manda a A $KAC(N, K, B, KBC(K, A))$
T si sostituisce a A
3. A decodifica, verifica N e B ident. e manda a B $KBC(K, A)$ - T intercetta e (come A) invia a B $KBC(K', A)$
4. B decodifica, verifica id. A crea nonce N' e invia a T (e non a A) K' (sono B, N')
5. T invia a B K (sono A, $N'-1$)

Nota: T usa vecchie chiavi di sessione e si finge A
A è sicura che C sia 'attivo' mentre B non lo sa

Autenticazione autor. fidata

Protocollo NS - timestamp

Challenge-Response :

1. A genera N (nonce) e invia a C A, B, N
2. C genera K e manda a A $KAC(N, K, B, \dagger, KBC(K, A, \dagger))$
3. A decodifica, verifica N e B ident. e manda a B $KBC(K, A, \dagger)$
4. B decodifica, verifica id. A crea nonce N' e invia a A $K(\text{sono } B, N')$
5. A invia a B $K(\text{sono } A, N'-1)$

Necessità di clock sincroni tra A e B

Autenticazione con Public key

Protocollo Needham-Schroeder

1. A invia a C $\langle A, B \rangle$
2. C a A : $K_{SC}(K_{PB}, B)$
3. A verifica firma C crea nonce N e manda a B $K_{PB}(N, A)$
4. B decodifica e verifica A id. e manda a C $\langle A, B \rangle$
5. C invia a B $K_{SC}(K_{PA}, A)$
6. B verifica firma C crea nonce N' e manda a A $K_{PA}(N, N')$
7. A decodifica verifica e manda a B $K_{PB}(N')$

Prot. N-S public key - Attacco

- Trudy è utente del sistema che può effettuare autenticazione con A, B e C (C autorità certificazione)
- Due esecuzioni del protocollo interleaved: R1 A e T si autenticano; in R2 T si autentica come A con B, Man in the middle attack
- T è in grado di indurre A a iniziare sessione con T
- I passi 1,2,4,5 permettono di ottenere chiavi pubbliche
- I passi 3,6,7 eseguono l'autenticazione

Prot. N-S public key - Attacco

Si considerano solo passi 3,6,7 di R1 e R2:

1. A-->T : passo 3 di R1 invio di $KPT(N,A)$
2. T-->B: passo 3 di R2 invio di $KPB(N,A)$
3. B-->T: passo 6 di R2 invio di $KPA(N',N)$
4. T-->A: passo 6 di R1 invio di $KPA(N',N)$
5. A-->T: passo 7 di R1 invio di $KPT(N')$
6. T-->B: passo 7 di R2 invio di $KPB(N')$

B crede di parlare con A e condividere con A i nonce segreti

Protocollo Needham-Schroeder con public key - correzz.

1. A invia a C $\langle A, B \rangle$
2. C a A : $KSC(KPB, B)$
3. A verifica firma C crea nonce N e manda a B $KPB(N, A)$
4. B decodifica e verifica A id. e manda a C $\langle A, B \rangle$
5. C invia a B $KSC(KPA, A)$
6. B verifica firma C crea nonce N' e manda a A $KPA(\text{Bob}, N, N')$
7. A decodifica verifica e manda a B $KPB(N')$

Challenge-response Pub.Key

Sig_A è firma di A, cert_A è certificato di A

- **Autent. unilaterale con timestamps**
 1. A invia a B: certA,tA,B,Sig_A(tA,B)
- **Autent. unilaterale con nonce**
 1. B invia a A N
 2. A invia a B certA,N',B,Sig_A(N,N',B)
(uso di N' esclude attacchi tipo chosen-text)
- **Mutua autent. con nonce**
 1. B invia a A N
 2. A invia a B certA,N',B,Sig_A(N,N',B)
 3. B invia a A certB,A,Sig_B(N,N',A)

Nota: si firmano i nonce

X.509 Servizio Autenticazione

Parte degli standard del direttorio CCITT X.500

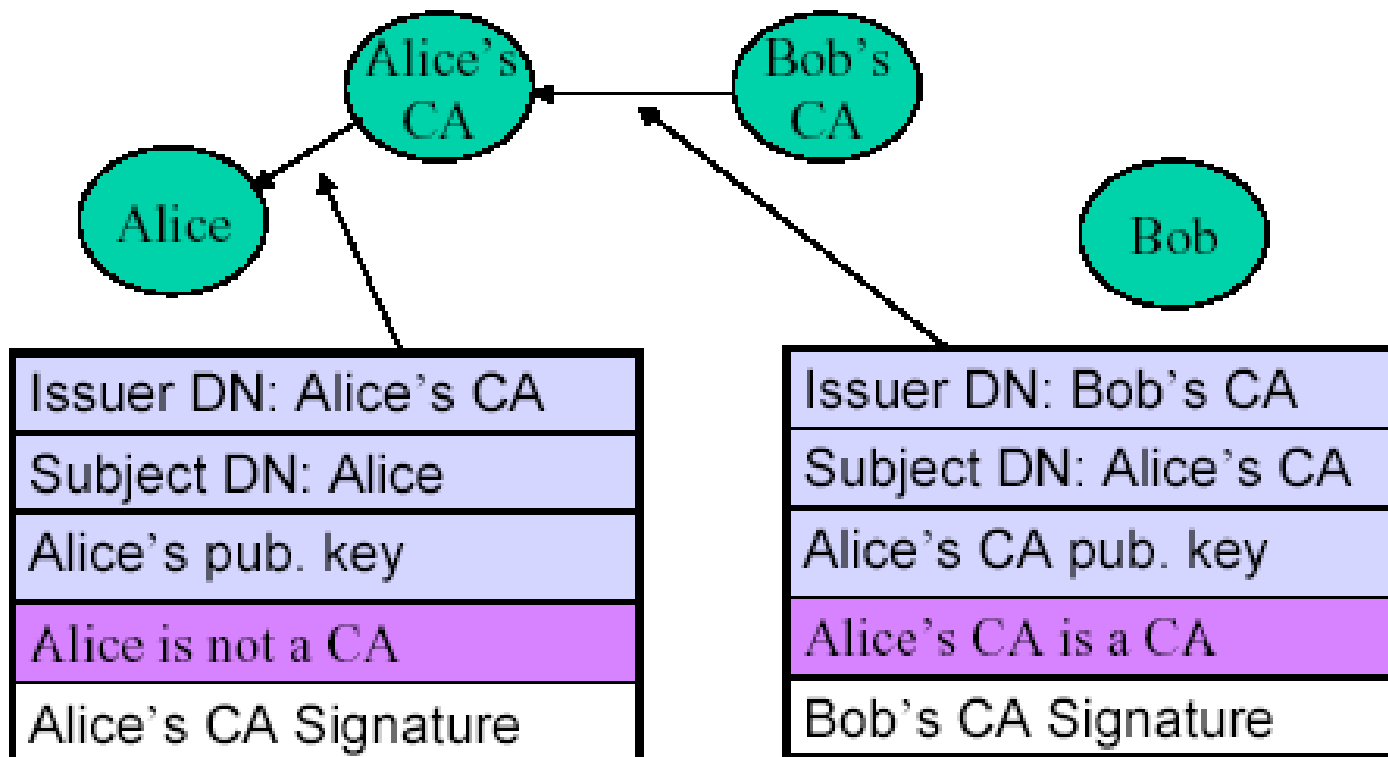
- Servizi di autenticazione
 - direttorio (Autor. certificazione) memorizza certificati di chiave pubblica con la chiave pubblica dell'utente firmata dall'autorità di certificazione
 - lista certificati revocati
- Definisce protocolli di autenticazione
- Usa crittografia chiave pubblica e firma digitale
 - algoritmi non sono nello standard ma si raccomanda RSA

Certificati X.509

Signed fields	Version		
	Certificate serial number		
	Signature Algorithm Object Identifier (OID)		
	Issuer Distinguished Name (DN)		
	Validity period		
	Subject (user) Distinguished Name (DN)		
	Subject public key information	Public key Value	Algorithm Obj. ID (OID)
	Issuer unique identifier (from version 2)		
	Subject unique identifier (from version 2)		
	Extensions (from version 3)		
	Signature on the above fields		

Gerarchia CA

Come fa Bob ad avere il certificato di Alice
(se non fanno parte della stessa CA?)



Procedure di Autenticazione

- X.509 include tre procedure di autenticazione con chiave pubblica:
- One-Way Authentication
- Two-Way Authentication
- Three-Way Authentication

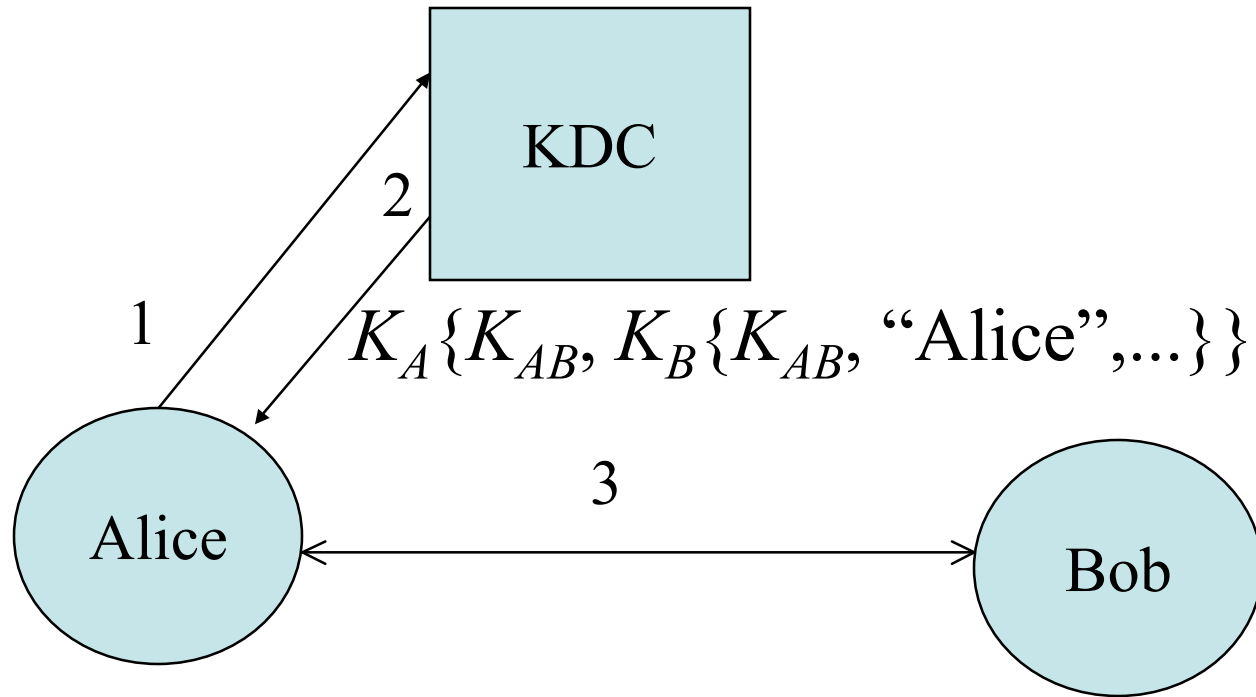
Si usano anche Nonce, Timestamp e Firma digitale

Protocollo Kerberos

Progetto Athena del MIT

- Scenario: A vuole accedere i servizi di B
 - autenticazione di A
 - opzionale: B si autentica
 - opzionale: stabilire chiave di sessione
- C è autorità fidata e condivide chiavi con A e B
- Idea: uso biglietti di durata prefissata (da C)

Biglietti Alice, Bob e KDC



Ticket di B = $KBC(K,A,L)$ L validità ticket
Uso di indirizzo rete del mittente

Chiavi di Sessione e TGT...

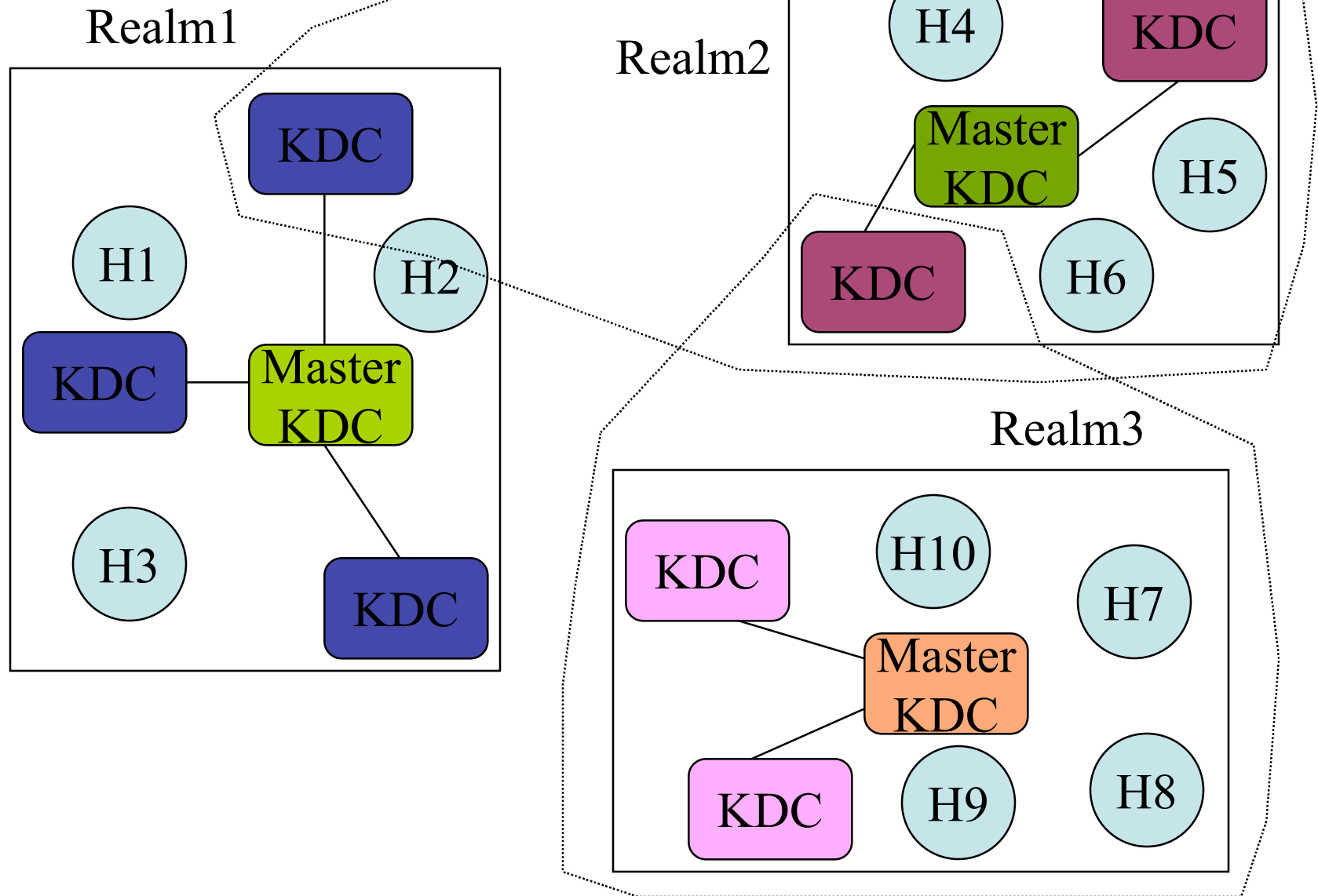
Per ridurre uso della password e/o della master key

- Al tempo del login iniziale una chiave di sessione per il principal S_B (per Bob) viene richiesta al KDC
- S_B ha un periodo di validità ridotto
- Un TGT per Bob viene anche fornito dal KDC, che include la chiave di sessione S_B e le informazioni che identificano Bob (codificando con la chiave master)

Kerberos - Prestazioni

- KDC devono mantenere solo TGT e tickets.
- Il carico maggiore è sui client.
 - Convincimi che ho fatto questo per te...
- KDC è coinvolto solo all'inizio nella fornitura del ticket
- KDC usa solo informazioni statiche (possibilità di repliche dei KDC).

Kerberos V. 4: Modello



Esercizio : autenticazione

Il seguente protocollo autentica A con B e utilizza un'autorità di certificazione (C) che possiede certificati delle chiavi pubbliche

1. A invia a C : A,B
2. C invia a A: KPA, KPB (invia i certif. chiavi pubb. A e B)
3. A invia a B: KPA, KPB, KPB(Sig_KSA(K,timestamp))

Nota nel passo 3 A invia codificato con la chiave pubblica di B il mess. firmato da A contenente un timestamp e la chiave di sessione proposta. (ovviamente i certificati sono verificabili da parte di B e se non sono validi il protocollo non autentica A).

Il protocollo è facilmente attaccabile.

1. Mostrare come B possa utilizzare l'informazione ottenuta in una sessione di autenticazione per sostituirsi a A verso un terzo utente (diciamo Dario)
2. Mostrare una semplice modifica del passo 3 che rende il protocollo sicuro.

Esercizio : autenticazione/2

1. A invia a C : A,B
2. C invia a A: KPA, KPB (invia i certif. chiavi pubbliche di A e B)
3. A invia a B: KPA, KPB, KPB(Sig_KSA(K,timestamp))

Attacco: B invia a D il messaggio ricevuto da A, codificando la firma di A con la chiave pubblica di C

Soluzione: inserire identità di A e B nel messaggio firmato

3. A invia a B: KPA, KPB, KPB(Sig_KSA(A,B,K,timestamp))

Autenticazione autor. fidata

Esercizio: attacco prot. Woo-Lam

Mostrare che il seguente protocollo non funziona
(sugg. T inizia due sessioni con B - alla fine si autentica con B come A)

1. A invia a B A
2. B invia a A N
3. A invia B $KAC(N)$
4. B invia a C $KBC(A, KAC(N))$
5. C invia a B: $KBC(N)$
6. Bob verifica se il nonce è quello generato da lui

Soluzione: Attacco prot. Woo-Lam

T inizia due sessioni con B: una come A e una come T

1. T(come A) invia a B A
2. T invia a B T
3. B invia a T (come A) N
4. B invia a T N'

T deve indurre C a comunicare a B KBC(N)

Soluzione: Attacco prot. Woo-Lam

T inizia due sessioni con B: una come A e una come T

1. T (come A) invia a B A
2. T invia a B T
3. B invia a T (come A) N
4. B invia a T N'
5. T (come A) invia a B $KTC(N)$!!
6. T invia a B $KTC(N)$!!!
7. B invia a C $KBC(A, KTC(N))$
8. B invia a C $KBC(T, KTC(N))$
9. C invia a B: $KBC(\text{immondizia})$ [messaggio senza senso]
10. C invia a B: $KBC(N)$
11. Bob verifica se il nonce è quello generato da lui SI!
12. Bob verifica se il nonce è quello generato da lui NO!!

Soluzione: Attacco prot. Woo-Lam

T inizia due sessioni con B: una come A e una come T

1. T (come A) invia a B A

2. T invia a B T

3. B invia a T (come A) N

4. B invia a T N'

5. T (come A) invia a B $KTC(N)$!!

6. T come A invia B $KTC(N)$!!!

7. B invia a C $KBC(A, KTC(N))$

8. B invia a C $KBC(T, KTC(N))$

9. C invia a B: $KBC(\text{immondizia})$ [messaggio senza senso]

10. C invia a B: $KBC(A, N)$

11. Bob verifica se il nonce è quello generato da lui PER A NO!

N.B.: questa modifica è comunque attaccabile in altro modo
(sugg. 1 sessione in cui T si finge A e si sostituisce a C)

Soluzione: includere identità
nel mess. 5 e verifica a 6

5. C invia a B $KBC(A, N)$

6. B verifica nonce e ident.
quando decodifica

Attacco prot. Woo-Lam modificato

T inizia una sessione con B in cui sostituisce A e C

1. T(come A) invia a B A
2. B invia a T (come A) N
3. T (come A) invia a B N (assume che $KAC(N) = N$!!)
4. B invia a T (come C) $KBC(A, B, N)$
5. T (come C) invia a B: $KBC(A, B, N)$
6. Bob verifica se il nonce è quello generato da lui **SI!**

Attacco detto reflection attack: T risponde con lo stesso messaggio (Nota che T deve alterare il messaggio originario)

Attacco prot. Woo-Lam modificato

T inizia una sessione con B in cui sostituisce A e C

1. T invia a B C
2. A invia a T N
3. T invia a A $KTC(N)$
4. A invia a T (come C) $KAC(C, KTC(N))$

5. A invia a T (come C) $KAC(C, KTC(N))$
6. Bob verifica se il nonce è quello generato da lui **SI!**

Esercizio IPSEC

Considerate una società che vuole installare un server di posta e un web server che sono accessibili ambedue da Internet. La società vuole fornire ai suoi dipendenti un accesso remoto ai loro computer in modo tale da permettere loro un accesso da casa. Si considerino le seguenti tecnologie e discutere brevemente quale/quali sono più appropriate:

- IPSEC in modalità tunnel tra il firewall della società e i calcolatori di casa degli impiegati
- IPSEC in modalità trasporto tra i calcolatori di casa degli impiegati e le apparecchiature che vogliono accedere da casa
- SSL
- PGP

Esercizio applicazione

IMS (Internet Music Station) distribuisce concerti dal vivo ai suoi membri in regola con il pagamento della quota associativa. I non membri o i membri che non hanno rinnovato la quota non devono poter ascoltare. IMS ha trovato uno schema che funziona con al più n clienti (sia attuali che utenti che abbiano disdetto). Per fare questo IMS distribuisce ai suoi utenti in regola n chiavi R_1, R_2, \dots, R_n .

Mostrare uno schema con cui IMS raggiunge il suo scopo (ogni utente in regola può ascoltare uno non in regola non può).

Nota che quando un utente non paga rimane comunque in possesso delle chiavi.

Esercizio Applicazione/2

L'utente i riceve tutte le chiavi tranne R_i . Sia S l'insieme degli utenti con chiavi valide.

1. Mostra che IMS può costruire una chiave K che permette solo a utenti in S di ricostruire K , mentre un utente non in S non riesce a ricostruire K . Si può assumere che S sia noto a tutti (perché, ad esempio, è trasmesso in chiaro all'inizio).

Sugg. Uso di one-way hash function

2. La soluzione proposta è resistente alle collisioni? Cioè due utenti che non fanno parte di S possono essere in grado di ricostruire K ?