SSL (cenni)

M. Danelutto
Lab. di programmazione di rete - Corso B
A.A. 07-08

Problema con socket standard

- Dati inviati viaggiano in chiaro
  - intercettabili (basta uno sniffer)
- Non è garantita autenticazione
  - ci si può spacciare per chi non siamo
- Non è garantita integrità
  - errori o sostituzione di dati non rilevati

SSL: secure socket layer

- Supporta sicurezza, autenticazione e integrità
  - prima era un pacchetto separato (JSSE fino all 1.4) per ragioni "strategiche"
  - adesso è parte integrante di SDK
    - package javax.net.ssl

"Protocollo" SSL

- utilizzo chiavi pubbliche/private per scambio di una chiave di sessione (simmetrica)
  - occorre chiave pubblica certificata
- checksum/digest
- algoritmi configurabili
  - handshake iniziale per agreement su

SSL: costo

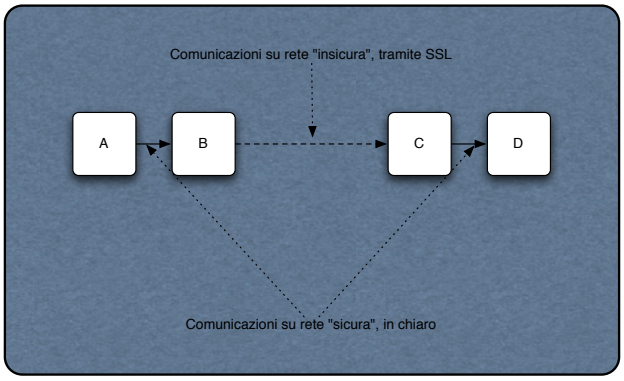
- Fase di inizializzazione costosa
  - generazione e scambio della chiave simmetrica
- A regime
  - cifratura e decifratura di tutti i pacchetti con aggiunta del checksum/digest

SSL: costo (2)

- Costo di inizializzazione
  - once and forall : trascurabile
- Costo di trasmissione
  - pagato continuamente
    - come CPU (salvo coprocessore o proxy)

Proxy pattern

- per eliminare il costo della cifratura



Utilizzo dei socket SSL

- Lato client
  - utilizzo socket a da javax.net.ssl
- SSLSocketFactory sslsocketfactory = (SSLSocketFactory) SSLSocketFactory.getDefault();
- SSLSocket sslsocket = (SSLSocket) sslsocketfactory.createSocket(hostName,

Utilizzo dei socket SSL (2)

- lato server
- SSLServerSocketFactory sslserversocketfactory = (SSLServerSocketFactory) SSLServerSocketFactory.getDefault();
  SSLServerSocket sslserversocket = (SSLServerSocket) sslserversocketfactory.createServerSocket(PORTA);
  while(true) {
  SSLSocket sslsocket = (SSLSocket) sslserversocket.accept();
  ...

Successivamente ...

- lato client e lato server utilizzano i metodi tipici di Socket e ServerSocket sugli SSL socket
- MA:
  - occorre da riga di comando o da programma configurare l'ambiente SSL

Da riga di comando (1)

- generazione di un keystore (con coppie di chiavi per il server)
- keytool -genkey -keystore ks1 -keyalg DSA
  - chiede dati (e passwd) e genera coppia di chiavi/certificato

```
[marco-daneluttos-macbook:~/Documents/workspace/LPRb0708] marcod% keytool -genkey -
keystore ks1 -keyalg DSA
Enter keystore password:  pippopluto
What is your first and last name?
  [Unknown]:  Marco Danelutto
What is the name of your organizational unit?
  [Unknown]:  Dip. Informatica
What is the name of your organization?
  [Unknown]:  Univ. Pisa
What is the name of your City or Locality?
  [Unknown]:  Pisa
What is the name of your State or Province?
  [Unknown]:  PI
What is the two-letter country code for this unit?
  [Unknown]:  IT
Is CN=Marco Danelutto, OU=Dip. Informatica, O=Univ. Pisa, L=Pisa, ST=PI, C=IT correct?
  [no]:  yes

Enter key password for <mykey>
    (RETURN if same as keystore password):
[marco-daneluttos-macbook:~/Documents/workspace/LPRb0708] marcod%
```

Da riga di comando (2)

- lancio del server con comando:
- java -Djavax.net.ssl.keyStore=ks1 -Djavax.net.ssl.keyStorePassword=pippoplut o ssl.SimpleServer
  - si passa il keystore e la passwd per accederlo da riga di comando (ATTENZIONE! si può vedere con un

Da riga di comando (3)

- si copia il keystore dove viene lanciato il client
- si lancia il client con il comando:
- java -Djavax.net.ssl.trustStore=ks1 -Djavax.net.ssl.trustStorePassword=pippoplu to ssl.SimpleClient

Da riga di comando (4)

- se vogliamo “vedere” cosa succede:
- -Djava.protocol.handler.pkgs=
  com.sun.net.ssl.internal.www.protocol
  -Djavax.net.debug=ssl

Versione senza trace

CLIENT

```
[marco-daneluttos-macbook:~/Documents/workspace/LPRb0708] marcod% java -
Djavax.net.ssl.trustStore=ks1 -Djavax.net.ssl.trustStorePassword=pippopluto
ssl.SimpleClient    Ciao come vaBene
passo e chiudo
[marco-daneluttos-macbook:~/Documents/workspace/LPRb0708] marcod%
```

SERVER

```
[marco-daneluttos-macbook:~/Documents/workspace/LPRb0708] marcod% java -
Djavax.net.ssl.keyStore=ks1 -Djavax.net.ssl.keyStorePassword=pippopluto
ssl.SimpleServer
Ciao come va
Bene
passo e chiudo
^C[marco-daneluttos-macbook:~/Documents/workspace/LPRb0708] marcod%
```

Versione con trace

CLIENT

```
[marco-daneluttos-macbook:~/Documents/workspace/LPRb0708] marcod% java -
Djavax.net.ssl.trustStore=ks1 -Djavax.net.ssl.trustStorePassword=pippopluto -
Djava.protocol.handler.pkgs=com.sun.net.ssl.internal.www.protocol -
Djavax.net.debug=ssl ssl.SimpleClient
setting up default SSLSocketFactory
use default SunJSSE impl class: com.sun.net.ssl.internal.ssl.SSLSocketFactoryImpl
class com.sun.net.ssl.internal.ssl.SSLSocketFactoryImpl is loaded
keyStore is :
keyStore type is : jks
keyStore provider is :
init keystore
init keymanager of type SunX509
trustStore is: ks1
trustStore type is : jks
trustStore provider is :
init truststore
adding as trusted cert:
  Subject: CN=Marco Danelutto, OU=Dip. Informatica, O=Univ. Pisa, L=Pisa, ST=PI, C=IT
  Issuer:  CN=Marco Danelutto, OU=Dip. Informatica, O=Univ. Pisa, L=Pisa, ST=PI, C=IT
  Algorithm: DSA; Serial number: 0x475f04b4
  Valid from Tue Dec 11 22:44:20 CET 2007 until Mon Mar 10 22:44:20 CET 2008

init context
trigger seeding of SecureRandom
done seeding SecureRandom
instantiated an instance of class com.sun.net.ssl.internal.ssl.SSLSocketFactoryImpl
```

Versione + corretta (keystore non

- Sul server si generano le chiavi
  - keytool -genkey -keystore kss -keyalg DSA
- poi si esporta un certificato
  - keytool -export -keystore kss -file certif.cer
- il certificato si importa sul keystore del client
  - keytool -import -file certif.cer -keystore ksc

Gestione parametri “a programma”

SERVER

```
SSLContext context = SSLContext.getInstance("SSL");
KeyManagerFactory kmf = KeyManagerFactory.getInstance("SunX509");
KeyStore ks = KeyStore.getInstance("JKS");
char [] passwd = "pippopluto".toCharArray();
ks.load(new FileInputStream("ks1"), passwd);
kmf.init(ks,passwd);
context.init(kmf.getKeyManagers(), null, null);

SSLServerSocketFactory factory = context.getServerSocketFactory();
ServerSocket ss = null;
try {
    ss = factory.createServerSocket(PORT);
} catch (IOException e) {
    e.printStackTrace();
}
SSLServerSocket sslSs = (SSLServerSocket) ss;
```

Tool per vedere pacchetti (solo da root)

- tcpdump
  - front end grafico: ethereal
- permette di copiare su file tutti i pacchetti che passano su una interfaccia
- e di vederne il contenuto

Esempio di comandi

- cattura di una traccia
  - sudo tcpdump -i en0 -s 0 -w PCK.dat
- lettura di una traccia catturata
  - tcpdump -s 0 -n -e -x -vvv -A -r PCK.dat

Esempio di output

```
22:19:46.759764 AF IPv6 (30), length 88: (hlim 64, next-header: TCP (6), length:
 44) ::1.58954 > ::1.23: S, cksum 0xed84 (correct), 334879346:334879346(0) win 6
5535 <mss 16324,nop,wscale 2,nop,nop,timestamp 583866323 0,sackOK,eol>
        0x0000:  6000 0000 002c 0640 0000 0000 0000 0000
        0x0010:  0000 0000 0000 0001 0000 0000 0000 0000
        0x0020:  0000 0000 0000 0001 e64a 0017 13f5 da72
        0x0030:  0000 0000 b002 ffff ed84 0000 0204 3fc4
        0x0040:  0103 0302 0101 080a 22cd 17d3 0000 0000
        0x0050:  0402 0000
22:19:46.759791 AF IPv6 (30), length 64: (hlim 64, next-header: TCP (6), length:
 20) ::1.23 > ::1.58954: R, cksum 0xdb04 (correct), 0:0(0) ack 334879347 win 0
        0x0000:  6000 0000 0014 0640 0000 0000 0000 0000
        0x0010:  0000 0000 0000 0001 0000 0000 0000 0000
        0x0020:  0000 0000 0000 0001 0017 e64a 0000 0000
        0x0030:  13f5 da73 5014 0000 db04 0000
22:19:46.773370 AF IPv6 (30), length 88: (hlim 64, next-header: TCP (6), length:
 44) ::1.58955 > ::1.23: S, cksum 0x9a08 (correct), 3052571632:3052571632(0) win
 65535 <mss 16324,nop,wscale 2,nop,nop,timestamp 583866323 0,sackOK,eol>
        0x0000:  6000 0000 002c 0640 0000 0000 0000 0000
        0x0010:  0000 0000 0000 0001 0000 0000 0000 0000
        0x0020:  0000 0000 0000 0001 e64b 0017 b5f2 8bf0
        0x0030:  0000 0000 b002 ffff 9a08 0000 0204 3fc4
        0x0040:  0103 0302 0101 080a 22cd 17d3 0000 0000
```

Comandi server

- keytool -genkey -keystore ks1 -keyalg DSA
- keytool -export -keystore ks1 -file certif.cer
- keytool -printcert -file certif.cer
- keytool -import -file certif.cer -keystore ksc
- java -Djavax.net.ssl.keyStore=ks1 -

Comandi client

- java -Djavax.net.ssl.trustStore=ks1 -Djavax.net.ssl.trustStorePassword=pippoplu to ssl.SimpleClient
- java -Djavax.net.ssl.trustStore=ks1 -Djavax.net.ssl.trustStorePassword=pippoplu to -Djava.protocol.handler.pkgs=com.sun.net.ssl.internal.www.protocol -

Links

- Procedura keystore http://tvilda.stilius.net/java/java_ssl.php
- Link tcpdump http://developer.apple.com/qa/qa2001/qa1176.html