Sicurezza nei Sistemi Informativi

Alcune note su OpenSSL

Ing. Orazio Tomarchio
Orazio.Tomarchio@diit.unict.it

Dipartimento di Ingegneria Informatica e delle Telecomunicazioni
Università di Catania

Cos'è OpenSSL?

- "The OpenSSL Project is a collaborative effort to develop a robust, commercial-grade, full-featured, and **Open Source** toolkit implementing the Secure Socket Layer (SSL v2/v3) and Transport Layer Security (TLS v1) protocols as well as a full-strength general purpose cryptography library."
- "The project is managed by a worldwide community of volunteers that use the Internet to communicate, plan, and develop the OpenSSL toolkit and its related documentation"

www.openssl.org

Cos'è OpenSSL?

- Libreria open source in C
- Implementazione di numerosi algoritmi di crittografia
- Implementazione di funzionalità per gestire certificati digitali X.509
- Implementazione del protocollo SSL/TLS
- È utilizzato all'interno di molti altri applicazioni legate alla sicurezza (es: mod_ssl: modulo SSL per il web server Apache)

Comandi OpenSSL

Standard commands

- asn1parse ca ciphers crl crl2pkcs dgst dh dhparam
  dsa dsaparam ec ecparam enc engine errstr gendh
  gendsa genrsa nseq ocsp passwd pkcs12 pkcs7 pkcs8
  prime rand req rsa rsautl s_client s_server s_time
  sess_id smime speed spkac verify version x509

Message Digest commands (see the `dgst' command for more details)

- md2 md4 md5 rmd160 sha sha1

Cipher commands (see the `enc' command for more details)

- aes-128-cbc aes-128-ecb aes-192-cbc aes-192-ecb aes-256-cbc aes-256-ecb
  base64 bf bf-cbc bf-cfb bf-ecb bf-ofb cast cast-cbc
  cast5-cbc cast5-cfb cast5-ecb cast5-ofb des des-cbc des-cfb des-
  ecb des-ede des-ede-cbc des-ede-cfb des-ede-ofb des-ede3 des-ede3-cbc
  des-ede3-cfb des-ede3-ofb des-ofb des3 desx idea idea-cbc
  idea-cfb idea-ecb idea-ofb rc2 rc2-40-cbc rc2-64-cbc rc2-cbc
  rc2-cfb rc2-ecb rc2-ofb rc4 rc4-40

OpenSSL: certificati self-signed

openssl **req** -config openssl.cnf -newkey rsa:512 -days 1000 –nodes -keyout cakey.pem -out cacert.pem -x509 -new

- req indica la richiesta di un nuovo certificato
- x509 indica che il certificato deve essere self-signed
- config indica il file con le configurazioni da usare per default
- newkey specifica il formato della chiave
- days indica la durata di validità (in giorni)
- nodes indica che la chiave privata sia salvata in chiaro
- keyout indica il nome del file con la chiave privata
- out indica il nome del file col certificato

OpenSSL: richiesta di certificati

openssl **req** -new -newkey rsa:512 –nodes -keyout key.pem –out req.pem -config openssl.cnf

- new indica che è una nuova richiesta di certificato
- config indica il file con le configurazioni da usare per default
- newkey specifica il formato della chiave
- nodes indica che la chiave privata sia salvata in chiaro
- keyout indica il nome del file con la chiave privata
- out indica il nome del file col certificato

OpenSSL: rilascio di certificati

openssl **ca** -policy policy_anything -out cert.pem -config openssl.cnf –infiles req.pem

- config indica il file con le configurazioni da usare per default
- policy indica le politiche da utilizzare per il rilascio
- infiles indica il nome del file con la richiesta
- out indica il nome del file col certificato
- ca è l'opzione per la firma di un certificato

OpenSSL: il comando x509

- L'utility x509 di OpenSSL gestisce i certificati digitali
  - Permette la conversione tra formati di certificati
  - Consente la visualizzazione delle informazioni contenute in un certificato
  - Permette di conoscere l'hash di un certificato da utilizzare per referenziarlo come certificato di una Certification Authority in una directory

OpenSSL: il comando x509

- Ecco le principali opzioni dell'utility:
  - -in indica il file di input col certificato
  - -out indica il file di output col certificato
  - -inform indica il formato di input
  - -outform indica il formato di output
  - -text visualizza le informazioni contenute nel certificato
  - -noout non visualizza il certificato nel suo formato
  - -hash visualizza l'hash del certificato nel formato necessario per usarlo come una CA in una directory

Importare un certificato in un browser Web

- Per importare certificati e chiavi in un browser web viene utilizzato il formato PKCS12
- Il comando pkcs12 consente a partire da un certificato x.509 e dalla corrispondente chiave privata di creare un file in formato PKCS12 (.p12)
- Tale file potrà poi essere importato molto semplicemente all'interno del nostro browser Web
- ATTENZIONE: un file .p12 contiene oltre al proprio certificato personale anche la chiave privata e quindi deve essere adeguatamente protetto

OpenSSL: il comando x509

openssl **pkcs12** -export -chain -CAfile cacert.pem -inkey Key.pem -name Abc –in Cert.pem –out Cert.p12

- Devono essere indicati i file con le informazioni necessarie per ottenere un file in formato PKCS12
- -export genera il file PKCS12
- -chain include la (eventuale) catena dei certificati
- -CAfile indica il certificato della CA che ha firmato il certificato
- -in indica il certificato che si deve convertire nel formato PKCS12
- -inkey indica la chiave privata associata al certificato
- -name indica il nickname che avrà quel certificato quando sarà importato nel browser
- - out indica il nome del file che sarà generato

Importare il certificato

- Sistemi Windows/IE:
  - Il formato .p12 è automaticamente riconosciuto da Windows
  - Un doppio click consente l'importazione guidata nel database dei certificati di Windows
- Firefox/Mozilla
  - Options/Advanced/Security/View Certificate/Import

OpenSSL: le utility s_client e s_server

- Le utility s_server e s_client vengono distribuite con OpenSSL e sono uno dei principali strumenti di debug utilizzati da chi sviluppa applicazioni client/server sicure
- Consentono di simulare un server ed un client SSL, permettendo il setting di vari parametri
- Possono essere eseguite “indipendentemente” l’una dall’altra e sono configurabili con sequenza di argomenti che consentono di scegliere il tipo di connessione SSL desiderata

OpenSSL: s_server

- L’utility s_server è parte del package OpenSSL
- E’ un server SSL utile per il debug di applicazioni client col supporto di SSL
- E’ possibile configurare l’esecuzione di questa utility impostando degli argomenti nella riga di comando
- Prevede ad esempio l’eventuale uso di certificati, autenticazione client, selezione di cipher suite, della versione del protocollo

OpenSSL: parametri di s_server (1/2)

- -accept arg porta TCP/IP del server (default 4433)
- -verify arg richiede l’autenticazione client
- -Verify arg fallisce la connessione se non c’è autenticazione client
- -cert arg indica il file col certificato server (default server.pem)
- -key arg indica il file con la chiave privata (default server.pem)
- -dcert arg eventuale secondo certificato (in generale DSA)
- -dkey arg eventuale seconda chiave (in generale DSA)
- -dhparam arg file con i parametri DH
- -debug vengono visualizzate maggiori informazioni per il debug
- -bugs l’esecuzione tollera alcuni noti bug

OpenSSL: parametri di s_server (2/2)

- -CApath arg directory con i certificati delle CA
- -CAfile arg file con i certificati delle CA
- -cipher arg uso di particolari cipher suite
- -ssl2 uso di SSLv2
- -ssl3 uso di SSLv3
- -tls1 uso di TLSv1
- -no_ssl2 non uso di SSLv2
- -no_ssl3 non uso di SSLv3
- -no_tls1 non uso TLSv1
- -www Risposta a “GET /” con una pagina di prova
- -WWW Risposta a 'GET /<path> HTTP/1.0' con il file ./<path>

OpenSSL: parametri di s_client

- -connect host:port indica il server desiderato (default localhost:4433)
- -verify arg imposta la verifica del certificato del server
- -cert arg indica il certificato da usare
- -key arg indica la chiave da usare
- -CApath arg indica la directory con i certificati delle CA
- -CAfile arg indica il file con i certificati delle CA
- -reconnect interrompe la connessione e la riprende (riesuma)
- -showcerts mostra i certificati ricevuti
- -debug visualizza maggiori informazioni
- -ssl2/ssl3/tls1 imposta un solo protocollo
- -no_tls1/-no_ssl3/-no_ssl2 disabilita qualche protocollo
- -bugs Imposta l'uso di soluzioni ai bug comuni
- -cipher specifica le chiper suite

SSL in Apache: mod_ssl

- **mod_ssl** consente l'utilizzo del protocollo SSL/TLS all'interno del server Web Apache
- Sono necessarie le librerie OpenSSL su cui si appoggia per l'uso delle funzionalità crittografiche

www.modssl.org

ModSSL: direttive di configurazione

- SSLPassPhraseDialog
  - Specifica la chiave con cui è cifrata la chiave privata del server
- SSLRandomSeed
  - Consente di impostare una base per la generazione di valori random
- SSLSessionCache
  - Consente di specificare l'uso di una cache
- SSLEngine
  - Imposta l'uso di SSL

ModSSL: direttive di configurazione

- SSLProtocol
  - Indica la versione del protocollo da utilizzare
- SSLCipherSuite
  - Indica le ciphersuite desiderate
- SSLCertificateFile
  - Indica il file col certificato del server
- SSLCertificateKeyFile
  - Indica il file con la chiave privata
- SSLCACertificateFile
  - Indica il file con il certificato della CA

ModSSL: direttive di configurazione

- SSLVerifyClient
  - Imposta la richiesta del certificato del client
- SSLLog
  - Indica il file di log
- SSLOptions
  - Configura alcune opzioni tra cui la possibilità di esportare informazioni ai CGI in variabili di environment
- SSLRequireSSL
  - Nega l'accesso quando non è in uso SSL

Configurazione del server Web

- Port 80
- Listen 80
- Listen 443
- LoadModule ssl_module modules/mod_ssl.so
- AddModule mod_ssl.c
- <Virtualhost 127.0.0.1:443>
  - DocumentRoot C:/... path del filesystem che verrà servito in modalità SSL
  - SSLEngine on
  - SSLCertificateFile servercert.pem
  - SSLCACertificateFile CAacerts.pem
  - SSLCipherSuite RSA
  - SSLLog logs/ssl_engine_log
  - <Location /cgi-bin/>
    - SSLVerifyClient require
    - SSLOptions +StdEnvVars
    - SSLOptions +ExportCertData
  - </Location>
- </VirtualHost>