DATORSÄKERHET

Att definiera datorsäkerhet kan göras på väldigt många sätt. Oftast brukar det avgränsas till tre områden *Konfidentialitet, Integritet* och *Tillgänglighet*.

- KonfidentialitetFörhindra att obehöriga användare kan läsa känslig information.   
- *Integritet*  
Förhindra att känslig data modifieras av en obehörig användare.  
- *Tillgänglighet*Data som tillhör en behörig användare borde vara tillgänglig.

VAD ÄR SSL/TLS?

TCP, *Transmission Control Protocol*, är ett väldigt använt internetprotokoll vars huvuduppgift är skapa pålitlig dataöverförning mellan två system. Protokollet kan upptäcka om ett paket går förlorat under överföringen eller då de anländer i fel ordning. Det kan till och med avfärda kopior. Från ett säkerhetsperspektiv så är TCP undermåligt då den inte kan bistå med en säker överföring av känslig data. För att skapa säkrare förbindelser så utvecklades SSL,*Secure Socket Layer*, och senare dess efterträdare TLS,*Transport Layer Security*. Deras mål är bland annat att tillhandahålla klient och server autentisering, data konfidentialiet och data integritet. TLS är nästintill identiskt med SSL version 3 och har därför fått namnet SSL/TSL.

|  |
| --- |
| Application layer |
| *SSL/TLS* |
| Transport layer |
| Internet layer |
| Link layer |

*Fig. 1.1*

I TCP/IP modellen så ligger SSL/TLS mellan *Application layer* och *Transport layer* där den arbetar med att skapa säker, krypterad förbindelse mellan två system. Till sin hjälp så använder protokollet digitala certifikat som hjälper till att identifiera de olika användarna. En gemensamt pålitligt tredje part, så som CA, utfärdar certifikat efter att identiteten har bekräftats. SSL/TLS använder sig utav symetrisk nyckel, känt som *private key*, för att kryptera meddelande och en asymmetrisk nyckel, *public key*, för att autentisera.

Digitala Certifikat

Digitala certifikat används idag som ID-kort på internet. Varje certifikat, som också kallas den publika nyckeln, binder ett objekt med dess privata nyckel. Det är endast CA, *Certificate Authority*, som kan utfärdat ett certifikat. Med asymmetrisk kryptografi så kan i princip vem som helst skicka ett krypterat meddelande till en användare och endast användaren själv kan dekryptera meddelandet med sin privata nyckel. I vårt system så har vi valt att använda oss utav av X509 systemet för att utfärda certifikat.

Projekt

Läkarjournaler innehåller väldigt känslig information om någon person och bör därför hanteras väldigt varsamt. Man vill undvika att en obehörig person ska kunna få tillgång till denna data. Eftersom SSL/TLS eftersträvar klient och server autentisering samt datakonfidentialitet och dataintegritet så bidrar en implementation av detta protokoll till ett säkrare system i vårt projekt.

Vårt system

Server  
cert.crt  
key.jks  
[trustStore]

"Database"  
Journal(doctor, nurse, patient)

Handshake

*Trusted Third Part*

CA

DoctorXY

Entity  
cert.crt  
key.jks  
[trustStore]

NurseXY

PatientXY

Agent

*Fig. 1.2*

Servern har lagrad information om alla läkare och sjuksköterskor tillsammans med deras avdelning. Dessutom lagras alla läkarjournaler med uppgifter dels om patienten men också om vilken läkare och sjuksköterska som behandlade patienten. Användare som kan ta del av dessa journaler är läkare, sjuksköterskor, personer från statliga organ och patienten, som har rätt till att se sin egen läkarjournal.

Hur fungerar SSL/TLS i vårt system.

SSL/TLS drar full nytta av digitala certifikat för att autentisera avändaren till vårt system. Alla som vill ansluta till vår server måste ha ett certifikat som är särskilt utfärdad av den pålitliga tredje parten som vi kallar CA.

Persons

PatientXY

Database

Server

Employee

NurseXY

Government

Agent

Employee

DoctorXY

*Trusted Third Part*

CA

*Fig. 1.3*

För att kunna få ett certifikat så måste alla nya användaren genomgå samma process.

Handshake

Innan någon krypterad överföring kan påbörja så måste klienten och servern genomgå en sk. *handshake*. Avsikten med detta är autentisera och för båda system att komma överens om hur de ska kryptera sin förbindelse.

En *handshake* ser ut på följande sätt i vårt system.

1. Client -> Server: Client Hello  
2. Server -> Client: Server Hello, [Server Certificate], [Server Key Exchange],  
 [Client Certificate Request], Server Hello Done  
3. Client -> Server: [Client Certificate], Client Key Exchange,[Certificate Verify],  
 Change Cipher Spec, Finished  
4. Server -> Client: Change Cipher Spec, Finished

1. Client -> Server: ClientHello

Klienten är den som alltid påbörjar en *handshake* med servern. *Client Hello* används för att starta en ny session eller att återuppta en existerande. Första meddelandet innehåller:

|  |
| --- |
| **Version Number.***Den högsta version av SSL/TLS som klienten stödjer.*  TSLv1. |
| **Client Random.***Ett 4-byte långt nummer som innehåller klientens datum och tid samt ett 28-byte långt kryptogenererat slumpmässigt nummer. Detta kommer att användas vid beräkning utav Master Secret.* |
| **Cipher Suite.** *Klienten listar alla krypteringsuppsättningar den stödjer. Ett exempel är TLS\_RSA\_WITH\_DES\_CBC\_SHA.* |
| **Compression Algorithm.** *Valfritt ifall man vill använda komprimeringsalgoritmer. Vi har valt att inte använda den.* |

Client Hello:

2. Server -> Client: Server Hello, [Server Certificate], [Server Key Exchange],  
 [Client Certificate Request], Server Hello Done

Servern svarar tillbaka på *Hello* men skickar även med autentisering mot klienten och kräver att klienten också gör det.

|  |
| --- |
| **Version Number.** *Servern skickar tillbaka högsta versionen som stödjs av båda sidorna. Detta protokoll kommer användas under hela uppkopplingen.**Vårt fall :*  TSLv1. |
| **Server Random.** *Servern generar också ett 28-byte långt krypografiskt slumpmässigt nummer och 4-byte som definierar serverns tid och datum. Tillsammans med klientens slumpmässiga värde så kommer både servern och klienten att generera Master Secret* |
| **Cipher Suite.** *Servern kommer välja den starkaste chiifer som både klienten och servern stödjer. I vårt fall så är det:* TLS\_DHE\_DSS\_WITH\_AES\_128\_CBC\_SHA |
| **Compression Algorithm.** *Använder vi oss inte av.* |

Server Hello:

Tolkningen av krypteringsuppsättningen, *Cipher suits*:

- TLS: Anger vilken protokollversion som ska användas  
- DHE & DSS: Anger vilka algoritmer som kommer att användas vid   
 beräkning av *Master Secret.*  
- AES 128 & CBC: Anger krypteringsalgoritmer som kommer att används.  
- SHA1:A Anger hashfunktionen.

## Attackmetoder

MITMEtt farligt scenario är om data som skickas från klient till server kan stoppas på vägen av en illvillig part (kallad man in the middle, MITM) och använder denna information för att ”ta över” uppkopplingen mot servern. Förutsatt att detta lyckas styr MITM all kommunikation mellan server och klient. Han får således tillgång till all sekretessbelagd information som skickas och har även möjlighet att utföra egna handlingar i klientens namn. Journalsystemet avhjälper detta genom att klient/server att först autentiserar sig mot varandra med hjälp av certifikat signerade av en TTP.

När klienten påbörjar uppkopplingen mot servern (skickar ”client hello” som beskrivet i första delen) kan en MITM genskjuta klientens paket, modifiera det och därmed se till att serverns svar når honom först. Serverns svar innehåller dock ett krav på att användaren ska skicka sitt certifikat och ett ”certificate verify” (sessionsspecifikt meddelande krypterat med klientens privata nyckel). Så även om MITM kan sno certifikatet och hävda att det är han som skickat det har han inte tillgång till klientens privata nyckel och kan därför inte skapa en egen session mot servern. Han har heller inte möjlighet att återanvända det uppsnappade certificate verify senare i ett försök till en replay-attack, eftersom meddelandet endast är giltigt för den pågående sessionen.  
Under förutsättning att MITM inte har ett eget giltigt certifikat som är inlagt i antingen klientens eller serverns truststore kan denne inte mer än vidarebefordra autentiseringsmeddelandena.

Avsaknaden av klientens (eller serverns) privata nyckel innebär också att han inte heller har möjlighet att få tag i den symmetriska nyckeln som krypterar överföringen av patientjournalerna. Den använda metoden (Diffle-Huffman) möjliggör skapandet av den hemliga nyckeln utan att den faktiskt skickas mellan parterna.

Med andra ord, MITM kan inte heller utge sig för att vara någon av parterna eftersom han saknar de privata nycklarna som krävs för autentiseringen. Då han inte har kännedom om den hemliga nyckeln kan inte heller ta del när överförandet av den känsliga informationen börjar, även om han får tag i alla paket.

Utöver dessa säkerhetsåtgärder säkerställer en MAC att det paketet är detsamma som det avsända. Alla försök att sabotera paket längs vägen kommer därmed noteras.

Systemet är av ovan nämnda väldigt motståndskraftigt mot MITM-attacker så länge de privata nycklarna förblir okända för alla andra utom just den tilltänka användaren. Tilläggas kan dock att en MITM kan utföra en DOS-attack på grund av möjligheten att strypa paketflödet i en av riktningarna.

Spoofing  
Systemet är känsligt för spoofing-attacker, i och med att alla keystores integritet är vital för att stänga ute obehöriga. En ”falsk” klient som samlar in information om en användares keystore skulle få tillgång till såväl nyckelpar som certifikat. Detta medför att den specifika användarens inloggning skulle vara röjd och samtliga journaler denne användare har tillgång till utsatta för risk. Detta kan motverkas genom att utbilda användarna i vad de bör kontrollera för att verifiera klientens äkthet, innan de börjar använda klienten.

I och med att det är troligt att någon form av malware måste användas till att utföra spoofing-attacken kan man även vidta åtgärder på de fysiska maskinerna där klienten finns installerad. Att t.ex. begränsa nedladdnings-/installationsmöjligheterna på samtliga datorer där klienten finns installerad skulle kunna förhindra obehöriga från att få tillgång till journaldatabasen. Detta kan dessvärre vara svårt att implementera eftersom det kan finnas verksamhet som gör att man inte kan genomföra den typen av restriktioner.

Avlyssnad kommunikation  
Det finns alltid en risk för att paket innanhållande journalinformation snappas upp av någon som avlyssnar kommunikationslinjen mellan klient och server. Eftersom dessa paket är krypterade med en svårforcerad symetrisk kryptering (AES med 128 bitars nyckel) är det rimligt att anta att man med dagens datorkraft inte kan forcera krypteringsskyddet inom rimlig tid. Detta innebär således att när uppkopplingen mot servern övergått till det krypterade läget kan användaren känna sig säker på att ingen kan snappa upp informationen längs vägen.

Det går däremot inte att skydda paket mot offline-attacker i all evighet, vilket innebär att någon med datorkraft och tid nog förr eller senare kommer åt journalinformationen oavsett hur svårforcerad krypteringsalgoritmen är. Även de starkaste symmetriska krypteringsalgoritmerna som inkluderas i mjukvaran, så som AES, kommer med stor säkerhet gå att forcera i framtiden. Detta eftersom beräkningskapaciteten för den här typen av operationer ökar i samband med att ny datorteknik görs tillgänglig.

Falska certifikat  
En icke-auktoriserad användare skulle kunna försöka skapa ett förfalskat certifikat och med hjälp av detta försöka få tillgång till materialet på servern. Emellertid skulle detta inte vara signerat av CA-certifikatet och skulle således uppdagas som falskt. Systemet utestänger alltså alla som inte fått ett certifikat utfärdat, vilket gör att man inte kan penetrera systemet utan ett giltigt certifikat.

Enda möjligheten att lyckas med ovan nämnda attack är om utfärdaren av klientcertifikaten skulle hackas och utfärda certifikat till obehöriga.

En annan variant hade varit att i klientens trust store placera ett falskt certifikat och med hjälp av detta försöka få klienten att koppla upp sig mot en falsk server. Detta kräver dock lösenordet till klientens truststore, vilket bör vara därmed mindre intressant att hacka (än lösenordet till klientens keystore) i och med att detta inte ger direkt tillgång till servern. Ett osannolikt men dock möjligt scenario hade varit att man genom att plantera falska ”Trusted CertEntries” skulle kunna få klienter att koppla upp sig mot en falsk server och få ut felaktig journalinformation.

Falska certifikat kan dock användas som en del i en DOS-attack, vilket behandlas nedan.

Lösenordspolicy  
En strikt lösenordspolicy är nödvändig för att kunna undgå vanliga typer av attacker för att forcera lösenord. Lösenordet bör vara minst 12 tecken långt, innehålla varierande teckentyper och inte bestå av ord. En sådan policy gör det förvisso svårare för användarna att minnas sina lösenord men har å andra sidan säkrat systemet mot Brute force och Dictionary-attacker.

Vårt system loggar antalet felaktiga för att skydda enskilda keystores och kan göra systemadministratörer varse om att försök görs att hacka en specifik användare. Skyddet för keystores bygger utöver detta på användaren väljer ett starkt lösenord och på att datorer som har klienten installerad förblir fria från malware. Med restriktioner i vad som får finnas laddas ned/installeras på berörda datorer kan man minska sannolikheten att en angripare lyckas installera program designade för att utföra ovan nämnda attacker.

Eftersom lösenordet inte sparas i vare sig klartext eller hashad version, utan enbart är nyckeln i algoritmen som låser upp en specifik keystore, finns det ingen risk för attacker som innefattar dumpning av en lösenordsfil. Regnbågstabeller och memory-tradeoff-attacker kan därför inte användas för att knäcka lösenordet.   
  
Mänskliga faktorer  
Innan systemet tas i bruk måste samtliga användare informeras om vikten av att inte lämna oavslutade sessioner igång då detta kan leda till att obehöriga får tag i/ändrar journalinformation.

Användarna bör instrueras att välja ett starkt lösenord (se lösenordspolicy) och byta det vid misstanke om att någon annan fått kännedom om det eller på uppmaning från en administratör. Lösenordsbyte å regelbunden basis är däremot inte en policy som kommer att genomdrivas. Att byta ett redan starkt lösenord försvårar maximalt en bruteforce-attack med en faktor 2 och uppmuntrar användarna till att välja lösenord som är lätta att minnas men svaga.

Användarna ska också uppmanas att aldrig ge ut lösenordet till någon och att också rapportera alla typer av försök att fiska lösenord till systemadministratörer.  
  
Utöver detta måste det finnas restriktioner mot att på spara och hantera journalinformation utanför klienten, eftersom okrypterade lokala kopior av informationen utgör en säkerhetsrisk.

DOS-attackDet är svårt att skydda sitt system mot attacker som är designade för att överbelasta ett och på så vis utestänga legitima användare. Icke desto mindre kan man försvåra processen genom att kräva att användare uppfyller vissa krav för att tillåtas koppla upp sig.

I detta fall görs detta genom att klienten kräver att användaren har en keystore som denne kan låsa upp med hjälp av användarnamn och lösenord. Så även under förutsättning att en eller flera datorer med klienten installerade skulle bli en del av ett botnet skulle systemet fortfarande bara kunna angripas av användare med en keystore.

Man behöver däremot inte ha ett äkta certifikat för att iscensätta en DOS-attack. Detta möjliggör ett angreppsscenario liknande en slowloris-attack, där en eller flera datorer med klienten installerad skapar keystores med falska certifikat. Sedan startar dessa mängder av uppkopplingar och låter servern vänta på att användaren ska skicka sitt certifikat. Detta gör att servern fyller upp sin kvot för maximalt antal användare som kan vara uppkopplade samtidigt och hindrar därmed användare från att ansluta till servern.

Genom att enbart tillåta att administratörer hanterar skapandet av nya användare och keystores kan dock en sådan attack försvåras avsevärt.

### Avläsning av knapptryckningar / Keystroke logging

Som tidigare nämnt är systemet beroende av att lösenordet till varje klients keystore. Att plantera en keystroke logger på en dator där klienten finns kan få tag i såväl lösenord som användarnamn och ID.

Eftersom en keystroke logger kan underminera säkerheten i hela systemet är det av yttersta vikt att den typen av mjukvara/hårdvara inte finns installerad på datorerna.

Elektromagnetisk avläsning av knapptryckningar är i fallet ett orimligt scenario eftersom datorerna befinner sig inne på ett sjukhus där mängder av datorer används simultant. Att urskilja en specifik dators knapptryckningar för att kunna få ut ett lösenord är under sådana omständigheter inte möjligt med dagens teknik.

### Sammanfattning

Det finns i synnerhet två skeden där systemet är utsatt för risk, nämligen när användaren skriver in sitt alias/lösenord och när den faktiska journalinformationen överförs. I den förstnämnda ligger stort ansvar på användaren och på maskinen där klienten är installerad medan systemet själv försvarar mot det sistnämnda.

Journalsystemet skyddar mycket väl mot alla typer av MIT M och Eavesdropping-attacker. All känslig information skickas krypterad med 128 bitars AES-kryptering och den hemliga nyckeln aldrig skickas mellan servern och klienten. Det senare görs genom att förhandla fram nyckeln till denna symmetriska kryptering med hjälp av Diffle-Huffmanns metod.

Systemet är dock känsligt för alla typer av lösenordsstöld som kan inträffa på lokal nivå. Här krävs en kombination av ett svårforcerat lösenord, säkra datorer och medvetenhet hos användarna för att upprätthålla hög säkerhet.