Uppgift 1: Åtkomstövervakning

Åtkomstövervakning är en av de viktigaste uppgifterna när man skapar en trygg datateknisk omgivning. Åtkomstövervakningens syfte är att förhindra obehörig inloggning på datorn samt otillåten användning av program och filer. Vanligen är det här förverkligat så att datorn eller programmet frågar efter *användarnamn* och *lösenord*. Ifall dessa ges korrekt kan användaren logga in på datorn och han får rättigheterna som motsvarar användarnamnet. Rättigheterna definierar vilka mappar och filer användaren kan se, vilka program han kan köra och vilka externa enheter, såsom skrivare och externa minnen, han kan använda. I fortsättningen kallar vi mappar, filer, program och externa enheter med det gemensamma namnet *resurser*.

Den enklaste attacken mot ett datasystem är att försöka gissa kombinationen av användarnamn och lösenord. Gissandet kan automatiseras så att man kan göra en väldig mängd gissningar inom en rimlig tid. Nuförtiden används så kallade ordböcker i samband med gissningarna. Dessa är långa listor som innehåller exempelvis nästan alla språkets ord. Förutom vanliga ord används också andra vanliga kombinationer, såsom ord baklänges och kombinationer av ord och siffror. En sådan attack kallas *brute force attack*. För att avvärja en sådan attack rekommenderas det att man inte använder något språks verkliga ord som lösenord, utan en godtycklig kombination bokstäver, siffror och specialtecken. Dessutom bör lösenordet vara tillräckligt långt, minst 10 eller 12 tecken.

Inte bara användaren utan också systemet som kontrollerar användarnamnet och lösenordet bör beakta möjliga attacker. Användarnamnen, rättigheterna som hör till dem och lösenorden är lagrade i filer. Om utföraren av attacken lyckas få användarrättigheter till systemet, till exempel genom ett fel i programvarans datasäkerhet, kan han möjligtvis läsa också filen med lösenord. Av denna orsak får lösenord inte lagras i klartext. Lösenorden i filen är antingen krypterade eller nuförtiden används ofta så kallad kryptografisk hash. En hash kan beräknas för en godtyckligt lång teckensträng och resultatet är av en viss längd, vanligen 128, 256 eller 512 bitar långa bitsträngar (dvs en sträng av tecknen 0 och 1). Hashfunktionen bör uppfylla vissa väsentliga krav innan den kan användas i samband med lösenord. Om f är en hashfunktion, M teckensträngen för vilken hashen beräknas, och $f(M)$ hashen för M, bör man inte kunna bestämma M från resultatet $f(M)$ inom någon rimlig tid. Med andra ord är hashfunktionen *enkelriktad*: hashen är rätt lätt att beräkna, men utgående från hashen är det inte möjligt att bestämma den ursprungliga teckensträngen. Ett annat krav på hashen är att det inte rimligtvis är möjligt att producera två teckensträngar med samma hash. Med andra ord, det är inte möjligt inom en rimlig tid att hitta två teckensträngar M_1 och M_2 för vilka $f(M_1) = f(M_2)$. Rimlig tid betyder här några år.

Ett exempel på en enkel hashfunktion är följande. Låt M vara en teckensträng och L dess längd. Vi delar upp M i k avsnitt av längd n: $M_1, M_2, \ldots, M_k$. Här är n mycket mindre än L. Det sista avsnittet utfylls med bitar så att också det har samma längd som de andra, dvs n. Nu behandlar vi bitarna i avsnitten en bit i taget, så att motsvarande bitar i avsnitten summeras enligt följande regler: $0 + 0 = 0$, $0 + 1 = 1$, $1 + 0 = 1$, $1 + 1 = 0$. Exempelvis $M = 1010\ 1110\ 0011\ 0101$, $L = 16$, $n = 4$, $k = 4$. Då blir $1010 + 1110 + 0011 + 0101 = 0010$ ($1 + 1 + 0 + 0 = 0$, $0 + 1 + 0 + 1 = 0$, $1 + 1 + 1 + 0 = 1$, $0 + 0 + 1 + 1 = 0$). Denna hash är för kort och hashfunktionen för enkel för att kunna vara en verklig hashfunktion, men fungerar som en sorts grund för mera komplicerade hashfunktioner. I verkliga hashfunktioner utförs motsvarande summeringar av bitar, men dessutom andra operationer, såsom utbyte av olika platsers bitar.

Enbart genom att ersätta lösenord med hash kan man inte avvärja brute force attacker. För att sådana attacker inte skall lyckas, görs beräkningen av hashen långsam med avsikt. Exempelvis kan en hash beräknas många gånger. Även om en människa inte märker någon fördröjning, kan beräkningen i verkligheten vara mer än hundra gånger långsammare, vilket förhindrar att en brute force attack skulle lyckas inom rimlig tid. Ett annat enkelt tillvägagångssätt är att man inte meddelar vid misslyckad inloggning exakt vad som gick fel. Till exempel meddelar man inte om det var lösenordet som var fel eller om det var användarnamnet som var fel, utan helt enkelt att kombinationen av användarnamn och lösenord var fel. Då vet inte utföraren av attacken vilkendera som var rätt eller om båda var fel.

En annan situation uppstår om utföraren av attacken fått tag på själva lösenordsfilen. Nu har utföraren av attacken effektivare möjligheter att testa lösenord och användningen av ordböcker är mera avancerad. Utföraren av attacken har färdigt beräknade hash på basen av orden i ordboken, ordnade så att det går snabbt att jämföra hashen i lösenordsfilen med de färdigt beräknade hashen. Tabellerna med hash kan vara av storleksordningen terabyte. För att göra sådana här attacker svårare lägger man en extra bitsträng till lösenordet innan man beräknar hashen. Denna extra bitsträng kallas *salt*. Saltets längd är 64 eller 128 bitar. För varje användare skapas ett individuellt saltvärde. Således innehåller lösenordsfilen (eller lösenordsfilerna om användarnamnens information är uppdelat på flera filer) åtminstone användarnamn, saltet och hashen av det saltade lösenordet.

Till åtkomstövervakningen hör också mekanismer för att hålla bokförning över rättigheterna. Exempelvis bör systemet veta gällande filerna vem som har vilka rättigheter för respektive fil. De vanligaste konstruktionerna för en sådan bokföring är *åtkomstmatriser*, *åtkomstlistor* och *rättighetslistor*. En åtkomstmatris är en $(m \times n)$-tabell där m är antalet subjekt och n är antalet objekt. Subjekten är här användarna, även om de i andra sammanhang också

kan vara till exempel program. Objekten är resurser, vilka vi definierade tidigare. I tabellens plats (i, j) antecknas rättigheten som subjektet i har till objektet j. Problemet med åtkomstmatriserna är att de lätt blir väldigt stora, även om de flesta platserna är utan rättigheter. Åtkomstlistorna sparar utrymme. Objektets åtkomstlista innehåller alla subjekt som har rätt till objektet. Subjektets rättighetslista är den motsatta listan med alla objekt som subjektet har rättigheter till.

Innan man bestämmer vilken åtkomstövervakningsmekanism man ska välja att implementera är det befogat att skissera principerna för åtkomstövervakningen. Olika åtkomstövervakningsmodeller underlättar den här planeringen. De tre viktigaste modellerna är *regelbaserad*, *individbaserad* och *rollbaserad* åtkomstövervakning. I den regelbaserade modellen ges på förhand reglerna enligt vilka rättigheterna bestäms. Subjektet kan inte ändra på objektets rättigheter, även om han hade maximala rättigheter till objektet. I den individbaserade modellen kan subjekten bestämma rättigheterna för sina objekt. Subjekten kan också ändra rättigheterna för sina objekt. I den rollbaserade modellen har rollerna rättigheterna. Subjektet bakom en roll kan bytas ut och flera subjekt kan ha samma roll. Subjektet kan också ha många roller. Den rollbaserade modellen kan i övrigt vara antingen regelbaserad eller individbaserad.

Till slut kan man ännu konstatera att det inte är enkelt att definiera rättigheterna. Exempelvis har filer vanligen tre rättigheter: läsrättighet, skrivrättighet och exekveringsrättighet. Samma rättigheter hänför sig också till mapparna. Redan här har vi ett tolkningsproblem: vad betyder exekveringsrättighet för en mapp? Ett annat problem uppstår om användaren har rätt att skriva till filen, men inte rätt att skriva till mappen där filen finns. Får han då skriva till filen eller ej? Exempelvis Windows och Linux löser dessa problem på olika sätt.

I Linux betyder läsrättigheten till mappen att subjektet kan studera mappens innehåll, dvs studera vilka filer och undermappar det finns i mappen. Skrivrättigheter till mappen betyder att subjektet kan skapa en fil i mappen. Exekveringsrättighet till mappen betyder att subjektet kan gå vidare från mappen till dess undermapp. Ett speciellt drag i Linux är att man kan läsa en fil endast om man har exekveringsrättighet till alla mappar på mappvägen från roten till filen.

I Windows är rättigheterna mera komplicerade. Det finns flera sorters rättigheter och subjekten har större möjligheter att göra regler exempelvis för olika grupper av användare. Dessutom har Windows uttryckliga förbudsregler, vilka saknas i Linux. I Linux är allt sådant automatiskt förbjudet som inte är direkt tillåtet. I synnerhet gällande mapparna skiljer sig Windows från Linux: det är möjligt att läsa en fil i en mapp till vilken man inte har några rättigheter.

Frågorna

Fråga 1. Om tangentbordet har m tecken, hur många olika lösenord av längd k finns det? (maximalt antal poäng 2)

Fråga 2. Beskriv vilka slags lösenord är dåliga, även om de skulle vara tillräckligt långa. Ge också karakteristika på ett gott lösenord. (maximalt antal poäng 3)

Fråga 3. För hashfunktionen som ges i materialet är det lätt att hitta sådana bitsträngar som resulterar i samma hash. Vi ändrar nu på hashfunktionen så att före summeringen av avsnitt i roteras bitarna i steg till vänster. Till exempel om det första avsnittet är 1100, så blir det 1001. Om avsnittet var det andra avsnittet, skulle det bli 0011. Om det var det tredje avsnittet, skulle det bli 0110. Om det var det fjärde avsnittet, så skulle det bli oförändrat. Ge för denna modifierade hashfunktion två olika 16 bitar långa bitsträngar som resulterar i samma fyra bitar långa hash. (maximalt antal poäng 3)

Fråga 4. Hur försvårar användandet av salt gissningen av lösenorden i det fallet att utföraren av attacken har lagt beslag på lösenordsfilen, även om användarnamnen och saltet syns i klartext i lösenordsfilen? (maximalt antal poäng 3)

Fråga 5. Jämför åtkomstlistor med rättighetslistor. Använd som bas för jämförelsen användningen av utrymme och effektiviteten av sökningarna. Exempel på sökningar: vem har läsrättighet till en fil eller till vilka objekt har ett subjekt någon rättighet. (maximalt antal poäng 3)

Fråga 6.
a) Ge ett exempel på ett system eller en applikation hörande till en datoromgivning som följer en regelbaserad modell och ett motsvarande exempel som följer en individbaserad modell. (maximalt antal poäng 3)
b) Ge motsvarande exempel på organisationer eller system som inte hör till en datoromgivning. (maximalt antal poäng 3)

Fråga 7.
a) I slutet av materialet diskuteras mappar och mappvägar. Definiera dessa begrepp. (maximalt antal poäng 2)
b) I Windows är rättigheterna mera omfattande och mera komplicerade än i Linux. Dryfta kort denna lösnings goda och dåliga sidor. (maximalt antal poäng 3)