

MARTIN KARRESAND AMUND HUNSTAD



FOI är en huvudsakligen uppdragsfinansierad myndighet under Försvarsdepartementet. Kärnverksamheten är forskning, metod- och teknikutveckling till nytta för försvar och säkerhet. Organisationen har cirka 1000 anställda varav ungefär 800 är forskare. Detta gör organisationen till Sveriges största forskningsinstitut. FOI ger kunderna tillgång till ledande expertis inom ett stort antal tillämpningsområden såsom säkerhetspolitiska studier och analyser inom försvar och säkerhet, bedömning av olika typer av hot, system för ledning och hantering av kriser, skydd mot och hantering av farliga ämnen, IT-säkerhet och nya sensorers möjligheter.

Martin Karresand
Amund Hunstad

Försvar av IT-system

Slutrapport

Titel	Slutrapport – Försvar av IT-system
Title	Final report – IT System Defense
Rapportnr/Report no	FOI-R--2921--SE
Rapporttyp Report Type	Användarrapport
Sidor/Pages	34 p
Månad/Month	December/December
Utgivningsår/Year	2009
ISSN	ISSN 1650-1942
Kund/Customer	FM
Projektnr/Project no	E7115
Godkänd av/Approved by	Anders Törne

FOI, Totalförsvarets Forskningsinstitut	FOI, Swedish Defence Research Agency
Avdelningen för Informationssystem	Information Systems
Box 1165	Box 1165
581 11 Linköping	SE-581 11 Linköping

Sammanfattning

Detta är slutrapporten som redovisar tre års verksamhet i FoT-projektet Försvar av IT-system (FITS). Rapporten redovisar en övergripande bild av verksamheten med tyngdpunkt på hur resultaten kommer Försvarsmakten till nytta. Verksamheten har omfattat två huvudspår; datatypsidentifiering av datafragment och IT-säkerhetsövningar. Forskningen om datatypsidentifiering kommer FM till nytta genom att den utökar förmågan hos intrångsdetekteringssystem. Övningsutvecklingsverksamheten kommer FM till nytta genom att fungera som avtappning av den erfarenhet som FOI byggt upp, samt genom FM:s användning av den övningsplattform med tillhörande spårbarhetssystem som utvecklats.

Nyckelord: IT-system, IT-försvar, IT-säkerhet, övning, identifiering av datatyp

Summary

This is the final report presenting three years of activity in the R&D project Defence of IT systems. The report gives an overview of the activities focusing on how the results will benefit the Swedish Armed Forces (SAF). There has been two main paths of research; identification of data type of data fragments, and IT security exercises. The research concerning identification of data type will benefit the SAF by extending the capability of intrusion detection systems. The IT security exercise development activity will benefit the SAF by working as a transfer media for the experiences that FOI has gained in the field, and through the SAF's use of the exercise platform with connected logging tools that has been developed.

Keywords: IT system, IT defence, IT security, exercise, file carving

Innehåll

1	Inledning	7
1.1	Bakgrund	7
1.2	Nyttomål	7
1.3	Frågeställningar	7
2	Projektresumé	8
2.1	Publikationer	8
2.2	Intrångsdetektion	9
2.3	IT-försvar	9
2.3.1	Bytefrekvensfördelning	11
2.3.2	Förändringsgrad	12
2.3.3	Bytevärdesrepetition	14
2.3.4	Bytepar	15
2.3.5	Manhattanavstånd	18
2.3.6	Viktad kvadratsumma	18
2.3.7	Sannolikhet	19
2.3.8	Vinkel mellan vektorer	20
2.3.9	Typexemplar	22
2.3.10	Sätta samman datafragment	23
2.4	Gemensam lägesbild	24
2.5	IT-försvarsövning	25
2.5.1	Övningens upplägg	26
2.5.2	Försvarsmaktens nytta av IT-försvarsövningar	27
2.5.3	FOI Reconstruction & EXploration (F-REX) Studio	28
2.5.4	Övningsförberedelser	28
2.6	Omvärldsanalys	29
2.6.1	Datatypsidentifiering	29
2.6.2	Övning	31
2.7	Nyttjandeplan för Försvarsmakten	32
2.8	Samverkan ”Lägesförståelse på informationsarenan”	32
3	Summerande slutsatser	33
3.1	Datatypsidentifiering	33
3.2	Övningsverksamhet	34

1 Inledning

Under tre års tid har FOI bedrivit forskning kring försvar av IT-system inom ramen för FoT-projektet Försvar av IT-system. Detta är slutrapporten för projektet i form av en användarrapport, som beskriver hur Försvarsmakten (FM) kan använda de resultat som projektet har uppnått.

1.1 Bakgrund

De angripare som framtida ledningssystem kan komma att möta är kompetenta och mycket resursstarka och skiljer sig på så sätt från de angripare som oftast förekommer i den civila världen, och som därför kommersiella IT-säkerhetssystem är designade för. Därför behöver metoder för och aspekter kring försvar av IT-system studeras och då speciellt aktiva åtgärder i form av bland annat detektion och övervakning.

1.2 Nyttomål

Projektet bidrar till FM:s operativa förmågor vad avser möjligheten att uppnå ledningsöverläge genom att bidra till ett väl fungerande försvar av egna ledningssystem. De tjänster man vill kunna tillhandahålla i ett framtida taktiskt ledningssystem kräver bland annat en högre grad av detektions- och övervakningsförmåga än vad dagens IT-säkerhetssystem kan erbjuda. Därför har ett av målen med projektet varit att utveckla metoder och verktyg för att på sikt stärka FM:s förmåga till detektion och övervakning. Projektets verksamhet kring övningar av IT-säkerhet har bland annat syftat till att möjliggöra praktisk tillämpning av detektion och övervakning i en kontrollerad miljö.

Mer detaljerad information om nyttan med projektets olika verksamheter återfinns i separata underrubriker i Kap. 2.

Fremsta avnämare av resultaten från projektet är FM/ITF. Resultaten från projektet kan dock även användas av till exempel MUST/SÄK, Utlandsstyrkan, FMV och FRA för en mängd uppgifter. På den civila sidan återfinns potentiella användare inom bland annat MSB (före detta KBM), RKP, SÄPO och SKL.

1.3 Frågeställningar

Arbetet i projektet har varit inriktat på att i möjligaste mån besvara följande forskningsfrågor. Frågorna är specificerade i projektplanen och har därför en mer generell formulering än vad de resultat som presenteras i rapporten antyder.

Varje fråga, de relaterade resultaten och nyttan av dem behandlas under separata rubriker i Kap. 2.

- Hur detekteras generella intrångs- och störningsförsök i det egna systemet? (Kap. 2.2)
- Vilka tekniker och metoder kan användas för försvar av IT-system och för IT-spaning och intrångsanalys? (Kap. 2.3)
- Hur sammanställs och tolkas en gemensam lägesbild av hela nätets status? (Kap. 2.4)
- Hur ska övningar i IT-försvar planeras, utrustas och genomföras? (Kap. 2.5)
- Vilken verksamhet pågår i omvärlden? (Kap. 2.6)

2 Projektresumé

I sedvanlig ordning har kunskapsöverföring till FM skett genom vetenskaplig publicering, FOI-rapporter, referensgruppsmöten och seminariedagar tillsammans med de övriga FoT-projekt på IT-säkerhetssidan. Dessutom har personal från projektet deltagit vid workshopar och utbildningar med FM-anknytning. Verksamheten har då främst inneburet Tivolikörningar. Våra nära relationer med relevanta grupperingar inom FM gör att kunskapsöverföringen blir direkt, men mer informell.

Verksamheten inom projektet kan delas upp i två huvudspår. Det ena spåret har behandlat intrångsdetektering och då främst möjliga sätt att utöka kapaciteten hos befintliga system genom datatypsidentifiering¹ av datafragment. Ett exempel är fragment i form av nätverkspaket och sökning sker då främst efter maskinkod, det vill säga den operativa delen av elakartad kod.

Det andra spåret inom projektet har syftat till att praktiskt undersöka, utveckla och utvärdera övningsverksamhet inom IT-säkerhetsområdet. Detta har gjorts genom att en övningsplattform tagits fram med tillhörande stödsystem i form av ett avancerat spårbarhetssystem. Plattformen har sedan bland annat använts för en internationell, distribuerad övning tillsammans med Estland. Erfarenheterna från den övningen har sedan utgjort underlag för förbättringar och förändringar av plattformen och metodiken för upplägg av kommande övningar.

Planering har skett för framtida samverkan med projekten ”Testning av metoder och verktyg för värdering av informationssäkerhet” och ”Situations- och hotanalys för BG2011”. Vid de möten som hållits har eventuella synergieffekter och gemensamma beröringspunkter mellan de inblandade projekten diskuterats. Mer information om detta finns i Kap. 2.8.

En översiktlig plan för FM:s nyttjande av resultaten har tagits fram i samråd med Claes Doverfors, FM/TM. Planen presenteras i Kap. 2.7.

2.1 Publikationer

Följande publikationer med koppling till projektet har producerats under projektets gång. Vetenskaplig kvalitetssäkring har i tillämpliga fall erhållits genom bidrag till peer-reviewade konferenser. Listan innehåller även en licentiatavhandling.

- Reassembly of fragmented JPEG images containing restart markers. (FOI-S-3091-SE), (European conf. on computer network defense, EC2ND), (p. 25-32). Författare: Karresand, Martin och Shahmehri, Nahid. Ort Linköping, FOI, 2009.
- Completing the picture — fragments and back again. (FOI-S-2925-SE), (2008), (p. 1-112). Författare: Karresand, Martin. Ort Linköping, FOI, 2008.
- Försvar av IT system — Förstudie om projektets fortsatta verksamhet. (FOI Memo 2114). Författare: Karresand, Martin. Ort Linköping, FOI, 2007.
- CADS exercise — Emulating real-life DDOS attacks. (FOI Memo 2759). Författare: Persson, Mats. Ort Linköping, FOI, 2009.
- Sammanfattning av demonstrationsdag IT-säkerhet. (FOI Memo 2623) Författare: Westerdahl, Lars. Ort Linköping, FOI, 2008.
- Sammanfattning av Workshop avseende samverkan gällande lägesförståelse informationsarenan [sic!]. (FOI Memo 2829). Författare: Bengtsson, Johan. Ort Linköping, FOI, 2009.

¹File carving.

- Inriktningsförslag FoT-projekt spaning och motmedel på informationsarenan diskuterat vid referensgruppsseminarium i Enköping 2009-04-21. (FOI Memo 2871). Författare: Karresand, Martin. Ort Linköping, FOI, 2009.
- Leverans av milstolpe 2 Q4 2009. (FOI Memo 3025). Författare: Karresand, Martin. Ort Linköping, FOI, 2009.
- Konferensbidrag skickat till EC2ND'09. (FOI Memo 3026). Författare: Karresand, Martin. Ort Linköping, FOI, 2009.

FOI Memo 2829 och FOI Memo 2623 har skrivits i andra projekts regi. Dokumenten har dock inkluderats i publikationslistan för Försvar av IT-system eftersom personal från detta projekt deltog i workshopen och resultaten direkt berör projektet.

2.2 Intrångsdetektion

Resultaten från studien av de underliggande kraven för att möjliggöra detektion av intrågs- och störningsförsök i egna system har använts vid utvecklingen av de metoder och verktyg för IT-försvar som tagits fram inom projektet. Studien, liksom de praktiska erfarenheterna från metodutvecklingen, pekar på vikten av helhetssyn. Det går inte att utesluta någon abstraktionsnivå eller något abstraktionslager i ett datorsystem eller nätverk om målet är bästa möjliga detektionsförmåga. Det är också viktigt att använda den metainformation som finns i kopplingar och beroenden mellan enskilda parametrar för intrångsdetektion.

Detta återspeglas också i den forskning som bedrivs i omvärlden inom intrångsdetekteringsområdet. På senare tid har korrelering av larm och data från flera system vuxit som delområde. Utvecklingen har även utökat området åt andra hållet i och med att det angränsande datorforensiska forskningsområdet har expanderat kraftigt. Inom det området ligger intresset på den information som data på bitnivå innehåller. Den forskning som bedrivits inom projektet Försvar av IT-system har bidragit till områdets expansion.

Vanligtvis så används det eller de lager som ger mest effekt relativt insatsen, vilket är en bra lösning i civila standardsystem där ekonomiska aspekter är de förhärskande. I militära sammanhang är kraven på detektionsförmåga ofta annorlunda och fokus ligger på sekretess och tillförlitlighet hos informationen i de berörda systemen. Här kan det alltså vara befogat att försöka nå högsta möjliga detektion med lägsta möjliga nivå av falsklarm. Resultatet av studien är inte revolutionerande eller förvånande på något sätt, men kan ändå vara till nytta att peka på eftersom kommersiella system främst är inriktade mot den civila världens ekonomiska balansgång mellan kostnad och effekt.

Försvarsmakten kan nyttja resultaten av studien både i närtid och på längre sikt. I närtid handlar det om att inskräpa vikten av att ha kontroll på såväl detaljer som på hur de är beroende av varandra vid upprättande av operativa datorsystem vid till exempel utlandsuppdrag. På längre sikt handlar det om att på ett tidigt stadium få in ett nytt tankesätt vid utvecklingen av framtida ledningssystem, vare sig det handlar om system som byggs av färdiga produkter från hyllan, eller egenutvecklade systemkomponenter.

2.3 IT-försvar

Projektet har utvecklat en metod, bytefrekvensfördelning med sannolikhetsbaserad metrik², som med 95,7 % säkerhet och 0,006 % falsklarm kan detektera maskinkodsfragment om 512 byte (B) i dataströmmar. Tekniken kan användas för att förbättra funktionen i intrångsdetekteringssystem genom att upptäcka maskinkod insprängd i

²Begreppet metrik kommer från latinet och kan översättas med matematisk avståndsfunktion.

annan trafik. Typiska exempel är upptäckt av exploits av serverprogramvara via nätverkstrafik och detektion av maskinkod som bäddats in i ofarlig data.

Projektet har också utvecklat metoder för att först kunna upptäcka och filtrera ut JPEG-data ur datamängder och sedan sätta ihop de datafragment som hittats till en bild (eller en del av en bild) igen. Metoden för att detektera JPEG-data bygger på frekvensfördelning av bytepar och en viktad kvadratsummemetrik. Metoden har en detektionsgrad på 100 % vid 0,12 % falsklarm. Ett typiskt användningsområde är återställning av data från skadad lagringsmedia. Ett annat användningsområde är vid övervakning av nätverkstrafik så att inte otillåten verksamhet pågår, till exempel distribution av multimediafiler.

Andra användningsområden är forensiska undersökningar av datorer, detektion av vissa typer av steganografi och lågnivåkontroll av krypteringsgrad hos nätverkstrafik.

Alla datatypsidentifikationsalgoritmer (se Kap. 2.3.1 – 2.3.4) som utvecklats inom projektet bygger på mätning av frekvensfördelning för vissa specifika egenskaper hos datafragment. Genom att jämföra mätvärden för dessa egenskaper hos fragment av okänd datatyp med modeller av olika datatyper kan datatypstillhörigheten hos fragmenten bestämmas. En datatypsmodell skapas genom att frekvensfördelningsvärden för en specifik datatyp lagras i en fil vars format är beroende av vilken algoritm och metrik som används.

När datatypen hos ett fragment av okänd datatyp ska identifieras jämförs det med en eller flera modeller och differensen mellan fragmentets och olika datatypmodellers mätvärden beräknas enligt en förutbestämd metrik. Värdet representerar avståndet mellan fragmentet och de olika modellerna. Fragmentet klassificeras sedan som tillhörande samma datatyp som den närmaste modellen.

Beroende på de krav som ställs på den tjänst som datatypsidentifikationsalgoritmen ska ingå i kan olika metriker användas. Eftersom en metrik är en matematisk funktion ska den entydigt bestämma avståndet mellan två fragment. Om avståndet mellan fragment i två grupper mäts ska inte metriken resultat variera i någon större utsträckning. Eftersom det vi mäter ofta har minst 256 dimensioner måste metriken kunna hantera detta utan att resultaten blir lidande av problemet med hög dimensionalitet³, det vill säga att mätvärdenas precision inte påverkas av antalet ingående parametrar.

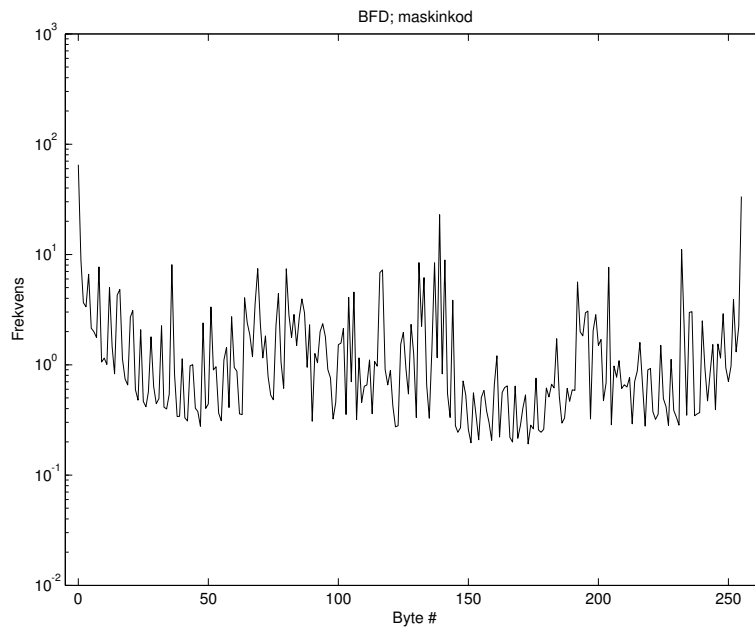
Vad som är en lämplig metrik att använda varierar beroende på tillämpning. Inom projektet har ett antal metriker använts i kombination med de datatypsidentifieringsalgoritmer⁴ som utvecklats. En närmare beskrivning av dessa metriker kommer efter beskrivningarna av algoritmerna (se Kap. 2.3.5 – 2.3.8). Vi presenterar även ett förslag på en alternativ metrik med stödfunktionalitet som använder sig av flera typexemplar⁵ för jämförelse. En närmare beskrivning av denna metrik återfinns i Kap. 2.3.9.

Inom projektet har också en algoritm för sammansättning av små datafragment till större utvecklats. Algoritmen möjliggör återskapande av ursprungsinformationen hos de inblandade fragmenten. Den beskrivs närmare i Kap. 2.3.10.

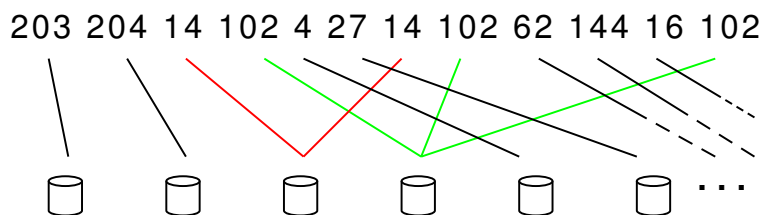
³Problemet med hög dimensionalitet kan enklast beskrivas som att slutvärdet påverkas allt mindre av ett enskilt delvärde ju fler delvärden som ingår. I fallet med bytefrekvensfördelningsalgoritmen och kvadratsummemetrik bidrar varje delvärde med $\frac{1}{256}$ av slutvärdet. För byteparsalgoritmen med kvadratsummemetrik blir bidraget $\frac{1}{65536}$ av slutvärdet.

⁴Den närmaste jämförbara och väl etablerade termen är engelskans *file carving algorithms*.

⁵Flera modeller som tillsammans avspeglar komplexa, heterogena datatyper i detalj.



Figur 2.1: En modell av maskinkod skapad med hjälp av BFD-algoritmen. Observera att Y-axeln har logaritmisk skala.



Figur 2.2: Principen för BFD-algoritmen

2.3.1 Bytefrekvensfördelning

Bytefrekvensfördelningsalgoritmen (BFD) mäter antalet förekomster av varje möjligt bytevärde i ett fragment av en viss storlek, det vill säga frekvensen av varje bytevärde för den fragmentstorleken. I Figur 2.1 visas medelvärde av frekvensen för varje möjligt bytevärde av en stor mängd maskinkodsfragment⁶ framtaget med hjälp av BFD-algoritmen.

Maskinkod har en relativt ojämn fördelning av bytevärden och dess histogram kan förklarats beskrivas som ett W. De höga frekvensvärdena kommer från framför allt hoppinstruktioner i maskinkoden, samt den utfyllnad som görs för att skapa jämna block av kod, vanligen 512 B stora. Dessa saker är exempel på egenskaper som används av BFD-algoritmen för att identifiera datafragment av okänd typ som maskinkod.

Principen bakom BFD-algoritmen är enkel, bytevärdena i en ström av data sorteras och sedan räknas antalet bytar för varje värde. Metoden kräver att ett datafragment av okänd typ ska ha samma storlek som de fragment som användes för att skapa modellen. Figur 2.2 ger en skiss av principen. Figuren föreställer en del av en dataström av maskinkod där bytevärdena visas i positiv decimalform. Notera hur bytevärde 14 och 102 har frekvens 2 respektive 3, övriga frekvens 1.

⁶Det vi här kallar maskinkod är *text*-delen i Windows Portable Executable (PE) filer kompillerade för Intels 32-bitars arkitektur (IA32).

Fördelar

Den enkla principen för BFD-algoritmen gör den mycket snabb jämfört med många andra algoritmer. Enkelheten bidrar också till att den är lätt att implementera och förstå. Det gör att den lämpar sig väl för tillämpningar där tillförlitlighet och låg risk för buggar är viktiga aspekter att ta hänsyn till.

Algoritmen har vid praktisk användning visat sig lämpa sig väl för identifiering av små fragment av data med hög entropi, det vill säga data med mycket slumpmässigt utseende. Det kan till exempel röra sig om krypterad eller komprimerad data. Algoritmen hanterar 512 B stora fragment utan problem, när storleken ökar till 4 KiB sjunker detektionsgraden både faktiskt och relativt de andra algoritmerna som tagits fram inom projektet.

Nackdelar

På grund av att strukturen i den dataström som modelleras inte kan återges av BFD-algoritmen är precisionen dålig, framför allt för större fragment av data. Precisionen är generellt dålig för algoritmen, men detta märks extra tydligt vid större fragmentstorlekar där andra algoritmers egenskaper kommer mer till sin rätt.

Den dåliga precisionen utmärker sig främst genom att specifika, tydliga och unika egenskaper hos en modellerad dataström inte kan återges till fullo. BFD-algoritmen jämnar ut dessa egenskaper och lämpar sig därför bättre för datatyper med relativt jämn bytefrekvensfördelning där det inte finns några tydliga avvikelser i frekvensfördelningskurvan.

Användningsområden

Tillsammans med den sannolikhetsbaserade metriken fungerar BFD-algoritmen mycket bra för exekverbar kod. Om den tränas med mycket data kan den till exempel särskilja AES och CAST5 krypterad data. Vår erfarenhet är att detta dock kräver uppåt 1 GiB data att träna på.

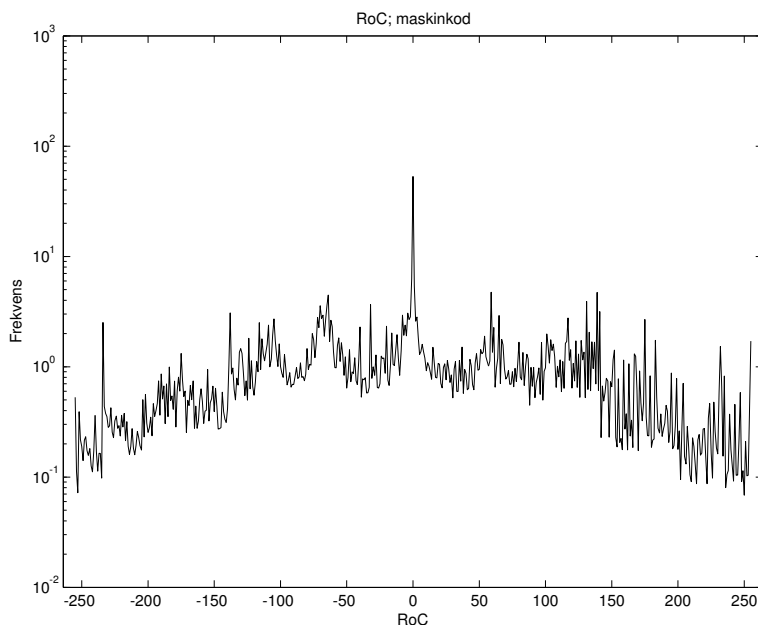
Algoritmen kan framför allt fungera som förstärkning av intrångs- och virussydd. Ett annat användningsområde är för kontroll av att kryptering används i nätverk, utan att det krävs någon dekryptering av nätverkspaketet. Det gör att kontrollen går snabbt och att säkerheten ökar genom att det inte krävs några krypteringsnycklar, som annars kunnat komma på avvägar. Likaså medför det att kontrollen kan utföras utan inblandning av klartext på andra ställen än hos avsändare och mottagare.

På grund av att algoritmen presterar bättre relativt andra algoritmer vid små fragmentstorlekar lämpar den sig för övervakning av nätverkstrafik. Även lågnivåavsökning av lagringsmedia, där till exempel hårddisksektorer är 512 B stora, är ett möjligt användningsområde.

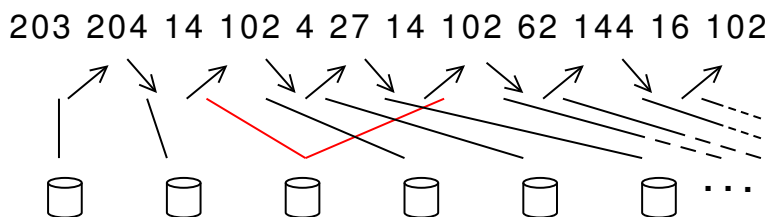
2.3.2 Förändringsgrad

Algoritmen som använder sig av förändringsgraden i en dataström är tänkt att komplettera BFD-algoritmen genom att ta hänsyn till strukturen hos strömmen. Förändringsgraden definieras av oss som differensen mellan två på varandra följande bytevärden x_i och x_{i+1} , det vill säga förändringsgrad = $x_i - x_{i+1}$.

I Figur 2.3 visas ett histogram över förändringsgraden hos maskinkod. Histogrammet visar att vissa stora positiva förändringsgrader är mycket mer vanliga än den teoretiska genomsnittsfrekvensen för förändringsgraderna. Förändringsgrad 0 har en mycket hög frekvens, det vill säga maskinkod innehåller ofta sekvenser med samma bytevärde. Den



Figur 2.3: Histogram för RoC-algoritmen körd på maskinkod. Observera att Y-axeln har en logaritmisk skala.



Figur 2.4: Principen för RoC-algoritmen

främsta förekomsten uppstår vid den utfyllnad som görs för att uppnå jämnstora block av kod. Klockformen hos kurvan kommer sig av den logaritmiska skala som använts och det faktum att antalet sätt att bilda differenserna ökar linjärt från 1 för respektive ± 255 till 256 för en förändringsgrad på 0.

Förändringsgradsalgoritmen (RoC) skapar ett histogram över antalet förekomster av en varje möjlig förändringsgrad. Principen för algoritmen kan ses i Figur 2.4. Dataströmmen i figuren innehåller två likadana förändringsgrader i form av bytevärdena 14 och 102, vilka ger en förändring på 88. Det finns dock 167 andra möjligheter att få en differens på 88, så värdet säger ytterst lite om vilka bytevärden som är inblandade⁷. Byteströmmen i Figur 2.4 kommer från en exekverbar Windows-fil som är något redigerad för läsbarhetens skull. De två identiska bytepar som återfinns där är dock äkta.

Pilarna, som i figuren visar riktningen på förändringen i dataströmmen, pendlar mellan en uppåt- och nedåtgående riktning. Detta är på intet sätt typiskt för maskinkod. Dataströmmen kan likaväl ha en långvarig likriktad trend, det är till exempel inte ovanligt med sågtandsmönster i maskinkod. Summan $X = \sum_{i=1}^{n-1} x_i$ av $n - 1$ på varandra följande differenser x_i hos ett n bytar stort fragment är begränsad som $-255 \leq X \leq 255$ vilket gör att sågtandsmönster är ett extremfall.

⁷Endast för ett förändringsvärde på ± 255 går det att säga att differensen kommer från en kombination av bytevärde 0 och 255.

Algoritmen fungerar bättre med större fragment (4 KiB) än med små (512 B). Vid 4 KiB stora fragment har algoritmen bättre detektionsgrad än BFD-algoritmen. För små fragment sjunker prestandan något och BFD-algoritmen är bättre.

Fördelar

RoC-algoritmen är tack vare sin relativa enkelhet snabb. Den har dessutom dubbelt så hög upplösning som BFD-algoritmen. Den största vinsten med algoritmen är att den tar hänsyn till strukturen hos den dataström som modelleras. Detta gör att den med fördel kan kombineras med BFD-algoritmen.

Hanteringen av olika datatyper är jämnare för RoC-algoritmen än för BFD-algoritmen. Framför allt är RoC-algoritmen bättre på att identifiera datatyper med lägre entropi och klarar av att modellera unika egenskaper hos datatypen någorlunda detaljerat. Prestandan sjunker dock om dataströmmen har hög entropi.

Nackdelar

En nackdel med RoC-algoritmen är att upplösningen är högre för stora förändringsvärden än för små eftersom det finns färre möjliga stora förändringsvärden än små. Det gör att alla någorlunda normalfördelade datatyper med oberoende sampel kommer att ha en stor övervikt av små förändringsvärden och därmed modelleras på likartat sätt. Detta kompenseras dock delvis av den höga upplösningen för stora förändringsvärden.

Empiriska försök har visat att algoritmens prestanda försämras märkbart när storleken på fragmenten krymper ner mot 512 B. Antalet stora förändringsvärden sjunker och enstaka förekomster får stort genomslag när modellen skapas. Detta fenomen gäller dock alla algoritmer med hög upplösning.

Användningsområde

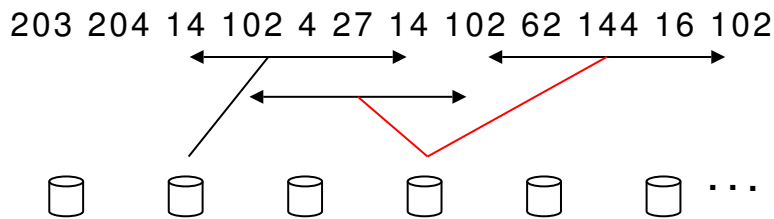
Det främsta användningsområdet för RoC-algoritmen är i kombination med BFD-algoritmen vid genomsökning av lagringsmedia. Fragmentstorleken bör helst ligga på 4 KiB. Mindre fragment går att hantera, men med lägre prestanda och precision. De båda algoritmerna kompletterar varandra väl och har vid empiriska undersökningar tillsammans gett bättre resultat än de enskilda algoritmerna för de flesta typer av data som använts.

Exakt hur algoritmen beter sig i kombination med BFD-algoritmen och för små fragment har inte utvärderats i detalj, men att döma av övriga resultat så borde RoC och BFD-kombinationen fungera väl även för små fragment. Det gör att vi inte avråder från användning av RoC-algoritmen som en del i ett intrångsdetekteringssystem för nätverkstrafik med små fragmentstorlekar, bara algoritmen kompletteras av BFD-algoritmen.

2.3.3 Bytevärdesrepetition

Algoritmen för bytevärdesrepetition (BR) togs fram för att testa om differensen mellan två bytar med samma värde var en egenskap som gick att använda för identifiering av datatypen hos datafragment. Den var främst tänkt att kunna användas i kombination med BFD och RoC. Algoritmen var dock en besvikelse och gav inte alls de förväntade resultaten. Några av orsakerna till det är att

- i datatyper med hög entropi blir olika statistiska mått mer eller mindre lika för alla bytevärden,
- för små fragment (512 B) blir dataunderlaget för litet och
- principen är för lik BFD, men saknar den algoritmens fördelar.



Figur 2.5: Principen för bytevärdesalgoritmen

Det är möjligt att BR-algoritmen fungerar bra för textbaserad data där det finns en tydlig relation mellan bytevärdena (bokstäverna). Denna typ av data faller dock utanför ramen för projektets omfattning och har därför inte testats. Algoritmens dåliga prestanda gör att det inte har skapats något histogram för den.

I Figur 2.5 visas principen för funktionen hos BR-algoritmen. Det framgår tydligt att den har en låg nyttjandegrad på den data som är tillgänglig, något som kraftigt påverkar algoritmens prestanda. En ökning av storleken på fragmenten skulle kunna ha en positiv inverkan på prestandan, men i och med den icke-optimala nyttjandegraden är det inte troligt att det räcker för att prestandan ska nå upp till de andra algoritmernas resultat vid motsvarande fragmentstorlekar.

Fördelar

Vid utvärderingen av BR-algoritmen framkom inte några praktiska fördelar. Teoretiskt sett skulle bra resultat kunna uppnås för stora fragment av välstrukturerad data, till exempel text. Det är också möjligt att BR-algoritmen skulle kunna öka den totala detektionsgraden hos en kombination av BFD-, RoC- och BR-algoritmerna. Det är dock tveksamt om denna ökning skulle vara mer än marginell.

Nackdelar

En av nackdelarna med BR-algoritmen är att den bara kommer till sin rätt i några få specialfall där avstånden mellan bytar med lika värde är korta och homogena. För alla andra datatyper utnyttjar algoritmen dataunderlaget för dåligt för att den ska vara tillämpbar.

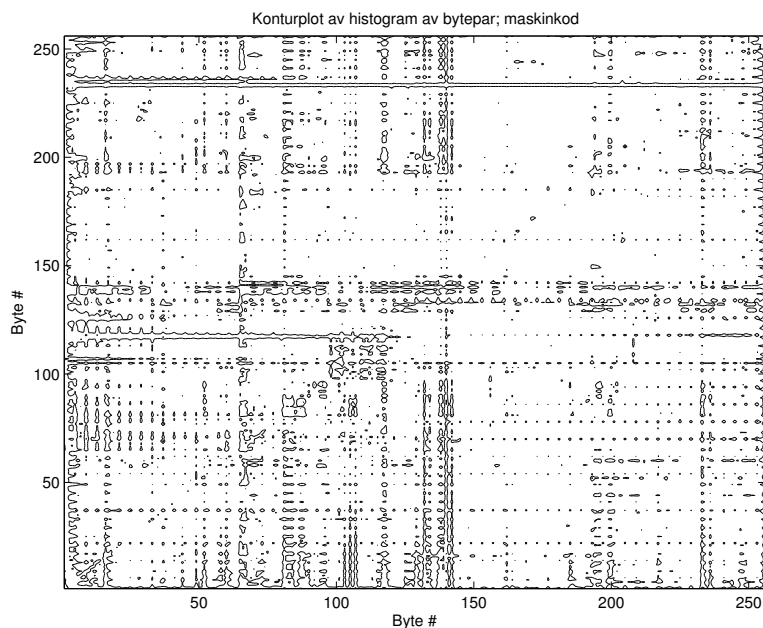
Användningsområde

I praktiken har algoritmen inte något användningsområde. Teoretiskt sett skulle den kunna öka den sammantagna detektionsgraden om den kombinerades med framför allt BFD- och RoC-algoritmen.

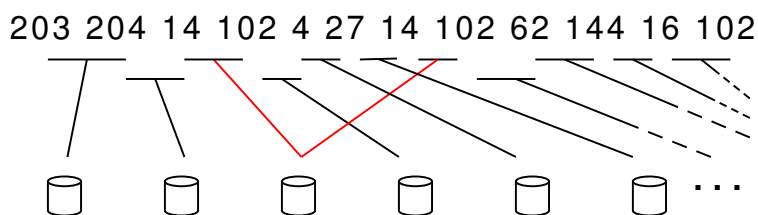
2.3.4 Bytepar

Byteparsalgoritmen använder bytepar för att skapa datatypsmodeller. Det gör att detaljeringsgraden stiger kvadratisk, men det gör även mängden RAM-minnet som krävs och dessutom sjunker exekveringshastigheten i motsvarande grad.

Den högre precisionen hos byteparsalgoritmen gör att det ställs högre krav på kvaliteten hos de data som används för att skapa datatypsmodeller. För att fullt ut utnyttja den prestanda som byteparsalgoritmen har måste datamängden vara större än för enbytesalgoritmerna. Datamängden måste också innehålla alla bytepar som är möjliga enligt datatypens standardformat. Om inte så är fallet kommer det statistiska underlaget bli för dåligt och resultaten otillförlitliga.



Figur 2.6: En konturplott av ett histogram för maskinkod skapad med byteparsalgoritmen.



Figur 2.7: Principen för byteparsalgoritmen

Figur 2.6 visar en konturplott av ett 2-dimensionellt histogram av maskinkod skapat med hjälp av byteparsalgoritmen. Det går att urskilja till exempel hoppinstruktioner och nollutfyllnad av kodblock. Ett exempel är den kvadrat som syns vid bytepar där både bytevärdena ligger mellan 100 och 120. Kvadraten representerar gemenskaperna, det vill säga de små bokstäverna i alfabetet. Det är textsträngar som ligger inbäddade i maskinkoden. Det horisontella strecket vid bytevärde 233 på Y-axeln motsvara en hoppinstruktion.

Byteparsalgoritmen bygger på samma princip som BFD-algoritmen, det vill säga den skapar ett histogram över fördelningen av en vald enhet, i det här fallet bytepar. Figur 2.7 visar en förenklad modell av principen.

Algoritmen flyttar beräkningsfönstret en byte i taget, vilket gör att de behandlade byteparen länkas samman så att även strukturen hos dataströmmen tas i beaktande. På det här sättet är det möjligt att extrahera ofta förekommande längre sekvenser ur en datatypsmodell. Vi har till exempel kunnat verifiera flera formatspecifika sekvenser i mp3-kodad data. Detsamma gäller för JPEG-data.

Algoritmen har en funktion som gör den mer lämpad än de andra algoritmerna vi utvecklat för datatyper som inte utnyttjar hela det tillgängliga bytevärdesområdet. Det kan till exempel röra sig om bytesekvenser som inte är tillåtna enligt standarden för datatypen. Skulle ett otillåtet bytepar dyka upp i ett fragment av okänd datatyp underkänns det direkt av algoritmen. Det här ställer dock krav på mängden data att träna på.

Eftersom det finns 256 gånger fler bytepar än bytar så bör datamängden som datatyp-modellen skapas från för byteparsalgoritmen vara mycket större än för BFD- och RoC-algoritmerna.

Om standarden som specificerar datatypen inte utesluter några bytesekvenser, måste datamängden som modellen skapas från innehålla alla 65536 bytepar som finns. Om inte så är fallet kommer algoritmen att felaktigt identifiera ett korrekt fragment som varandes av en annan datatyp, det vill säga ge ett falsk negativt resultat. Är det så att algoritmen används för att detektera maskinkod och därmed eventuellt skadlig kod kommer algoritmen följaktligen att släppa igenom något som borde stoppats.

Fördelar

En fördel med byteparsalgoritmen är dess mycket höga precision som möjliggör säker identifiering med mycket få falsklarm av de flesta datatyper. Prestandan är något lägre för datatyper som har en mycket jämn bytevärdesfördelning.

Byteparsalgoritmen utnyttjar den tillgängliga datamängden på ett effektivt sätt och återger både bytevärdesfördelning och struktur hos den modellerade datatypen. Det är möjligt att använda algoritmen för att hitta datatypsspecifika egenskaper i form av mycket vanliga eller ovanliga bytesekvenser. Dessa kan sedan användas som karaktäristiska kännetecken för datatypen vid andra former av datatypsidentifiering.

Nackdelar

Algoritmen kräver 256 gånger mer minne än BFD- och RoC-algoritmerna. Likaså exekverar den 256 gånger långsammare än enbytesalgoritmerna.

Kraven på den data som används för att skapa datatypsmodeller är högre för byteparsalgoritmen än för BFD- och RoC-algoritmerna. Dels behövs en större mängd data och alla giltiga bytepar för datatypen måste kunna återfinnas i datamängden. Det finns annars en risk att algoritmen ger upphov till falska negativ och alltså inte hittar alla fragment av en eftersökt datatyp.

Användningsområde

Identifiering av multimedidata är en av byteparsalgorithms starka sidor. Den visar även goda resultat för maskinkod, men på grund av för lite data att skapa modeller av har inte detta kunnat verifieras ordentligt. Vid träning på 63 MiB maskinkodsdata och med 0,0067 % falsklarm presterar BFD-algoritmen en detektionsgrad på 95,7 % och byteparsalgoritmen 89,9 %. Det beror på att den första och andra halvan av träningsdatamängden har något olika uppsättningar av bytepar. Vissa av de bytepar som saknas i den ena halvan återfinns i den andra och ger då upphov till att hela fragment underkänns av byteparsalgoritmen. På enbytesnivå märks inte detta likaväl och BFD-algoritmen hanterar därför situationen bättre än byteparsalgoritmen.

Den något långsammare exekveringen och höga precisionen gör att algoritmen lämpar sig bättre för tillämpningar där tidsåtgången spelar mindre roll. Det kan till exempel vara identifiering av datatyper på lagringsmedia, dataräddning eller stickprovskontroll av nätverkstrafik. Det kan även handla om genomsökning i efterhand av nätverkstrafik-dumpar vid utredning av intrång.

2.3.5 Manhattanavstånd

Manhattanavståndet är en mycket enkel metrik. Det populärvetenskapliga namnet kommer sig av att metriken motsvarar den kortaste sträcka en taxichaufför kan köra mellan två punkter på Manhattan med sitt rutmönstrade gatunät. Kortfattat så måste chauffören sicksacka sig fram längs gatorna eftersom han inte kan åka igenom eller över kvarteren.

Fördelar

Det är mycket enkelt att mäta Manhattanavstånd och det går därför fort att beräkna. Det gör det lämpligt för tillämpningar där hastighet är a och o och precision inte är lika viktigt.

Nackdelar

Ett problem med Manhattanavståndet är den på grund av sitt linjära beteende har svårt att skilja på närliggande punkter i ett mångdimensionellt rum. Det här gör att den här metriken inte lämpar sig för mer än grova uppskattningar av avståndet mellan ett fragment och en modell. Vi har använt den tillsammans med RoC-algoritmen i kombination med BFD-algoritmen och resultatet har aldrig varit särskilt bra.

Användningsområde

Manhattanavståndet har ett begränsat användningsområde i och med att det har så dålig precision. Vi använde det vid våra undersökningar för att ha en referens att jämföra de andra metrikerna mot. Resultaten var så nedslående att metriken övergavs. Det har därför inte gjorts några tester med hur byteparsalgoritmen fungerar ihop med Manhattanavståndsmetriken.

Det är tänkbart att Manhattanavståndet skulle kunna användas för datatyper med mycket specifika egenskaper, som är lätta att skilja från andra datatyper. Vi kan tyvärr inte ge något exempel, utan det får avgöras från fall till fall.

2.3.6 Viktad kvadratsumma

Metriken som bygger på viktad kvadratsumma är en vidareutveckling av det förenklade Mahalanobis avståndet⁸ som är mycket vanligt förekommande hos andra forskargrupper inom datatypsidentifieringsområdet. Tanken bakom att använda en kvadratsumma är att många små differenser mellan ett fragment av okänd datatyp och en datatypsmodell ska premieras framför några få stora differenser. Detta för att stora avvikelser, om än få, ofta pekar på att fragmentet och modellen inte är av samma datatyp.

Viktningen av kvadratsumman görs med hjälp av standardavvikelsemättet hos datatypen. Om standardavvikelsen för en egenskaps värdefördelning är stor innebär det att värdena varierar mycket och det medelvärde som används för jämförelse mellan modell och fragment av okänd datatyp är inte så tillförlitligt. Därför minskar vi just det jämförelsevärdets betydelse för det totala avståndet.

Den viktade kvadratsumman är relativt lättberäknad, men ger ändå möjlighet att ta hänsyn till individuella avvikelser i tillförlitligheten hos modellens ingående värden. Värt att notera är att ju lägre standardavvikelsen är desto bättre, med idealfallet 0 för helt statistiska värdemängder. Det vill säga om ett specifikt egenskapsvärde inte varierar när modellen skapas kommer det att få en standardavvikelse på 0, vilket kan användas för att höja precisionen i datatypsidentifieringsalgoritmen.

⁸Den viktade kvadratsummemetriken kan även ses som en vidareutveckling av det vanliga Euklidiska avståndsmättet.

Den matematiska formuleringen av den viktade kvadratsummemetriken visas i Ekvation 2.1. Det som mäts är differensen mellan datatypsmodellens n medelvärden c_i i vektorn $\vec{c}$ och det fragmentets frekvensfördelningsvärden s_i i vektorn $\vec{s}$. Standardavvikelsen hos datatypsmodellens frekvensfördelningsvärden representeras av σ_i . För att undvika division med noll så används utjämningsfaktorn $\alpha = 0.000001$. Ekvationen lyder

$$d(\vec{s}, \vec{c}) = \sum_{i=1}^n \frac{(s_i - c_i)^2}{(\sigma_i + \alpha)}. \quad (2.1)$$

Vid tester med den viktade kvadratsummemetriken har den presterat bra. Allra bäst fungerar den ihop med byteparsalgoritmen. En av anledningarna är metriken förmåga till selektion av vilka egenskaper som ska användas. När denna förmåga paras ihop med byteparsalgoritmens höga precision blir resultatet mycket bra. Dessutom innebär den viktade kvadratsummemetriken en god kompromiss mellan detaljerad mätning och låga krav på minnesmängd, vilket framför allt är viktigt vid användning av byteparsalgoritmen.

Fördelar

En av fördelarna med den viktade kvadratsummemetriken är att den erbjuder en bra kompromiss mellan avancerad mätning med möjlighet till viktning av de ingående värdena och att den premierar resultat med många små differenser framför några få stora. Metriken kräver endast ett minnesutrymme motsvarande 2 gånger antalet bytevärden eller bytepar, vilket gör den lämplig för algoritmer som genererar stora mängder differensvärden.

Nackdelar

Metriken fungerar sämre för datatyper med en bytevärdesfördelning som ger upphov till stora standardavvikelser. Det får till följd att alla differensers betydelse minskas och avståndsmåttet blir instabilt.

Viktningen av kvadratsummorna ställer också krav på att de datamängder som används för att skapa datatypsmodeller är tillräckligt stora och har hög kvalitet, det vill säga inte innehåller andra datatyper eller felaktiga, för datatypen ej tillåtna, bytevärden.

Användningsområde

Den viktade kvadratsummemetriken är en generell metrik som har gett bra resultat för en stor mängd datatyper. Den är speciellt lämpad för byteparsalgoritmen men har även presterat bra tillsammans med BFD-algoritmen för datatyper med ojämn bytefördelning.

2.3.7 Sannolikhet

Den sannolikhetsbaserade metriken vi använt baseras på bytevärdena eller byteparens frekvenser relativt den valda fragmentstorleken. Sedan beräknas $\log_{10}$ för dessa värden och resultatet kvadreras för att göra alla värden positiva, men också för att förstärka effekten av låga sannolikhetsvärden. I praktiken är det alltså inte de riktiga sannolikhetsvärdena som används för avståndsberäkningarna.

Avståndet mellan ett fragment av okänd typ och en datatypsmodell beräknas i vår implementation av algoritmen genom uppslagning och multiplikation av varje bytevärde i fragmentet. I och med att en logaritmisk representation av sannolikhetsvärdena lagras

i modellen kan en addition utföras i stället för en multiplikation, vilket gör avståndsberäkningen snabbare. I det här fallet kommer ett högt värde att representera en liten avstånd mellan modell och fragment.

Genom att den här metriken kan implementeras så att den enbart göra pekning i minnet och sedan addition blir den mycket snabb. Användningen av en extra dimension gör dock att den kräver stora mängder minne och därför lämpar sig sämre för vissa minneskrävande algoritmer. Kraven på minnesutrymme gör också att metriken bara går att använda för maximalt 512 B stora fragment.

Metriken och den typ av modell som används utgör ett förstadium till typexemplaralgoritmen, men här görs ingen åtskillnad mellan de olika histogram som lagras i modellen. Det finns inte någon koppling mellan minnesuppslagningarna, så värden från helt olika histogram kan ingå i ett avståndsvärde.

Fördelar

Sannolikhetsmetriken är mycket snabb och enkelt uppbyggd. Den tar hänsyn till det faktum att de flesta datatyper inte ger några jämna och symmetriska bytevärdesfördelningar. Istället har de ofta flera toppar utspridda över bytevärdesintervallet. Dessa toppar lagras i datatypsmodellen och kan sedan utnyttjas för identifiering med högre noggrannhet än för den viktade kvadratsumman.

Nackdelar

En av nackdelarna med sannolikhetsmetriken är att när beräkningen av avståndet mellan ett fragment av okänd datatyp och en datatypsmodell görs kan värden från flera olika bytevärdesfördelningar användas tillsammans. Detta faktum sänker precisionen i identifieringen, men vi anser att den enkla principen bakom beräkningen till viss del uppväger denna nackdel.

De krav på minnesutrymme som metriken ställer gör att den inte kan användas i tillämpningar där större fragment än 512 B hanteras. Likaså begränsar minneskraven urvalet av identifieringsalgoritmer till sådana med låga krav på minnesutrymme.

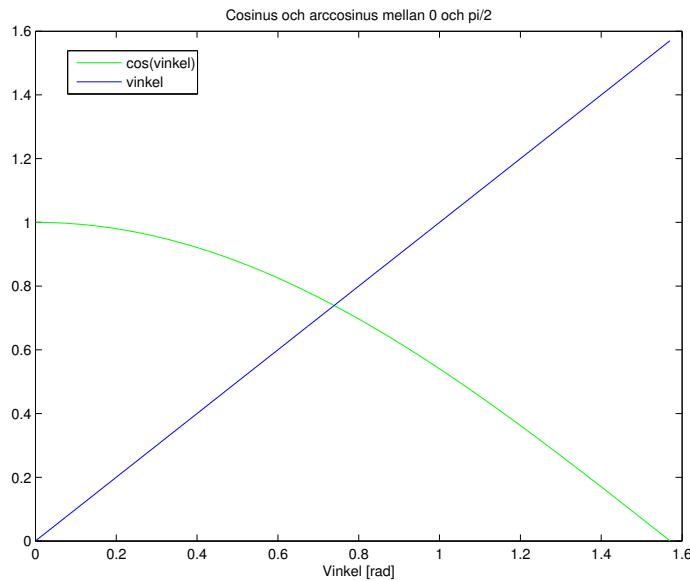
Användningsområde

Den här metriken passar bäst för algoritmer som har lägre krav på minnesutrymme och som kan hantera små fragment. Den har visat sig förbättra detektionsgraden betydligt för framför allt BFD-algoritmen. För BFD-algoritmen är resultaten för identifiering av maskinkod är bättre än för byteparsalgoritmen. Även för andra filtyper är resultaten avsevärt bättre än när den viktade kvadratsummemetriken används.

2.3.8 Vinkel mellan vektorer

Flera forskargrupper, däribland vår, har oberoende av varandra varit inne på att använda vinkeln mellan vektorrepresentationen av en datatypsmodell och ett fragment av okänd typ som mått på avståndet mellan dem. Det finns två varianter av vektorvinkelmetriken, antingen används cosinusvärdet som kortfattat motsvarar förhållandet mellan två vektorer, eller så används vinkeln mellan två vektorer. Cosinus av vinkeln α mellan två vektorer $\vec{x}$ och $\vec{y}$ beräknas genom

$$\cos(\alpha) = \frac{\vec{x} \cdot \vec{y}}{|\vec{x}| |\vec{y}|}.$$



Figur 2.8: Vinkeln och cosinus beräknat för vinkeln $\alpha = [0, \pi/2]$ (0 till 90 grader).

Vinkeln α erhålls följaktligen genom

$$\alpha = \arccos \left(\frac{\vec{x} \cdot \vec{y}}{|\vec{x}| |\vec{y}|} \right). \quad (2.2)$$

Cosinus av α är lättare att beräkna, men precisionen varierar. Vinkeln däremot har ett linjärt beteende, men är mer krävande att beräkna. I Figur 2.8 syns detta tydligt.

Figur 2.8 visar att för små vinklar minskar cosinusvärdet långsamt, för att accelerera och öka som mest för stora vinklar. Det här är ett oönskat beteende om differensen mellan två vektorer ska beräknas relativt en gemensam referenspunkt. Anledningen är att det är svårt att tolka skillnaden i avstånd mellan vektorerna och referenspunkten när avståndet inte ökar linjärt.

Det linjära beteendet hos vinkeln gör att den är bättre lämpad för våra tillämpningar än cosinusvärdet av vinkeln. Förändringstakten för vinkeln är konstant lika hög som den högsta för cosinus, vilket gör att upplösningen på differensvärdena alltid är bättre eller lika bra som för cosinus. Linjäriteten gör också att det går att jämföra avstånd mellan olika vektorer på ett enkelt sätt.

Fördelar

En fördel med vektorvinkelmetriken är att den går att förstå intuitivt. Om vinkeln används får metriken dessutom ett linjärt beteende, vilket är önskvärt.

Nackdelar

I och med att Manhattanavståndet av en modellvektor alltid är detsamma för lika stora fragment och samma algoritm begränsas urvalet av möjliga avstånd. Det gör att sannolikheten för att två helt olika fragment ska få samma vinkel till en modellvektor är så stor att det utgör ett problem. Vinkelmåttet innehåller en mindre andel relevant information om förhållandet mellan ett fragment av okänd datatyp och den datatypsmodell det jämförs med, än både kvadratsummemetriken och sannolikhetsmetriken.

En annan nackdel med vektorvinkelmetriken är att den lider av problem med hög dimensionalitet. Hur många element två vektorer än har så ligger värdet för vinkeln mellan dem inom intervallet 0 och $\pi/2$ mätt i radianer. Det gör att trots att vinkeln används så minskar effekten av differenserna mellan olika fragment och jämförelse av avståndet mellan dem blir svårare på grund av begränsningar i datorers flyttalsprecision.

Användningsområde

Andra forskargrupper inom datatypsidentifieringsområdet rapporterar att vektorvinkelmetriken är mycket bra att använda när avkortade bytefördelningsvektorer⁹ används för datatypsidentifiering. Detta är logiskt eftersom effekten av differenser mellan två vektorer ökar när vektorerna blir kortare, medan till exempel den viktade kvadratsummetriken drabbas av ett minskat informationsinnehåll och därmed sämre tillförlitlighet vid sådana förhållanden.

Vektorvinkelmetriken har också ett användningsområde som en del i den typexemplarmetrik som presenteras i Kap. 2.3.9.

2.3.9 Typexemplar

Metriken för typexemplar finns inte implementerad, utan bara som ett skissartat utkast. Vi tror dock tillräckligt mycket på dess potential för att presentera huvuddragen bakom principen. Inspiration har hämtats från andra forskargrupperns arbete (se mer i Kap. 2.6.1), som sedan kombinerats med idéerna bakom våra metriker.

I grova drag har vi tänkt oss att en modell innehållande flera typexemplars histogram skapas av en datatyp på liknande sätt som för sannolikhetsmetriken (se Kap. 2.3.7). Antalet histogram får gärna vara stort, men det kräver då ett kompakt lagringsformat, helst kopplat till en snabbuppslagningsfunktion i form av en hashtabell eller liknande.

Allra helst ska hashfunktionen vara lite ”luddig”¹⁰, det vill säga att den ska acceptera nära missar som träff. Det skulle göra att även om inte alla teoretiskt möjliga histogram och deras sannolikhet återfinns i datatypsmodellen, så ska hashfunktionen kunna samla nästan lika fragments histogram i samma grupp, även om de inte ger en exakt träff bland datatypens typexemplar.

Ett exempel på en tänkbar hashfunktion är kombinera metriken för vinkel mellan vektorer (se Kap. 2.3.8 och Ekv. 2.2) med en beräkning av masscentrum $\bar{x}$, där x_i anger position och m_i värde i datatypsmodellens vektor, enligt följande ekvation

$$\bar{x} = \frac{1}{m} \sum_i m_i x_i; \quad m = \sum_i m_i. \quad (2.3)$$

Först mäts vinkeln mellan histogrammet för ett datafragment av okänd typ och till exempel enhetsvektorn. Sedan beräknas masscentrum för histogrammet i enlighet med Ekv. 2.3. Tillslut beräknas produkten av de två värdena och den utgör sedan histogrammets hashvärde. Alternativt kan de båda värdena sparas i en 2-tupel¹¹ som får representera ett hashvärde för histogrammet.

⁹De sorterar elementen i vektorerna och använder bara en delmängd (de x största) för sina avståndsberäkningar.

¹⁰Fritt översatt från engelskans *fuzzy (logic)*.

¹¹En två enheter lång lista, som betraktas som en odelbar enhet. Den betecknas ofta som $\langle x, y \rangle$.

Avståndet mellan ett fragment av okänd datatyp och modellen får sedan utgöras av skillnaden mellan deras hashvärden. Värdena beräknas med lämplig avrundning för att uppnå vald ”luddighet”.

Eftersom vi inte har utvärderat typexemplarsmetriken ännu kan vi bara uttala oss hypotetiskt om dess för- och nackdelar. Detsamma gäller för metriken användningsområde.

Fördelar

Metriken fördelar är att den tillåter en högre precision hos modeller av heterogena datatyper genom att faktiska exempel på histogram för den aktuella datatypen används vid identifiering av okända data. På så sätt undviks den inbyggda osäkerhet som statistiska mått alltid har. Samtidigt medger den ”luddiga” hashfunktionen att metriken inte blir för exakt och ger upphov till en mängd falska negativ.

Vår förhoppning är att metriken ska tillåta oss att plocka de bästa bitarna från de andra metriker vi utvecklat och att samtidigt slippa deras nackdelar.

Nackdelar

En nackdel som vi ser med typexemplarsmetriken är att dess behov av minnesutrymme kan bli stort. Det kan leda till att antalet histogram som sparas i modellen måste begränsas, med följd att hashfunktionens ”luddighet” måste ökas. Det i sin tur gör att precisionen sjunker hos metriken och vi närmar oss prestandan hos övriga metriker som utvecklats inom projektet.

Användningsområde

Användningsområdet för metriken är i stort sett detsamma som för sannolikhetsmetriken och vektorvinkelmetriken. Arbetshypotesen är att den ska gå att använda ihop med alla datatypsidentifieringsalgoritmer och höja deras prestanda. Typexemplarsmetriken är också tänkt att vara snabb och förhoppningsvis medge användning för till exempel intrångsdetektionstillämpningar och kontroll av nätverkstrafik.

2.3.10 Sätta samman datafragment

Metoden för att sätta samman datafragment utvecklades för att i möjligaste mån kunna erbjuda sätt att återskapa filer från de data som identifierats med hjälp av BFD-, RoC- och byteparsalgoritmen. Principen för metoden bygger på att hitta och utnyttja periodiserade egenskaper för datatypen. För att exemplifiera metoden valdes JPEG-bilder med Restart Markers¹².

Ett av målen vid metodutvecklingen var att undvika beroende av formatspecifik avkodning. Utvecklingen utgick därför från ett ”underifrån och upp”-perspektiv där varje teknisk nivå i datatypens standard utvärderades utifrån dess informationsvärde för sammansättning. För JPEG-data uppnåddes tillräckligt innehåll först efter att en enkel form av avkodning av data gjorts till en grov svartvit bild.

Principen bygger på att vertikala linjer i en JPEG-bild avspeglas i dess svartvita representation. Genom att mäta avståndet mellan dessa repetitioner kan datafragmenten sättas samman med hög tillförlitlighet.

¹²Restart Markers används för att kunna återstarta avkodningen av JPEG-data vid fel i dataströmmen. De uppträder med ojämna intervall i dataströmmen, men ger tillräckligt med information för att vi ska kunna sätta samman fragment.

Fördelar

Den grundläggande principen för metoden är anpassningsbar och generellt applicerbar. Metoden är robust och klarar av att avståndet mellan två repetitioner av en vertikal linje varierar om förändringen är tillräckligt liten. Vad som är tillräckligt litet bestäms av egenskaperna hos den bild som återskapas. Metoden kan hantera fragment från olika filer av samma datatyp via ett iterativt arbetsätt.

Nackdelar

Metoden har inte gått att generalisera helt och kräver därför datatypsspecifika anpassningar för att fungera. I fallet med JPEG-data krävdes det att dataströmmen avkodades i enlighet med JPEG-standarden för att sammansättning skulle vara möjlig.

Eftersom metoden bygger på att utnyttja repetition i data måste fragmenten vara så stora att ett repetitionsintervall får plats. I teorin kan det gå att leta sig fram till rätt intervall även om fragmenten är små, men det kräver att antalet fragment är förhållandevis litet och att egenskapen som repeteras inte lätt kan förväxlas med någon annan.

Beroende på datatyp kan viss förfiltrering av indata behövas, utöver det att det inte får förekomma några andra datatyper bland de fragment som ska sättas samman. Metoden kan inte användas för tillämpningar där det finns krav på realtidsnära exekvering. Beroende på hur stor mängden data som ska behandlas är kan exekveringen ta upp till många timmar.

Användningsområde

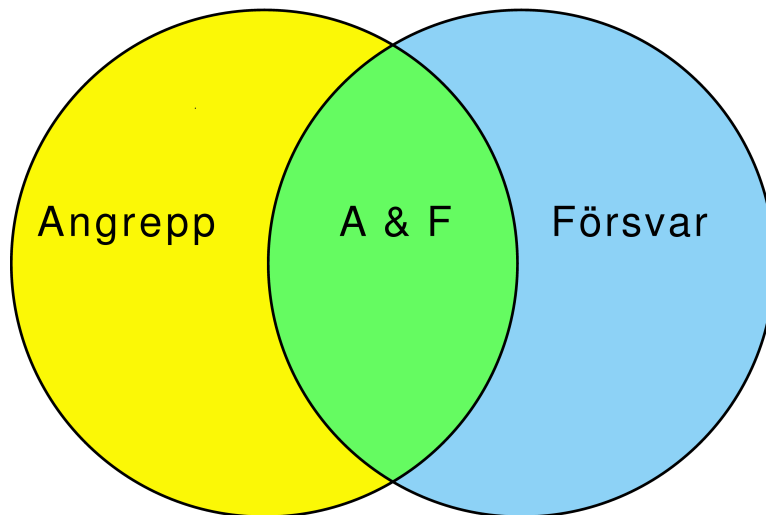
Användningsområdena är desamma som för datatypsidentifieringsalgoritmerna, men den här metoden är tänkt att användas i mer speciella fall där det finns önskemål om att datafragment som hittats ska länkas ihop för att på så sätt kunna ge mer information. De mest självklara användningsområdena återfinns inom dataräddning, forensiska undersökningar och nätverksövervakning där det inte finns krav på realtidsexekvering.

2.4 Gemensam lägesbild

Begränsade svar på frågeställningen om gemensam lägesbild ges genom observation och mätning av övningsdeltagarnas uppfattning av läget. Praktiska resultat kommer från övningen som hålls 2009-12-10 (se Kap. 2.5). Expertis ur kompetensklossen Människa-Systeminteraktion från enheten för Systeminteraktion vid FOI medverkar i projektet för att studera hur en gemensam lägesbild byggs upp för ett för övningsdeltagarna tidigare okänt datornätverks status. De primär tekniska förberedelserna för övningen presenteras närmare i Kap. 2.5.

I och med samarrangemanget med projektet *Säkerhet i digitala kontrollsystem (SCADA)* omfattar övningen på en övergripande nivå IT-säkerhet i SCADA-system. Scenariot som används för övningen är enkelt och utgår från ett för deltagarna okänt system av datornätverk med flera olika användningsområden. De är sammankopplade via olika säkerhetslösningar. I och med samarrangemanget med SCADA-projektet är flera nätverk kopplade till symboliska fabriker som ska slås ut på olika sätt. Dessa nätverk hör i sin tur ihop med olika företagsnät. Övningsdeltagarna har alla olika bakgrund och har inte fått någon möjlighet till gemensamma förberedelser.

Den heterogena sammansättningen av gruppen av övningsdeltagare gör att det finns stora möjligheter att studera hur en gemensam lägesbild skapas från grunden och vilken kvalitet den då får. Övningsscenariot korrelerar till verkliga situationer där en grupp IT-säkerhetsexperten snabbt sammankallas för att försöka få överblick över ett system



Figur 2.9: Det finns tre typer av IT-övningar. Rena angreppsövningar, rena försvarsövningar och en kombination av dessa.

av sammankopplade nätverk som är helt okänt för dem. Där måste de skapa sig en bild av de svagheter som nätverken innehåller, för att på så sätt sedan kunna gå vidare med nästa steg, vare sig det innebär offensiva eller defensiva åtgärder.

Ytterligare en målsättning med övningen är att försöka besvara frågan om vad en gemensam lägesbild av ett nätverks status innebär.

- Vad ställs för krav på innehållet i lägesbilden?
- Vilka möjligheter till förbättringar finns?
- Vad bidrar till att förenkla sammanställningen av nätverkets status?
- Vilka försvårande omständigheter finns det och
- hur övervinns de i så fall?

Resultaten från övningen kommer att redovisas i ett FOI-Memo så snart utvärderingen av övningen har slutförts.

2.5 IT-försvarsövning

IT-krigslabbet har utvecklats inom projektet och under 2008 användes det för en IT-försvarsövning. Denna vidareutvecklas och anpassas till rådande behov inom projektet. Erfarenheter från användningen av IT-krigslabbet i övningen 2008 är en av utgångspunkterna för en övning som planeras hållas 2009-12-10.

IT-säkerhetsövningar kan delas in i tre grupper utifrån vilket upplägg som används (se Figur 2.9). Den vanligaste typen är kombinerade övningar där två eller flera grupper både angriper andra övningsdeltagares system och försvarar sina egna system. Dylika övningar är vanliga inom hackervärlden och kallas vanligtvis för flaggerövring¹³.

Näst vanligast är övningar där övningsdeltagarna får försvara sig mot angrepp från ett professionellt lag av IT-säkerhetsspecialister. Denna typ av övning är vanlig inom militär verksamhet och syftar ofta till att utbilda eller examinera studenter med IT-säkerhetsinriktning vid försvarshögskolor. FOI arrangerade en sådan övning inom SCADA-projektet 2009 och den estländska, internationella IT-säkerhetsövning som FOI var medarrangör till 2008 var av denna typ.

¹³Capture the flag på engelska.

Den tredje typen av övning, där övningsdeltagarna enbart använder offensiva tekniker, är inte lika vanlig eftersom deltagarunderlaget är begränsat¹⁴. Detta är den typ av övning som kommer att hållas 2009-12-10. Huvudsyftet med övningen är att studera hur en gemensam lägesbild av ett systems eller nätverks status växer fram under så fria och realistiska förutsättningar som möjligt.

Även om konceptet med att öva angrepp på IT-system vid en första anblick kan verka fel, är det nödvändigt att öva även detta för att bättre kunna förstå hur en angripare tänker. Det gör att personal som ska försvara IT-system bättre kan förutse en angripares nästa drag och därmed har större chans att framgångsrikt försvara det angripna systemet.

2.5.1 Övningens upplägg

Övningen 2009 avser att erbjuda möjligheter för praktisk IT-säkerhetsträning riktad till teknisk kvalificerad personal. Praktisk, laborativ träning av explorativ art kan ge helt andra kunskaper än vad teoretiska resonemang kan. Ett vanligt sätt att arrangera IT-säkerhetsövningar är med ett angripande rött lag respektive ett försvarande blått lag. I den nu planerade övningen ingår endast ett angripande rött lag, som fokuserar på att kartlägga ett för deltagarna okänt nät. Deltagarna kommer från olika svenska myndigheter och förväntas under övningen

- skapa en gemensam bild av hur ett komplext övningsnät realiserat i ett datorkluster är uppbyggt,
- detektera och identifiera sårbarheter i nätet och
- utnyttja sårbarheter i nätet för angrepp.

Övningen ska även hjälpa till att utvärdera den plattform för IT-säkerhetsövningar med tillhörande spårbarhetssystem som utvecklats inom projektet.

Övningsnätet ingår i ett SCADA-system knutet till en uppsättning i nätet utspridda modellfabriker. När man hittar en fabrik i nätet innebär ett angrepp på denna att en isfackla tänds. I nätet finns även ett antal webbkameror som angriparna kan ta över styrningen av.

Nätets komplexitet och storlek avses vara tillräckligt för att ge en realistisk känsla under övningen. En angripare måste dels kartlägga SCADA-nätet och dels inse att det finns utrustning (modellfabrikerna) som är knutet till nätet. Denna kartläggning utgör en tekniskt orienterad lägesbild av övningsnätet.

Utöver den tekniska lägesbilden studeras ur ett människa-/systeminteraktionsperspektiv hur kommunikationen utvecklar sig under övningen. Exempelvis kan det vara av intresse att studera hur konstellationer i det röda laget bildas och ombildas.

Övningen beaktar inte närmare konsekvenser av angrepp, vilket medför att riskperspektivet strikt taget inte hanteras. Övningen är därmed begränsad till att identifiera sårbarheter i nätet och som hotaktör utnyttja dessa sårbarheter. Kunskap om angrep och angreppsstrategier förutsätts dock kunna användas till att bygga upp praktisk, systematiserad kunskap om hur försvarsstrategier och -mekanismer bör utformas. För att kunna skydda sig aktivt krävs operativ kunskap om hur offensiva aktörer kan tänkas agera. Till detta kan övningen ge viktiga bidrag.

¹⁴ Det är kommersiellt mer gångbart att marknadsföra sig som försvarare av IT-system, än som angripare.

Övningen förväntas indikera vikten av tillgång på nätkunniga tekniker för att effektivt kunna skydda kritiska nät och utrustning knutna till dessa. Huruvida andra kompletterande kompetenser behövs för sådant skydd siktar övningen inte på att besvara. Eventuellt kan det ändå tänkas att resultat av övningen kan indikera sådana behov. Exempelvis vore det intressant att studera vad som bör ingå i operativ kunskap om offensiva aktörers agerande. Räcker det för detta med tekniskt kunnande eller krävs annan kompletterande kunskap från andra domäner?

2.5.2 Försvarsmaktens nytta av IT-försvarsövningar

Tekniskt orienterade IT-försvarsövningar har till stor grad saknats inom Försvarsmakten. Därmed finns det här ett område i behov av vidare forskning, studier och verksamhet. Icke minst blir detta av vikt att studera då datornät och -system på olika sätt i verkliga situationer är knutna till för samhället viktiga resurser. I civila sammanhang kan det exempelvis vara tal om kritiska infrastrukturer styrda av SCADA-system. I militära sammanhang kan det vara tal om lednings- och kommunikationssystem med hantering av sekretessbelagd information, samt kopplingar till vapen- och logistiksystem.

Genom att samma utrustning kan användas för SCADA-relevant träning, såväl som för militärt IT-försvar, existerar tydliga möjligheter för positiva synergivinster. Likheter såväl som olikheter i erfarenheter torde utgöra en värdefull bas för resonemang utgående från SCADA-perspektiv och militärt perspektiv

I situationer som dessa där samhällsviktiga resurser riskerar att utsättas för riktade angrepp av kvalificerade motståndare, krävs ett aktivt försvar med synnerligen god teknisk förmåga att granska och förstå vad som försiggår i berörda datornät och datorsystem. IT-försvarsövningar är nödvändiga för att *bygga upp kunskaper* om offensiv verksamhet, för att bättre och mer effektivt kunna skydda oss. Övningar behövs också för att identifiera *vilken kunskap som behovet är störst av*. För att bedöma detta krävs betydande experimentell verksamhet. Slutligen spelar återkommande övningar en viktig roll som ett sätt att *upprätthålla och vidareutveckla kunskap*, och icke minst *upprätthålla och vidareutveckla nätverken* av experter.

Med denna kunskap kan man generera en lägesbild av hur säkerheten ser ut i ett nätverk. Termerna lägesbild och gemensam lägesbild används med olika innebörd av olika aktörer. Av vikt torde dock vara att samagerande aktörers information överensstämmer, som till exempel för lagmedlemmar i en IT-försvarsövning. Det är nödvändigt att man har samma uppfattning om hur nätet är uppbyggt. Vid val av verktyg och arbetsmetodik är det nödvändigt att dessa bidrar till överensstämmande lägesbilder eller en i någon mening gemensam lägesbild. En gemensam lägesbild behöver dock inte betyda att alla inledningsvis har tillgång till exakt samma information. Det skulle i sämsta fall medföra att man dränks av stora volymer information som i praktiken medför att en gemensam lägesbild inte kommer att kunna skapas inom rimlig tid.

För att generera säkerhetslägesbilden krävs dels den operativa kunskapen om attacker och effekter av attacker, dels kunskap om hur denna kunskap totalt sett kan bidra till en värdering av säkerhetsläget. Här kan man se hur två olika perspektiv på IT-säkerhet i sammanhangen har potential att korsbefrukta varandra.

FM kan nyttja resultaten och erfarenheterna från övningen på flera sätt. Först och främst bidrar övningen till att höja kvaliteten på kommande övningar. Den medför också att tiden och resursåtgången för att förbereda en IT-säkerhetsövning minskar för FM genom att de kan använda sig av tidigare erfarenheter från FOI. Likaså kan studien av processen bakom skapandet av en gemensam lägesbild användas vid utveckling av framtida ledningssystem. Där kan personal från FOI till exempel medverka som experter under

utvecklingsprocessen, något som till viss del sker redan idag. Erfarenheterna från övningen möjliggör dock ett djupare engagemang och en bredare erfarenhet än tidigare.

2.5.3 FOI Reconstruction & EXploration (F-REX) Studio

F-REX Studio är ett verktyg som utvecklats i syfte att användas som ett hjälpverktyg vid övnings- och insatsutvärderingar. Med hjälp av verktyget kan förinspelad data, såsom ljud, video, stillbilder, händelseloggar, skärmdumpar med mera importeras för att sedan spelas upp tidssynkroniserat och presenterat i olika typer av vyer. Inspelning kan ske via till exempel en dators inbyggda webbkamera och ljudkretsar. Med hjälp av loggning av händelser i en dator, antingen via standardloggar eller med specialskrivna program, kan sedan ytterligare detaljeringsgrad erhållas vid sammanställningen i F-REX Studio.

Ett exempel på en vy är direktpresentation av stillbilder i stillbildsvyn. Detta är en relativt enkel representation. F-REX Studio hanterar dock även mer komplicerade vyer i form av statistikvyer, där statistik över exempelvis olika typer av kommunikation kan presenteras.

2.5.4 Övningsförberedelser

Försvar av IT-systemprojektet kommer att hålla en flaggerövringsövning tillsammans med ett annat FOI-projekt. Övningen kommer att gå av stapeln 2009-12-10 som en kvällsaktivitet under en workshop om säkerhet i SCADA-system som FOI arrangerar inom SCADA-projektet. Inbjudningar till övningen har skickats till ett flertal intressenter, däribland FM. För närvarande ligger det maximala antalet deltagare på 16, men om behov uppstår är det möjligt att skala upp delar av övningen så att fler kan få plats.

Övningen är tänkt att testa det mest avancerade spårbarhetssystem som hittills använts under IT-säkerhetsövningar inom FOI. För att möjliggöra detta har F-REX Studio utökats med nya funktioner och det kommer att finnas ljud, video, skärmdumpar och händelseloggning för varje deltagare. Dessutom kommer ett antal observatörer närvara och en elektronisk white board användas av deltagarna för att åskådliggöra lägesbilden momentant.

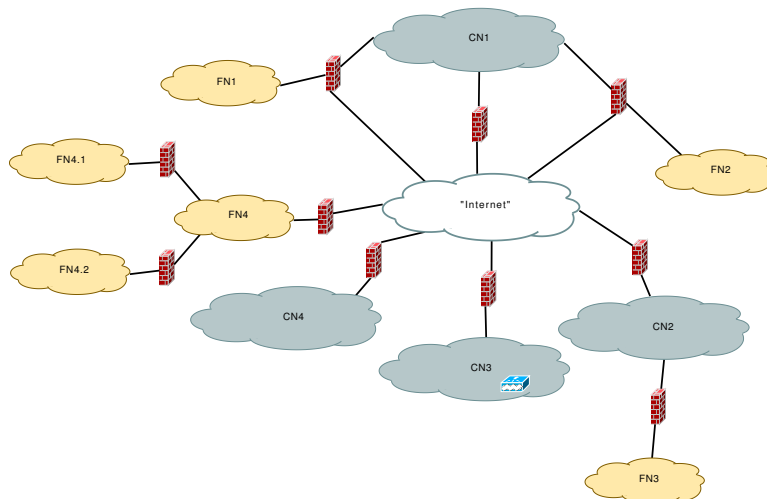
F-REX

Inför övningen har F-REX Studio vidareutvecklats för att kunna importera och presentera videosekvenser av formatet Flash Video (.flv). Det var ett krav för att kunna använda de inbyggda webbkamerorna i de bärbara datorer som övningsdeltagarna ska använda. Dessutom har datorerna preparerats med inspelningsskript för video från såväl den inbyggda webbkameran som skärmdumpar, samt tidssynkronisering mot en extern nätverksansluten tidskälla.

En generalrepetition av tekniken bakom övningen gjordes i samband med ett besök bestående av bland annat IT-säkerhetstekniker från MSB och FRA den 26:e oktober 2009, varpå såväl skripten på de bärbara datorerna som vidareutvecklingen av F-REX fungerade felfritt.

Flaggor att erövra

En skiss över nätverket som övningsdeltagarna ska kartlägga har tagits fram (se Figur 2.10). I figuren syns hur fyra företagsnätverk (CN) samt fem fabriksnätverk (FN) är sammankopplade via ett privat internät. Inför övningen har säkerhetshål i operativsystem undersökts för att förbereda de individuella klienterna i nätverksuppsättningen.



Figur 2.10: Skiss av det internätverk som kommer att användas under IT-säkerhetsövningen 2009-12-10. Förkortningen CN står för företagsnätverk och FN står för fabriksnätverk. Observera att nätverket CN3 innehåller en trådlös (WLAN) router.

Tanken är att deltagarna ska ta sig in via ett trådlöst nätverk (WLAN) i ett av företagsnätverken (CN3) och därifrån ta sig vidare till fabriksnäten via olika säkerhetshål i såväl arbetsstationer som brandväggar i nätverket.

Till vissa säkerhetshål är kopplat elektriskt utlösta isfacklor eller webbkameror. När ett sådant säkerhetshål framgångsrikt utnyttjas tänds en isfackla som symboliserar att en fabrik sprängs. Webbkamerorna ska användas för att till exempel filma ett lösenord. Detta kan sedan användas för att ytterligare eskalera behörigheten i systemet.

Deltagarna får inte veta det exakta antalet säkerhetshål som implementerats i övnings-systemen. Genom att metodisk och gemensamt skaffa sig en överblick av det virtuella internätet kommer övningsdeltagarna att visa hur de skapar sig en gemensam lägesbild. I och med den höga nivå som det utbyggda spårbarhetssystemet ger kommer det att gå att återskapa enskilda moment i övningen. Med hjälp av F-REX Studio som stöd kan sedan övningsdeltagarna få återge sina tankar och idéer bakom någon särskild situation. På så sätt går det att noggrant beskriva processen som leder fram till en gemensam lägesbild.

2.6 Omvärldsanalys

Verksamheten inom Försvar av IT-system kan delas upp i två områden, förbättrad intrångsdetektering genom identifiering av datatyp hos datafragment, samt utveckling av övningar i försvar av IT-system. Dessa två områden presenteras under var sin rubrik och kan verka vitt skilda, men har i praktiken mycket nära kopplingar till varandra. Den främsta gemensamma faktorn är att båda verksamheterna handlar om att förbättra möjligheterna till skydd av IT-system. Övningarna kan fungera som testbänk för metoderna för identifiering av datatyp och genom att tillämpa teorierna bakom identifieringsmetoderna kan övningsdeltagarna lära sig mer om de underliggande premisserna för intrångsdetektering.

2.6.1 Datatypsidentifiering

Det finns ett flertal internationella forskargrupper med liknande inriktning som projektet Försvar av IT-system. Det som skiljer grupperna åt är delvis vilket väg de valt mot

målet, men även målet i sig. Verksamheten inom Försvar av IT-system har varit inriktad på att utveckla enkla och robusta metoder för kategorisering av typ hos fragment av data, till exempel nätverkspaket. Störst vikt har lagts på att kunna hitta exekverbar kod (maskinkod) och JPEG-bilddata, med en generell begränsning till datatyper med hög entropi, det vill säga av en människa icke direkt läsbara.

Övriga forskargrupperns fokus ligger istället på text och bilder, några få gånger har exekverbar kod ingått i deras experiment och då endast för jämförelse skull. Vikten av kravet på att metoderna ska hantera datafragment utan tillgång till metainformation¹⁵ har nu år 2009 börjat diskuteras på en vidare basis inom forskningsområdet. FOI pekade på detta redan 2006.

Forskargruppen kring professor Salvatore J Stolfo vid Columbia University i USA har länge fokuserat på intrångsdetektering. Under åren 2004 till 2007 arbetade två av hans doktorander med filtypsbaserad intrångsdetektering. Doktoranden Wei-Jen Li koncentrerade sig på att upptäcka elakartad kod i (text)dokument genom att studera avvikelser i den dataström som dokumentet kan betraktas som. Doktoranden Ke Wang skapade intrångsdetektorerna PAYL och Anagram. PAYL använder sig av enskilda bytevärden och skapar modeller av korrekt trafik. Anagram baseras på sekvenser av bytes och överträffar PAYLs prestanda på alla områden. Tekniken som används för PAYL och Anagram ligger till grund för de första metoderna för datatypsidentifiering som utvecklades inom Försvar av IT-system. För närvarande verkar Stolfos grups forskning inom datatypsidentifiering ligga nere.

Professor Robert Erbacher och hans forskargrupp vid Utah State University i USA, är inriktade på dator- och nätverksforensisk forskning med visualisering som specialitet. De arbetar med heterogena filtyper och använder olika statistiska mått för att kunna avgöra vilken typ av data ett fragment innehåller. Resultaten är mycket bra för textbaserad data (100 % korrekt identifiering), men sämre för binära data (JPEG och exe/dll) där 76 % av fragmenten identifieras korrekt. Deras senaste publikationer inom området gjordes 2008.

Professor Cor Veenman vid University of Amsterdam i Holland forskar kring data mining-problem inom datorforensiska undersökningar. Han arbetar vid Intelligent Systems Lab Amsterdam Forensic Intelligence och är också kopplad till Netherlands Forensic Institute. Han har gjort en publikation med inriktning mot filrekonstruktion. Där använder han bland annat bytefrekvensen hos ett datafragment för att bestämma dess datatyp. Hans föreslagna metod för datatypsidentifikation når 78 % korrekt identifiering för exekverbar kod, men den använder sig av information från kringliggande datafragment för att höja detektionsgraden.

Publikationslistan pekar på att han för tillfället arbetar med forskning med inriktning mot hantering av digitalt bild- och ljudmaterial. Intresset för filrekonstruktion verkar ha varit tillfälligt.

Professor Hyunjung Shin och hennes forskargrupp vid Anjou University Data Mining Laboratory i Sydkorea har sedan ett par år tillbaka forskat kring identifiering av datatyp hos fragment. De arbetar med heterogena filtyper och berör även exekverbar kod, men har nått sina bästa resultat med textbaserad data. Metoden de utvecklar är en tvåstegsmodell där en första grovsortering av liknande data görs på liknande sätt som de andra forskargrupperna i världen. I nästa steg använder de olika data mining-algoritmer för att identifiera vilken datatyp ett datafragment tillhör. Metoden ger relativt bra resultat

¹⁵ Omgivande information, det vill säga all information som inte direkt hörhörer från det undersökta datafragmentet.

för exekverbar kod och mycket bra resultat för textbaserad data. Publikationstakten hos gruppen är för närvarande ökande.

Bloomsburg University i USA har nyligen startat Pennsylvania Center for Digital Forensics som en avknoppning från institutionen för matematik, datavetenskap och statistik. Hittills har de inte producerat så många publikationer. Arvet från matematikinstitutionen märks i den metod för datatypsidentifiering som de arbetat fram. De använder ett flertal statistiska mått för att jämföra datafragment av okänd typ med modeller av kända datatyper. Arbetet liknar flera andra gruppers metoder och är inte banbrytande på något sätt. De har inte heller använt exekverbar kod vid sina experiment, utan bara textbaserad data och bilddata.

Trots bildandet av Pennsylvania Center for Digital Forensics verkar gruppen inte ha hunnit etablera sig ordentligt ännu. Publikationerna indikerar att personalen har en övervikt av matematiker på bekostnad av datavetare med IT-säkerhetskompetens.

En grupp från Iran University of Science and Technology har utvecklat en metod för datatypsidentifiering baserad på utvinning av utmärkande egenskaper hos olika datatyper. Dessa egenskaper används sedan för att träna ett neuralt nätverk. De rapporterar mycket bra resultat och då även för exekverbara filer. Den enda publikation som hittats beskriver dock inte metoden och experimenten i tillräckligt hög grad för att det ska gå att värdera resultatens trovärdighet. Det har inte heller gått att identifiera författarna till artikeln för att kunna bedöma värdet på kompetensen hos gruppen.

2.6.2 Övning

Övningar i försvar av IT-system kallas traditionellt för ”Capture the Flag”-övningar (CTF). De inkluderar både defensiva och offensiva moment för respektive deltagande lag och är vanliga inom den civila världen. En variant är att organisera övningen så att varje lag bara har en offensiv eller defensiv uppgift. Denna variant kallas ofta för en ”Cyber Defense Exercise” (CDX) och förekommer främst inom militär verksamhet. CTF- och CDX-övningar är väl inarbetade koncept och den utveckling som sker omfattar främst förbättringar av redan befintlig teknik. FOI:s CDX-verksamhet profilerar sig genom en bättre möjlighet till spårbarhet än vad som vanligtvis erbjuds. Främsta skälet till den utökade spårbarheten är användningen av F-REX-konceptet med inspelning av bland annat tidssynkroniserad bild, ljud och nätverkstrafik.

En av de mer kända CDX-övningarna är den årliga NSA/CSS CDX där främst amerikanska militärhögskolor under en veckas tid försvarar sig mot attacker från ett lag från NSA. Övningen har en distribuerad karaktär och de olika högskolorna är uppkopplade via ett Virtuellt Privat Nätverk (VPN), som avskärmar verksamheten från övriga internet.

DEFCON CTF är en icke-distribuerad och mycket känd CTF-tävling som körs under den årliga DEFCON-konferensen i Las Vegas, USA. Kopplingen till DEFCON, samt det faktum att det är en CTF-tävling, ger DEFCON CTF ett stort inslag av hacking.

Den (obekräftat) största distribuerade CTF-tävling som hålls är iCTF och organiseras av professor Giovanni Vigna vid University of California, Santa Barbara, USA. Den första tävlingen hölls 2003 och omfattade 14 lag. 2008 hade iCTF vuxit till 40 lag med totalt flera hundra deltagande studenter, vilket gjorde den till den största praktiska IT-säkerhetsövning som hållits över internet, enligt Giovanni Vigna.

CIPHER är en europeisk version av iCTF. Den hålls i samband med DIMVA-konferensen och organiseras av Special Interest Group SIDAR (Security — Intrusion Detection and Response) inom German Informatics Society (GI). Denna tävling har hållit på sedan 2005 och ambulerar runt Europa. Giovanni Vigna finns med bland organisatörerna.

2.7 Nyttjandeplan för Försvarsmakten

En nyttjandeplan som beskriver hur resultaten ska komma Försvarsmakten till del har tagits fram. Primär mottagare av projektets resultat är Claes Doverfors, FMTM.

I och med att den arkitektur varemot de verktyg och metoder som utvecklats inom projektet inte är fastställd ännu kan inte någon detaljerad nyttjandeplan tas fram. På kundens begäran har verksamheten rörande nya intrångsdetekteringstekniker inom projektet varit inriktad mot forskning med nyttjandemöjligheter på längre sikt. De metoder för datatypsidentifiering som tagits fram är dock avsedda att kunna börja integreras i kommande verksamhet när så är lämpligt. Verktygen har ett brett användningsområde vilket gör att de kan nyttjas omedelbart med obetydliga anpassningskrav. Kunskapsöverföring har bland annat skett vid den tekniska demonstrationsdag som hölls i Enköping 2008-09-16.

Övningarna som hållits har byggt upp en unik kompetens om internationella distribuerade övningar med parter från samhällets alla grenar. Den tekniska plattformen i form av Monolith-klustret med stöd- och spårbarhetssystem har utvecklats och förädlats. Det gör att plattformen nu är redo att användas för övningar inom IT-säkerhet och IT-försvar med kort varsel. Genom det internt enhetsöverskridande samarbete som legat till grund för uppbyggnaden av övningsplattformen finns det en bred erfarenhet att ta av när nästa övning ska arrangeras. De exakta datumen för dessa övningar kommer att tas fram fortlöpande i dialog med kunden. Lämpliga tillfällen är till exempel vid utbildning av de delar av utlandsstyrkan och andra internationella insatsförband där det finns krav på god kompetens inom IT-säkerhet och IT-försvarsförmåga.

Erfarenheterna från övningsverksamheten har direktåterförts till kunden vid de tillfällen de deltagit. Erfarenheterna från den internationella övningen med deltagare från Estland har presenterats i form av ett FOI-memo.

2.8 Samverkan ”Lägesförståelse på informationsarenan”

På initiativ av verksamhetsområdeschef Anders Törne påbörjades planeringen av en framtida samverkan mellan FoT-projekten *Försvar av IT-system* (genom den föreslagna fortsättningen *Spaning och motmedel på Informationsarenan*), *Testning av metoder och verktyg för värdering av informationssäkerhet* och *Situations- och hotanalys för BG2011*. En första workshop hölls 2009-05-27 med ett uppföljande möte 2009-08-24. Resultatet av workshopen var ett beslut om att ”gemensamt formulera en vision för hur forskningsresultaten från projekten kan nyttiggöras i framtida implementationer av FM NOC/GLC.” [FOI Memo 2829]. Vid mötet 2009-08-24 diskuterades vad termerna NOC och GLC innebar, samt vilket område som var bäst att satsa på ur kundens perspektiv. Resultatet av diskussionerna var att FOI bör vara med och definiera termerna i detalj, samt att NOC för närvarande verkade vara den bättre inriktningen utifrån de tolkningar som projektdeltagarna gjorde.

Nytan för FM med en framtida samverkan mellan projekten är framför allt att eventuellt dubbelarbete undviks. Projektens gränsyta mot FM blir nu mer homogen och eventuella synergieffekter från kunskapsutbyte mellan relaterade områden tas till vara. Kunskapsöverföringen till FM blir också bättre genom att kunden kan träffa deltagare från tre projekt på en gång vid eventuella projektgemensamma seminarier och demonstrationer.

3 Summerande slutsatser

Projektet har varit uppdelat i två huvudgrenar, datatypsidentifiering och IT-säkerhetsövning. Datatypsidentifieringen är relaterad till intrångsdetektering, vilket också delvis berörts i projektet. Övningsverksamheten omfattar två större övningstillfällen, en internationell och distribuerad flaggerövringsövning tillsammans deltagare från Estland och en rent offensiv övning med deltagare från relevanta svenska myndigheter. Den senare övningen har ännu inte genomförts vid denna rapports färdigställande. Resultaten från den övningen redovisas separat i ett FOI-Memo.

Eftersom detta är en användarrapport ligger tyngdpunkten på redovisningen på hur resultaten av projektets verksamhet ska användas av FM. Till det har fogats förklarande text där så varit tvunget för läsbarhetens skull.

3.1 Datatypsidentifiering

Det finns inte någon universalparameter att detektera intrång med, utan ju mer information det finns att tillgå, desto säkrare blir detektionen. Det är alltså principen om informationsövertag som gäller vid intrångsdetektering, den som vet mest vinner.

Vi har visat att det går att uppnå bra resultat vid identifiering av datatyp hos datafragment utan tillgång till annan information än den som återfinns i fragmenten. Resultaten av vår forskning bidrar till att utöka intrångsdetekteringssystemens förmåga med en möjlighet att identifiera datainnehållet i nätverkspaket utan att behöva lita till meta-information. På så sätt kan till exempel körbar kod identifieras och sedan skickas för djupare analys till lämplig funktion.

Våra experiment har visat att mängden data som används för att skapa datatypsmodeller är viktig för enbytesalgoritmerna och mycket viktig för byteparsalgoritmen. Detta hänger ihop med den 256-faldiga ökning av mängden möjliga unika bytepar hos byteparsalgoritmen jämfört med enbytesalgoritmerna. Det gör också att valet av metod inte bara beror på kraven på exekveringshastighet och detektionsgrad som finns, utan även på hur mycket träningsdata som finns att tillgå. En annan viktig parameter att ta i beaktande vid val av identifieringsalgoritm är vilken metrik som lämpar sig för algoritmen. Det betyder att om exempelvis identifiering av maskinkod ska göras i nätverkstrafik bör BFD-algoritmen med sannolikhetsmetrik användas om det finns tillräckligt med primärminne att tillgå.

Ska kontroll av att all nätverkstrafik är krypterad göras passar BFD-algoritmen med viktad kvadratsummemetrik bäst. Den lämpar sig också för identifiering av komprimerad data. Är det istället multimediadata som ska identifieras och det inte finns några höga krav på exekveringshastighet lämpar sig byteparsalgoritmen med viktad kvadratsummemetrik bäst. Den är även bra för datatyper där formatspecifikationen anger att uppsättningen av tillåtna bytepar är mindre än mängden av alla möjliga bytepar.

Ett alternativ till ovanstående algoritmer är att använda en kombination av BFD- och RoC-algoritmerna med viktad kvadratsummemetrik. Denna lösning lämpar sig bäst för situationer där kraven på detektionsgrad inte är allt för höga, men där förmågan att detektera ett brett spektrum av datatyper är önskvärd.

Forskningen kring datatypsidentifiering expanderar och börjar nu bli etablerad i forskarvärlden. Ett flertal internationella forskargrupper ligger långt framme och det förspång som FOI fick via sin tidiga start minskar stadigt.

3.2 Övningsverksamhet

Ett av grundproblemen vid arbete med att skapa gemensamma lägesbilder för ett nätverks status är att det inte finns någon bra och enhetlig definition av termen. Vi hoppas att ett av resultaten av den IT-säkerhetsövning som ska hållas 2009-12-10 är möjligheten att ge ett förslag på definition. Syftet med övningen är att studera hur en heterogen grupp av IT-säkerhetstekniker går till väga när de skapar sig en gemensam lägesbild av ett nätverk de aldrig tidigare sett.

Ytterligare ett syfte med övningen är att utvärdera den plattform för IT-säkerhetsövningar med tillhörande spårbarhetssystem som tagits fram inom projektet. Spårbarhetssystemet medger loggning och inspelning av ljud, video, skärmdumpar och ett flertal andra parametrar som kan vara av intresse vid en utvärdering av en övning.

Spårbarhetssystemet är kopplat till F-REX Studio vilket gör det möjligt att utföra avancerade analyser av genomförandet av en övning och tillåter en omfattande återkoppling till övningsdeltagarna eller andra intressenter. Plattformen och de erfarenheter av övningsverksamhet som FOI bygger upp kommer FM till del dels genom direktöverföring när deras personal deltar i övningen, dels via den ökade förmågan till snabb framtagning av komplexa övningsupplägg som plattformen medger.

Övningsplattformens potential visade sig vid den internationella distribuerade övningen som hölls tillsammans med Estland. Där framgick också att det ställs stora krav på hårdvaran och att plattformen behövde uppgraderas i vissa delar. Anledningen var att hela systemet, inklusive det från övningsdeltagarna isolerade administrativa nätverket, slutade fungera när det angripande laget utförde en omfattande DDoS-attack.

Erfarenheterna från Estlandsövningen visar också att det krävs stora resurser för att sätta upp en bra övningsmiljö. Inlärningskurvan är dock brant och det finns stora vinster i att skapa återanvändningsbara funktionsmoduler. Likaså bidrar den ökande erfarenheten och kompetensen hos berörd personal på FOI till att underlätta framtida övningar, oavsett om de är repetitioner av tidigare övningar eller har helt nya upplägg.

Det finns tre typer av övningar, rent defensiva, rent offensiva och kombinationen av de två, vilket också är den vanligaste övningstypen. Övningsplattformen och personal från FOI har arrangerat eller varit involverade i alla tre typerna av övningar. Det ger FM möjligheten att fritt välja vilken typ av övning de vill ska hållas. FOI kan sedan snabbt iordningställa övningsplattformen i enlighet med önskat upplägg.

Plattformen med tillhörande stödsystem kan användas för utbildning, koncepttestning, vetenskapliga experiment, utvärdering av olika taktiker, utveckling av nya försvarsmetoder, med mera.