Projektnummer/Project no
E72779
Kund/Customer
Försvarsmakten
FoT-område
Operationer i cyberdomänen

Handläggare/Our reference
Tobias Lundberg

Datum/Date
2019-11-07
Memo nummer/number
FOI Memo 6943

**Logginsamlingsverktyg till CRATE**

Sändlista/Distribution:

LEDS CIO (avsett för FoT-ordförande David Olgart)
ITF (avsett för Mathias Bjärme)
FRA
PROD LEDUND (avsett för Gunnar Marcusson)
MUST
FOI (avsett för forskningsområdesföreträdare Operationer i cyberdomänen)

# Innehåll

# 1. Introduktion

Vid Totalförsvarets forskningsinstitut (FOI) i Linköping finns en laborations- och övningsplattform för cybersäkerhet kallad CRATE (Cyber Range and Training Environment)[1]. CRATE har utvecklats från att vara ett enkelt datorkluster byggd för en enskild övning 2008 av donerad hårdvara, till en anläggning av stor betydelse för de tjänster FOI erbjuder inom IT-säkerhet. CRATE består i huvudsak av mjukvaruverktyg för att skapa och kontrollera cybermiljöer i en datorhall. Till exempel kan man i CRATE använda grafiska gränssnitt för att skapa datornätverk av olika virtuella Windows- och Linuxdatorer, utföra angrepp mot datorerna och mäta effekterna av dessa angrepp.

CRATE och tillhörande kompetens har byggts upp för att möta behov i uppdrag finansierade av Försvarsmakten och Myndigheten för Samhällsskydd och Beredskap. Uppdragen har innefattat:

- utbildning, t.ex. då ”labbar” utförs som en del av kurser
- träning av IT-personal, t.ex. i verktyg och sårbarhetsanalys
- övning av IT-säkerhetsexperter, t.ex. incidenthanteringsövningar för CERT:ar
- tekniska tester, t.ex. av intrångsdetekteringssystem.

Det är vanligt att den typ av uppdrag som beskrivs ovan kräver insamling av data (t.ex. nätverkstrafik eller loggar från Windows-maskiner) som sedan ska läsas in av säkerhetsverktyg. Det har tidigare saknats verktyg för att med enkelhet konfigurera datainsamling i CRATE och varje ny tillämpning har krävt manuellt arbete.

På grund av detta planerades utvecklandet och införandet av ett logginsamlingsverktyg. Målet med verktyget är att lösa problemet med att låta en användare välja vad som ska loggas på virtuella maskiner i CRATE. Användaren ska även kunna välja vart de genererade loggfilerna ska flyttas. Bland annat följande krav sattes på verktyget:

1. Loggar ska kunna föras över mellan VirtualBox[2]-maskiner, oberoende av vilken virtualiseringsnod i CRATE maskinen körs på.
2. Sökväg till loggar och tidsintervall mellan överföringar av loggar ska kunna anges.
3. Det ska gå att välja vilka händelsetyper som ska loggas i Windows-maskiner, samt var loggarna ska sparas.
4. Det ska gå att välja vilka nätverksinterface som ska loggas i Linux-maskiner, samt var loggarna ska sparas.
5. Det ska gå att välja vilka syslog-händelser som ska loggas i Linux-maskiner, samt var loggarna ska sparas.

Under hösten 2019 togs en första lösning på ett logginsamlingsverktyg fram. Verktyget kan användas för att konfigurera vad olika maskiner i CRATE ska logga, och hur loggarna ska flyttas mellan maskinerna i spelnätet. Det här memot beskriver översiktligt hur verktyget är uppbyggt, vad det har för funktionalitet, hur det används, och hur det kan vidareutvecklas i framtiden.

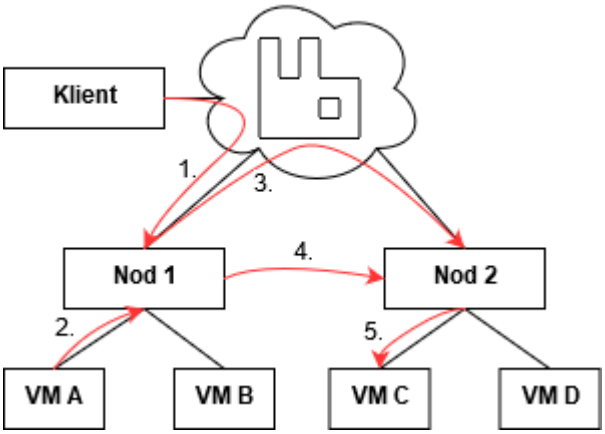
[1] https://www.foi.se/CRATE

[2] Den hypervisor som används i CRATE, se https://www.virtualbox.org/

# 2. Systemöversikt

Det här kapitlet ger en översikt över hur systemet är uppbyggt och vilka lösningar som har använts för att bygga logginsamlingsverktyget.

Ett krav på logginsamlingsverktyget är att kunna transportera data mellan två godtyckliga virtuella maskiner i CRATE, även om dessa inte har någon nätverkskoppling mellan sig i spelnätet. Det innebär att flödet måste gå genom virtualiseringsvärden (även kallad noden). Om en användare vill att data ska flyttas från VM A (som körs på Nod 1) till VM C (som körs på Nod 2), så ser dataflödet ut som Figur 1.




Figur 1 – Dataflöde genom logginsamlingsverktygets olika delar

1. Användaren av programmet meddelar Nod 1 att den önskar att data ska flyttas från VM A till VM C.
2. Nod 1 hämtar data från VM A och lagrar den temporärt.
3. Nod 1 meddelar Nod 2 att det finns ny data att hämta, och att datan ska läggas på VM C.
4. Nod 2 hämtar datan från Nod 1.
5. Nod 2 lägger in datan på VM C.

## 2.1 Agenter

För att uppfylla de ställda kraven infördes en agent som körs på varje nod. Agenterna är skriva i Python 3.6 och har två huvudsyften: att ta emot kommandon från en användare av logginsamlingsverktyget och att styra de virtuella maskinerna utifrån användarens önskemål. Kommandon tas emot genom RabbitMQ, som beskrivs närmare i kapitel 2.2. Styrningen av de virtuella maskinerna görs genom användning av virtualbox-python och vboxmanage, som finns beskrivna i kapitel 2.3.

## 2.2 RabbitMQ

För att lösa kommunikationen mellan noderna så valdes meddelandehanteraren RabbitMQ[3], med Advanced Message Queuing Protocol (AMQP) som överföringsprotokoll. Valet av RabbitMQ skedde av lera anledningar, varav de avgörande var:

- RabbitMQ kan skicka kommandon till och mellan noder med en garanti att de kommer fram. Detta utan att bekräftelse på att kommandot kommit fram behöver inväntas. RabbitMQ kan garantera detta i och med att det är en asynkron lösning med ett kösystem.

[3] https://www.rabbitmq.com/

- RabbitMQ bedömdes som smidigt att sätta upp och använda. Python har ett bibliotek vid namn *pika* som inte kräver speciellt många rader kod för att kunna börja skicka meddelanden eller börja lyssna efter meddelanden.

Andra lösningar som utvärderades var Apache Kafka eller ett eget REST-baserat system, men RabbitMQ var den lösning som passade bäst i sammanhanget.

En central RabbitMQ-server sattes upp i det administrativa nätet i CRATE. Agenten på varje nod registrerar sig som en prenumerant i ett RabbitMQ-utbyte vid namn cratelogs. Specifikt så använder varje nod FQDN[4]-adresserna till de virtuella maskinerna som körs på noden som dirigeringsnyckel. I slutändan innebär detta att man kan skicka ett kommando adresserat till ett visst FQDN, varpå RabbitMQ-servern kommer att skicka vidare det till den nod som kör en maskin med motsvarande FQDN.

Utöver kommunikationen från klient till nod behöver noderna även kunna skicka kommandon till varandra (exempelvis för att transportera filer). Detta fungerar på samma sätt som kommunikationen mellan klient och nod, med en nod som agerar klient för att kommunicera med en annan nod.

## 2.3 Virtualbox API, vboxmanage, och virtualbox-python

Varje nod behöver kunna skicka kommandon till de virtuella maskiner som noden kör. VirtualBox har två sätt att åstadkomma detta: dels genom ett COM[5]-baserat API, och dels genom ett skalbaserat verktyg vid namn vboxmanage. Vboxmanage använder sig av API:et i grunden, men utgör en ytterligare abstraktionsnivå som gör det lite lättare att använda.

Logginsamlingsverktyget använder vboxmanage i så utsträckning som möjligt för att exekvera kommandon på de virtuella maskinerna. Dock saknade vboxmanage viss funktionalitet (att namnge en gästsession vid start av bakgrundsprocesser), vilket gjorde att API:et behövde användas. Detta gjordes genom biblioteket virtualbox-python[6], som implementerar API:et på ett Pythoniskt vis, med viss extrafunktionalitet.

[4] Fully Qualified Domain Name

[5] Component Object Model

[6] https://github.com/sethmlarson/virtualbox-python

# 3. Funktionalitet

Det här kapitlet beskriver den funktionalitet som finns i verktyget, och vad en användare har möjlighet att använda verktyget till.

## 3.1 Flyttning av filer och mappar

De flesta loggningsverktyg involverar att filer med logginformation skrivs ner som filer till disk. Detta inkluderar även de logginsamlingsverktyg som körs på de virtuella maskinerna i CRATE. Detta innebär att en mycket central funktionalitet i logginsamlingsverktyget är att kunna peka ut de filer man är intresserad av och specificera vart de ska flyttas.

Logginsamlingsverktyget har stöd för att låta användaren specificera följande:

- vilka filer och mappar användaren är intresserad av, och på vilken maskin
- hur ofta de ska kontrolleras (periodicitet)
- till vilken maskin de ska flyttas (mottagare), och till vilken sökväg på den maskinen
- om filer får skrivas över eller inte.

Om det är en fil som användaren önskar flytta på kontrollerar agenten, med den angivna periodiciteten, om filen har ändrats sedan den senaste kontrollen. Om filen har ändrats skickas en kopia till den specificerade mottagaren, där filen lagras i den angivna mappen med antingen tidsstämpel konkatenerat med originalfilnamn som filnamn (om filer inte får skrivas över) eller bara med originalfilnamn (om filer får skrivas över).

Flyttning av mappar fungerar på samma sätt som flyttning av filer, men med en kontroll för varje fil i den angivna mappen.

## 3.2 Inställningar för Windows eventloggar

Vilka eventloggar som finns på en Windows-maskin skiljer sig lite mellan versionerna. Från undersökningar som gjordes av några typer av maskiner konstaterades att Security, Application, och System alltid verkar finnas tillgängliga. För dessa loggar har en användare möjlighet att ändra på diverse olika inställningar.

Om en användare är intresserad av att flytta en Windows-logg från en maskin till en annan, så går det att specificera vilken logg som ska flyttas, på liknande sätt som beskrivs i avsnitt 3.1. Skillnaden är att endast loggens namn (”Security”, ”Application”, eller ”System”) och inte fullständig sökväg behöver anges.

I Windows kan man ange en så kallad ”Audit Policy”, som bestämmer vad som ska loggas till Security-loggen. Användaren kan välja att slå på eller av loggning av följande händelser: ”System”, ”Logon/Logoff”, ”Object Access”, ”Privilege Use”, ”Detailed Tracking”, ”Policy Change”, ”Account Management”, ”DS Access”, och ”Account Logon”.

Det kan noteras att listan i föregående stycke inte är samtliga inställningar som går att ändra, och möjligheten att ändra på inställningar skiljer sig dessutom något mellan olika versioner av Windows. Det kan även noteras att loggade händelser inte omedelbart verkar sparas till filen, vilket gör att det kan dröja innan nya logghändelser registreras av logginsamlingsverktyget.

## 3.3 Tcpdump i Linux-maskiner

Tcpdump används ofta för att samla in information om nätverkstrafik på ett visst interface. I logginsamlingsverktyget kan användaren välja att starta tcpdump på linuxmaskiner. Dessa startas igång som en bakgrundsprocess, som körs tills den virtuella maskinen stängs av eller användaren väljer att

stoppa tcpdump. Vidare kan användaren välja godtyckliga parametrar till tcpdump, i form av flaggor och värden.

## 3.4 Syslog i Linux-maskiner

Syslog är en standard för loggning som används på merparten av alla Linux-maskiner i CRATE. I logginsamlingsverktyget kan användaren skicka in inställningar till en syslog-installation på en virtuell Linux-maskin. Inställningar specificeras som värdepar, där en syslog-selector kopplas till en viss sökväg på maskinen. Agenten på noden hittar och modifierar lämplig konfigurationsfil och startar om syslog, oavsett vilken implementation som används på den specifika maskinen.

# 4. Användning

Systemet används enklast genom en Python-klient som tar en inställningsfil som indata. Se exempel på en sådan inställningsfil nedan. Inställningarna i exemplet nedan specificerar att Windows-maskinen ”files.int.stensson.se” ska spara Application-loggen till *%systemroot%\\APPLICATION.evtx*, och Security-loggen till *%systemroot%\\SECURITY.evtx*. Security-loggen ska sedan skickas till */root/stenssonlogs* på maskin ”logs.int.cert.dk” och ”logs.stensson.se”, och kontrolleras varannan minut. Till Security-loggen ska ”Object Access” och ”Logon/Logoff”-händelser lagras, men inte ”Policy Change”. På Linux-maskinen ”fw.int.stensson.se” ska tcpdump logga interface *eth0* och *eth1* till */tmp/eth0log* respektive */tmp/eth1log*. Båda dessa ska skickas till */root/stenssonlogs/fw* på ”logs.int.cert.dk”, och kontrolleras var 60:de sekund.

Syslog ska logga ”mail.info”-händelser till */tmp/logging/mailinfo.log* och ”kern.alert”-händelser till */tmp/logging/kernalerts.log*. Hela mappen */tmp/logging* ska kontrollerad var 60:e sekund och uppdaterade filer ska skickas till */root/stenssonlogs/fw* på ”logs.int.cert.dk”.

```
{
   "files.int.stensson.se":
   {
      "log_events": [
         {
            "type": "systemlog",
            "systemlog": "Security",
            "periodicity": 120,
            "destinations": [
               {"fqdn": "logs.int.cert.dk", "path": "/root/stenssonlogs"},
               {"fqdn": "logs.stensson.se", "path": "/root/stenssonlogs"}
            ]
         }
      ],
      "audit_policy":
      [
         {"category": "Object Access", "value": true},
         {"category": "Logon/Logoff", "value": true},
         {"category": "Policy Change", "value": false}
      ],
      "eventlog_settings":
      [
         {"log": "Application", "path": "%systemroot%\\APPLICATION.evtx"},
         {"log": "Security", "path": "%systemroot%\\SECURITY.evtx"}
      ]
   },

   "fw.int.stensson.se":
   {
      "log_events": [
         {
            "type": "folder",
            "local_folders": [
               "/tmp/logging"
            ],
            "periodicity": 60,
            "destinations": [
               {"fqdn": "logs.int.cert.dk", "path": "/root/stenssonlogs/fw"}
            ]
         },
         {
            "type": "file",
            "local_files": [
               "/tmp/eth0log", "/tmp/eth1log"
            ],
            "periodicity": 60,
            "destinations": [
               {"fqdn": "logs.int.cert.dk", "path": "/root/stenssonlogs/fw"}
            ]
         }
      ],
      "syslog_settings": [
         {"selector": "mail.info", "local_file": "/tmp/logging/mailinfo.log"},
         {"selector": "kern.alert", "local_file": "/tmp/logging/kernelalerts.log"}
      ],
      "tcpdump_settings":
      [
         {"i": "eth0", "w": "/tmp/eth0log"},
         {"i": "eth1", "w": "/tmp/eth1log"}
      ]
   }

}
```

När inställningarna har skickats ut till rätt noder så övergår klienten till att visa en logg över de kommandon som varje agent genomför. Detta för att användaren ska kunna se om allt fungerar som det ska, och för att felsökning ska bli enklare.

# 5. Test av logginsamlingsverktyget i CRATE

För att säkerställa att logginsamlingsverktyget uppfyller de uppsatta kraven genomfördes tester av det i en träningsmiljö i CRATE. Miljön som användes var framtagen för att användas i SafeCyber 2019. Runt 50 maskiner i en viss organisation fick i uppdrag att samla in följande:

6. Samtliga Windows-maskiner (totalt 28 stycken) skulle flytta sina Security-loggar till insamlingsmaskinen, med en kontroll var 10:de sekund. Dessa maskiner inkluderade bland annat Windows 7, Windows Server 2008, och Windows XP.
7. En Linux-maskin skulle starta tcpdump och föra över pcap-filerna till insamlingsmaskinen, med en kontroll var 15:de sekund. En SlowLotus-attack kördes mot maskinen för att pcapfilerna skulle växa i storlek mellan kontrollerna.
8. Sex Linux-maskiner skulle föra över */var/lib/tcpdump* till insamlingsmaskinen, med en kontroll var 30:de sekund. All nätverkstrafik loggas med tcpdump till den mappen. Observera att logginsamlingsverktyget inte användes för att starta tcpdump på dessa maskiner.

Totalt involverades sex noder. Maskinerna som data samlades från låg utspridda på fem av dem, och insamlingsmaskinen låg på ytterligare en. Detta är mindre än vad riktiga övningar i CRATE kan vara, men tillräckligt för att observera logginsamlingsverktygets styrkor och svagheter.

De flesta av loggarna från Windows-maskinerna fördes över som de skulle. Vissa maskiner verkade inte vilja ta emot kommandon från vboxmanage vid vissa tillfällen. Se kapitel 5.1 för mer information.

Linux-maskinerna hade mycket data i */var/lib/tcpdump*, runt 3 GB totalt, vilket gjorde att en första överföring av dem tog lite tid. En fil på 500 MB tog ungefär 6 sekunder att flytta till rätt plats.

Testet flöt på som det skulle under 20 minuters observation, där de loggar som växte i storlek fördes över till insamlingsmaskinen med den angivna periodiciteten.

## 5.1 Stabilitetsproblem

Logginsamlingsverktyget bygger till stor del på verktyg som dras med vissa stabilitetsproblem. Som exempel händer det ibland att det plötsligt slutar fungera att logga in på en viss maskin utan att först starta om den. Vid vissa tillfällen kan man även få felmeddelanden när man försöker skapa mappar eller kontrollera om ändringar skett för en viss fil, men vid försök några sekunder senare fungerar det felfritt. Vad exakt dessa fel beror på är svårt att avgöra, men logginsamlingsverktyget har skrivits för att hantera felen graciöst.

# 6. Framtiden

Verktyget som beskrivs i detta memo har tagits fram för att lösa ett specifikt problem: hur kan man designa ett system som möjliggör att på ett enkelt sätt kontrollera loggning på virtuella maskiner, samt flytta loggar mellan virtuella maskiner?

Verktyget löser dock ett mer generellt problem, fjärrstyrning av program som finns på en virtuell maskin samt kommunikation mellan de noder som kör virtuella maskiner. Detta är något som går att använda till mer än att bara starta och stoppa olika logverktyg. Exempelvis är ett nytt utrullningssystem för virtuella maskiner på noder i CRATE planerat och det systemet skulle kunna använda sig av en utökad version av det system som finns beskrivet här. Detta kan åstadkommas genom att helt enkelt utöka agenten till att även hantera kommandon som installerar en ny VM givet vissa parametrar. Även vissa bottar i CRATE använder sig av ett system som kör någon typ av fjärrstyrning av virtuella maskiner, vilket skulle kunna gå att implementera som en del av agenten.

I framtiden kan det således vara önskvärt att dela upp agenten från den specifika logginsamlings-funktionaliteten, genom att modularisera systemet. På så sätt kan hanteringen av de olika loggverktygen (tcpdump, Windows eventloggar, med flera) hållas separat från kommunikationssystemet, och systemet som helhet kan bli enklare att utöka.

Förutom utökad funktionalitet finns det fler saker som kan göras med själva logginsamlingen. Möjligheten att välja filer som ska flyttas skulle kunna göras mer kraftfull, genom att exempelvis låta användaren skriva in ett reguljärt uttryck för filnamn. Man skulle även kunna implementera stöd för fler verktyg än de som beskrivs här.

För närvarande saknar verktyget även någon typ av gränssnitt som är enkelt att använda, då all användning sker via manuell redigering av JSON-filer. I framtiden kan det vara önskvärt att låta användaren modifiera loggning på maskinerna via ett gränssnitt i stil med det grafiska gränssnittet i CRATE 2.0.