Device Guard och VSM som ett skyddslager mot skadeprogram

Anton Lindholm

Kandidatavhandling

4/6/16

INNEHÅLLSFÖRTECKNING

REFERAT

Antivirusprogram har under de femton senaste åren allmänt fungerat som det primära skyddet mot skadeprogram. Samtidigt har skadeprogrammens förekomst och skadorna dessa program gett upphov till kraftig ökat. De traditionella antivirusprogrammen har visat sig vara otillräckliga. Metoder för att kringgå traditionell sekvenssökning, beteendebaserad sökning och emulering av kod existerar. Dokumentationen av dessa metoder finns gratis tillgänglig på nätet och de kriminella har tillgång till själva antivirusprogrammen.

Som ett verktyg mot detta har Microsoft under början av 2015 lanserat ny funktionalitet under namnet Device Guard. Istället för att köra allt som antivirusprogrammen inte tycker är farligt kör ett system med Device Guard applicerat endast signerad kod. Vilka certifikat man litar på bestäms av

systemadministratören och istället för den vanliga användaren.
Denna kod-integritets funktionalitet har separerats från operativsystemets kärna och körs i en skyddad virtuell container. Alla dessa virtuella säkerhetsåtgärder har Microsoft valt att marknadsföra under namnet Virtual Secure Mode och för att kunna aktivera detta nya läge krävs viss funktionalitet av hårdvaran.

Microsoft har satt strikta krav på både hårdvara och mjukvara. Detta är ett måste för att kunna garantera att systemet fungerar som en säker helhet. Företaget har även ökat på kraven man ställer på koden som tillåts köras i kärnan och pressat hårdvarutillverkare att fixa säkerhetshål som uppenbarats.

SYFTE

I januari år 2015 kom Ponemon Institute LLC ut med en rapport under namnet ”The Cost of Malware Containment” [1]. Målet med rapporten var att få en bild av hur mycket resurser it-bolag i USA tvingas lägga på kostnader som uppstår på grund av sabotageprogram.

För att utreda detta kontaktade man 630 personer inom företag som är verksamma inom it och it-säkerhet i USA. I snitt fick bolagen utstå 17 000 säkerhetsvarningar per vecka och av dessa ansågs i snitt 3 218 utgöra ett verkligt hot. Även om de endast valde att analysera en knapp tjugondel av dessa kostade det bolagen i snitt 1,27 miljoner dollar per år.

I undersökningen tog man reda på vilka metoder bolagen använde sig av för att skydda sig mot och åtgärda skador som uppstått av skadeprogram. Av de tillfrågade svarade 33 % att man inte har en plan för hur detta sköts. 24 % uppgav att de hade en plan som i huvudsak byggde på automation. Resten av de tillfrågade tillämpade en strategi som främst byggde på manuella åtgärder eller både manuella och automatiserade operationer. 60% av de tillfrågade uppgav att de upplevde en ökning eller en klar ökning i både antalet infektioner och allvarligheten av dessa.

I mars samma år presenterade Microsoft [2] en rad nya säkerhetsfunktioner för Windows 10. Dessa funktioner bygger upp en helhet man marknadsför under namnet Device Guard. Helheten ska fungera som en heltäckande lösning för att bygga upp en säker systemhelhet.

Med Device Guard vill Microsoft ge företag möjligheten att ha full kontroll över vilken kod som kan exekveras på deras arbetsmaskiner [3]. Man utnyttjar ny teknik inom virtualisering för att separera operativsystemets kärna från resten av systemet.

I denna avhandling kommer jag undersöka för- och nackdelarna med att utnyttja Device Guard. Den nya tekniken kommer även att jämföras med traditionella antivirusprogram.

ANTIVIRUSPROGRAM

I detta kapitel kommer jag att analysera antivirusprogram med fokus på hur dom fungerar överlag. I slutet av kapitlet gör jag en kort analys om hur väl de uppfyller sin uppgift i det säkerhetsklimat som råder idag.

I dagens värld utnyttjas antivirusprogram ofta som det primära skyddet för företags datorer. Detta medför att dessa applikationer måste erbjuda ett helhetskydd för systemet. Att en komponent har ett så stort ansvarsområde innebär dock att säkerheten försvagas. Om komponenten inte lyckas fullfölja sin uppgift har det allvarliga följder. National Institute of Standards and Technology(NIST) rekommenderar djupförsvar, men i praktiken är det vanligt att företag slarvar med implementationen av detta koncept. [5]

Metoderna antivirus program utnyttjar har inte genomgått märkbar förändring de under de senaste femton åren. Programmen klassar huvudsakligen filer enligt två metoder [4]:

- Signaturbaserad sökning
- Beteendebaserad sökning

I följande två underkapitel behandlar jag dessa två sökmetoder. Jag har valt att inkludera testexekvering i virtuella miljöer i slutet av det andra underkapitlet.

SIGNATURBASERAD SÖKNING

Den signaturbaserade sökningen går ut på att man identifierar bytesekvenser hos kända skadeprogram. Traditionellt kan en sådan sekvens direkt kopplas till skadeprogrammet. Sekvensen tas fram genom analys av innehållet i en skadlig fil. Denna sekvens identifierar den kända skadliga filen och sannolikheten att genuina applikationer innehåller samma sekvens är liten.

Förutom traditionella sekvenser utnyttjas även ibland sekvenser som identifierar operationer som skadliga program utför. Man söker fram bytesekvenser som representerar operationer typiska för skadeprogram. Denna metod gör att man potentiellt kan hitta inte ännu identifierad skadlig programvara. Tyvärr klassas ibland legitim mjukvara som skadlig för att den också använder sig av liknande operationer. [4]

Signaturerna jämförs med bytesekvenser från filer på värdens maskin. Om en sekvens hos ett program stämmer överens med sekvenserna hos ett redan känt skadeprogram klassas filen som potentiellt skadlig och sätts i karantän. Eftersom alla nya filer i systemet skannas kan antivirusprogrammet med hjälp av denna metod snabbt isolera suspekta filer från resten av systemet. [4]

Den signaturbaserade sökningen har dock stora svagheter. Båda tidigare nämnda rapporterna påpekar att denna metod inte lyckas hitta nya skadeprogram. Som Haffejee J. (2014) visar kan skadeprogram enkelt göras unika genom att de packas om och krypteras. Detta tillvägagångssätt kan som sådant inte klassas som suspekt. Legitim programvara genomgår ofta samma process eftersom filstorleken reduceras och man vill skydda programvaran från att bli piratkopierad [6].

BETEENDEBASERAD SÖKNING

Implementationen av den beteendebaserade sökningen skiljer sig mellan antivirusprogram. Vilka tekniker man utnyttjar och exakt vilka beteenden man granskar hemlighålls ofta av bolagen bakom antivirusprogrammen. Detta motiveras med att kännedom om hur antivirusprogrammen fungerar underlättar skapandet av skadeprogram som obemärkt kan exekveras i bakgrunden. [7]

I den empiriska forskningen som Orathai Sukuwong (2011) presenterar väljer man därför att lägga fokus på filer, minnet, operativsystemets register och systemets nätverksaktivitet. Studien visar att beteendebaserad sökning försvårar aktiviteten för skadeprogrammen, men tyvärr finns ändå brister hos samtliga testade lösningar.

För att kunna fortleva i ett system måste skadeprogram se till att deras skadliga kod exekveras igen efter att systemet startas om. Ofta skapar programmen nya filer och editerar eller binder sig till redan existerande filer. Vad antivirusprogrammen uppfattar som suspekt skiljer sig mycket. Typiskt för alla antivirusprogram var att de tillät att filer skapades och modifierades i mappar ämnade för temporära filer och i vanliga programmappar. Förändringar hos system- och biblioteksfiler övervakades extra noga eftersom dessa filer utnyttjas av en stor del av vanlig benign programvara.

Antivirusprogrammen hade samma prioritering när det gällde skydd av systemregistret. Nycklar som var relaterade till operativsystem- och nätverksfunktionalitet prioriterades. I avhandlingen [4] nämns att antivirusprogram ofta tillåter att nycklar som aktiverar eller stänger av specifik programfunktionalitet ändras. Detta i sin tur öppnar upp för nya attacker eller försvårar upptäckten av dessa.

För att skydda nätverket och nätverksaktiviteten använde de testade antivirusprogrammen sig av en egen brandmur, en modul som granskade email

trafiken och i många fall även insticksmoduler till olika webbläsare. Utifrån de avhandlingar jag tidigare nämnt drar jag slutsatsen att dessa moduler mer är till för att minimera risken för skador snarare än att finna skadeprogram.

Brandmuren som följde med antivirusprogrammen läste inte av och tolkade trafiken som sådan. Sukuwong och hans kolleger nämnde ledde detta till att skadeprogram kunde utnyttja tillåtna protokoll för att hämta fler skadliga filer från internet efter att de fått tillåtelse att passera brandmuren. Vissa av antivirusprogrammen märkte även om värdmaskinen utnyttjades för att attackera andra maskiner.

Förutom de funktioner man tog upp i ”Commercial Antivirus Software Effectiveness: An Empirical Study” [4] har antivirusprogram börjat utnyttja små virtuella miljöer för att testköra filer före filerna tillåts exekvera på värdmaskinen. Effektiviteten hos denna teknik har Arne Swinnen undersökt i sin avhandling ”One packer to rule them all: Empirical identification, comparison and circumvention of current Antivirus detection techniques” [7].

Swinnen beskriver idén bakom tekniken som ett verktyg för att hitta ännu inte identifierad skadlig programvara. Programmet körs först en kort stund i en virtuell miljö och minnet för miljön analyseras sedan för att hitta skadlig kod.

Genom denna testkörning försöker man ta sig förbi krypteringen som programvaran fått då den packats. Metoden har visat sig vara effektiv och därför har de som utvecklar skadeprogram lagt fokus på att ta sig förbi detta skydd. Detta i sin tur ställer hårda krav på emuleringen av programvaran.

Kraven på emuleringen är många. Emuleringen måste ge bilden av ett riktigt system. Därför emuleras filsystem, register, nätverkstrafik och processer. Emuleringen får inte heller belasta värddatorn eller märkbart fördröja exekveringen av program. Därför måste antivirusprogrammen med dessa krav i åtanke implementera teknik för kontra skadeprogrammens metoder.

Ur Swinnens rapport från 2014 framgår det att de finns stora skillnader i hur väl de olika antivirusprogrammen har lyckats med emuleringen. På grund av de stora skillnaderna mellan de testade antivirusprogrammen ansåg Swinne att det endast fanns värde i att mer noggrant undersöka två av de testade programmen. I

rapporten lyckades Swinne utveckla en metod för att med 100 % effektivitet upptäcka bägge programmens emuleringsmetoder. VARFÖR??? Han påpekar också att en liknande metod alltid kan forskas fram i och med att ett emulerat system alltid skiljer sig på någon aspekt från ett riktigt system. Detta ser jag som en indikation på att tekniken har potential men inte ännu helt lyckas fullföra sin uppgift.

ANTIVIRUSPROGRAM SOM ETT SKYDDSLAGER

Gemensamt för alla tidigare nämnda källor var att man ansåg att antivirusprogram ännu har stora brister och att dessa som sådana inte kan garantera ett systems säkerhet. Den signaturbaserade sökningen har bevisats vara trivial att undkomma och traditionell beteende baserad sökning försvårar, men utgör egentligen i praktiken inget hinder för att attackera system.

Haffejee [5] nämner att någon med ekonomiska motiv kan anses benägen att utnyttja de tekniker han beskriver för att dölja sina skadeprogram. Säkerhetskonferenser som Blackhat och Defcon publicerar deras föreläsares material gratis på sina hemsidor i efterhand. Denna information samt tillgång till antivirusprogram gör att kriminella kan förbättra och sedan testa sina skadeprogram innan de sprids ut.

VAD ÄR DEVICE GUARD?

I detta kapitel redogör jag för vad Device Guard är och vad man vill åstadkomma med funktionen. Hur man implementerar Device Guard och teknikerna man utnyttjar tas upp i efterföljande kapitel.

Device Guard är en ny funktion i Windows 10 som helt ändrar på hur program och operativsystem fungerar ihop. Tidigare har möjligheten funnits att fritt köra all programvara som hittas i systemet. Nu har Microsoft valt att göra en radikal

manöver och sätt till att endast program systemadministratören litar på får tillstånd att exekvera. Program med säkerhetshål eller icke uppdaterad programvara utgör inte mera ett hot eftersom dessa lätt kan blockeras från att köra på systemet. Detta är ett tillvägagångssätt som till stor del liknar det system som används av smarttelefoner idag.

Vad som anses säkert bestäms utgående från vilka mjukvaru- och hårdvarutillverkare systemadministratören och bolaget litar på. Detta kallar Microsoft för Code Integrity (fritt översatt: Kod integritet). Företaget har valt att dela upp detta koncept i tre delar:

- User Mode Code Integrity (UMCI)
  -Kontroll av all programvara som exekveras av systemets användare
- Kernel Mode Code Integrity (KMCI)
  -Kontroll av alla drivrutiner och systemfiler som exekveras i kärnan (kernel)
- Validering av katalogfiler
  -Kontroll av filer som matchar någon känd policy

För att kunna säkerställa att valideringen av kod integriteten inte kan attackeras har man valt att utnyttja virtualisering. Detta kräver givetvis att hårdvaran har stöd för just detta och funktionaliteten kallar Microsoft för Virtual Secure Mode (VSM). [8]

Då systemet startar upp med VSM aktiverat startas en skild minimal kärna som kör viktiga processer vid sidan av Windows egen kärna. Förutom en process som verifieringen av kod integriteten (Hyper-Visor Code Integrity, HVCI) finns även en process som kör LSAS (Local Security Auth Service) och en process som skapar en virtuell representation av TPM (Virtual Trusted Platform Module).

Om systemets hårdvara har en IOMMU (Input/Output Memory Management Unit) och stöd för Second Level Address Translation (SLAT) kan HVCI även sköta om datorns virtuella minne. Detta innebär att program som behöver mera minne måste fråga den säkra kärnan efter nytt virtuellt minne. HVCI granskar om programmet

uppfyller kodintegritetskraven och i så fall ger den säkra kärnan tillåtelse till den traditionella kärnan för att mera minna skall allokeras. Microsoft har satt restriktioner på hur minneshanteringen fungerar. En sida i det virtuella minnet kan numera vara antingen läsbar, exekverbar eller skrivbar. Eftersom den säkra kärnan inte litar på den traditionella kärnan har den säkra kärnan vetorätt när det gäller minnehantering. Vill en applikation läsa, skriva eller exekvera något i det virtuella minnet och den säkra kärnan inte anser att litar på applikationen, är detta inte möjligt och applikationen blockeras.

För att VSM skall fungera måste man ha kontroll över hela uppstarten av systemet. Om denna strikta kontroll inte finns kan någon attackera systemet genom attaker mot hypervisorns fasta program (firmware). Kontrollen fås genom UEFI funktionen Secure Boot (säker uppstart). Secure Boot granskar att alla fasta program som systemets hårdvara kräver har en giltig signatur som endast kan fås av efter att mjukvaran granskas av Microsoft. Om mjukvaran inte uppfyller Microsofts krav tillåts den inte köra på systemet. Secure Boot hindrar även användare från att starta andra operativsystem på hårdvaran, och på så sätt kan man säkerställa att hårdvaran inte missbrukas av användaren.

Virtual Secure Mode uppfanns för att man insett att problemet med tidigare versioner av Windows har varit att alla viktiga säkerhetsrelaterade moduler funnits representerade i kärnan. Lyckades någon ta kontroll över systemets kärna hade den som genomfört attacken direkt tillgång till alla dessa moduler.

VSM kärnan är extremt liten vilket gör att komplexiteten är låg. Kommunikationen mellan den säkra miljön och den vanliga miljön sker genom en begränsad API och alla anrop granskas innan de körs. Den säkra kärnan har även en enkel bild av de applikationer som borde utnyttja den. Allt detta bidrar till att möjligheterna för attacker minskar markant. Tekniken som utnyttjas för själva virtualiseringen är känd och har redan utnyttjats en längre tid av bland annat virtuella servrar. [9]

HUR SKAPAS EN NY POLICY

För att skapa en ny policy till ett system utnyttjar system administratören ett verktyg vid namn New-CIPolicy. Verktyget kan endast köras i kommandotolken och detta kräver att man är noggrann med att alla parametrar är korrekta. Microsofts ingenjörer har valt denna design för de som utnyttjar verktyget måste hålla tungan rätt i mun. En ny policy som är felkonfigurerad kan ha fatala följder för systemet och därför har man även inkluderat verktyg med vars hjälp man testar ens policyn innan man tillämpar dem på det verkliga systemet. Om man endast vill lägga till flera regler till en policy kan programmet New-CIPolicyRule utnyttjas. För att slå ihop flera policyn till en stor policy finns programmet Merge-CIPolicy. Det sistnämnda programmet är behändigt ifall ens bolag till exempel har fyra olika arbetsmaskiner och alla har en egen policy. Då kan systemadministratören skapa en skild policy för varje maskin, slå ihop dessa till en stor policy och sedan tillämpa den stora policyn på alla maskiner.

För att skapa en ny policy krävs vissa parametrar. Först och främst måste man specificera om man vill att policy skall tillåta eller blockera att senare specificerade filer får exekveras. Blockering åstadkoms genom parametern -Deny (neka). Sedan specificerar man sökvägen till filen eller filerna man vill jobba med och sökvägen till den fil man vill att programmet skapar. Som systemadministratör bör man även välja på vilken detaljnivå man vill att den nya policyn skall identifiera filer som hittas. För tillfället finns det 11 nivåer:

1. Hash
2. FileName
3. SignedVersion
4. Publisher
5. FilePublisher
6. LeafCertificate
7. PcaCertificate
8. RootCertificate
9. WHQL
10. WHQLPublisher
11. WHQLFilePublisher

Om filer inte kan identifieras på den specificerade detaljnivån kan man även ge alternativa nivåer.

I det vanliga läget söker programmet endast efter drivrutiner och filer som körs på kernelnivå. Med parametern -UserPEs hittar sökningen även filer som exekveras av användaren.

Då en ny policy skapas måste den ännu konverteras från en XML fil till en exekverbar fil för att kunna tillämpas på systemet. Verktyget som används för detta heter ConvertFrom-CIPolicy och det kräver endast in- och utfil som parametrar. Den exekverbara fil som produceras placeras sedan i ”CodeIntergrity” mappen i ”System32” och exekveras sedan vid nästa uppstart. Istället för att blockera exekvering loggar systemet nu all aktivitet som tillåts eller blockeras av policyn. Detta innebär att systemadministratören manuellt kan gå igenom denna log och granska att allt fungerar som förväntat. Hittas fel kan man uppdatera policyn på basen av loggen och uppdatera den exekverbara policyfilen för fortsatt testning. Då allt fungerar som förväntat skapar man en ny regel i policyn som tvingar systemet att utnyttja policyn fullt ut. Resultatet är en robust välreglerad arbetsmiljö för användaren.

Som en extra säkerhetsåtgärd kan man även signera policyn som skapas och sedan endast tillåta att signerade policyn utnyttjas av systemet. Detta innebär att inte ens en korrupt systemadministratör kan ändra policyn utan att ha tillgång till rätt certifikat och nycklar. [8]

Signering av filer

I dagens läge är mycket av den programvara datoranvändare kommer i kontakt med signerad. Detta innebär att den som publicerat programvaran har ansökt om ett certifikat från en myndighet som har makt att ge ut dessa. För att få ett certifikat krävs att man uppvisar sin identitet och att man garanterar att ens kod inte kommer att innehålla skadlig programvara. Större institutioner och företag kan även anhålla om att få ett grundcertifikat (Root certificate), som möjliggör att dessa på egen hand kan tillverka flera certifikat för egna ändamål.

Iden bakom signering av filer är att användaren på ett enkelt sätt skall kunna verifiera att filerna denne tillhandahållit faktiskt har kommit från utgivaren och att filerna inte modifierats då de kommit till användarens maskin. Signering är också fördelaktig vid uppdatering av befintlig programvara då man kan säkerställa att uppdateringen kommer från samma ursprungliga utgivare.

Verifieringen av koden möjliggörs genom att den som distribuerar programvaran skapar en ensidig hash som representerar den fil som signeras. Denna hash krypteras sedan med certifikatets privata nyckel. När slutanvändaren sedan granskar filen genererar hen även en ensidig hash. Därefter dekrypterar användaren hashen som följde med filen med hjälp av certifikatets publika nyckel och båda hasharna jämförs. Stämmer hasharna överens kan man vara säker på att filen inte modifierats av tredje part.

Det är viktigt att inse att signerad kod inte kan fungera som en garanti för att koden i sig inte är skadlig. Istället skall signering enbart utnyttjas som ett verktyg för att kunna verifiera den ansvariga utgivare och kodens integritet. [10]

KATALOGFILER

Iden med katalogfiler är att Microsoft har velat skapa en metod för att smärtfritt kunna installera äldre osignerad programvara och sedan köra den. I praktiken går detta ut på att man startar ett verktyg som Microsoft gett ut. Verktyget, som heter PackageInspector.exe, samlar upp alla förändringar som sker på filsystemet då programmet installeras och då allt är färdigt skapas en ny katalogfil. Denna fil kan sedan signeras av företaget och sedan placeras i en specifik mapp i systemet. Då det installerade programmet sedan körs granskar Device Guard om filerna finns med i någon av katalogerna i systemmappen och i så fall tillåts programmet exekvera.

Detta tillvägagångssätt innebär att man fortfarande kan utnyttja samma installationsfiler och program utan att behöva packa upp dem och signera allt för att sedan packa ihop allt igen för vidare distribution till ens arbetsmaskiner.

Microsoft rekommenderar dock att företag och institutioner borde övergå till att endast utnyttja signerade applikationer. [8]

HÅRDVARUKRAV

För att kunna skapa en säker virtuell miljö krävs att hårdvaran stöder detta. Saknas någon komponent kan det ändå gå att implementera en del av funktionerna, men systemet som helhet blir sårbart för redan kända attacker. I kommande textstycken behandlar jag de krav som ställs på hårdvaran för att man skall uppnå all funktionalitet Microsoft erbjuder.

Först och främst måste hårdvaran stöda x64-arkitektur samt klara av att köra Windows 10 Enterprise. Microsofts hypervisor fungerar endast 64-bits system och Device Guard samt VSM finns endast i Enterprise versionen av Windows 10.

Stöd för virtualisering innebär att systemets processor stöder antingen AMD-V eller Intel VT-x. I praktiken har de flesta nya processorer på marknaden stöd för någon av de tidigare nämnda teknikerna i och med att virtuella system blivit vanligare och att detta satt krav på hårdvarutillverkarna. Den nya minneshanteringen i VSM kräver en IOMMU som stöder SLAT, vilket de flesta processorer med stöd för virtualisering även har. [11] [12]

Moderkortet måste ha stöd för UEFI (version 2.3.1 eller högre) med både Secure Boot och Platform Secure boot. Stöd för säker uppgradering av firmware för hårdvara måste också finnas för att systemet inte skall kunna attackeras vid uppdateringar. Då en kund investerar i ny hårdvara kan denne givetvis granska specifikationen för hårdvaran. Annars har Microsoft lanserat mjukvara som testar om dessa krav är i skick, eftersom det annars är ganska komplicerat att hitta klara specifikationer om vad som exakt krävs av systemet. [8]

HUR EFFEKTIVT ÄR DEVICE GUARD OCH VSM SOM ETT SÄKERHETSLAGER?

Alex Ionescu har analyserat de nya säkerhetsfunktionerna Microsoft levererar med Windows 10. Ur hans analys framgår det att Microsoft verkligen satt ner tid på att bygga ett vattentätt system. Själv lyckades han inte hitta några direkta svagheter i strukturen. Han påpekar att det kan finnas buggar i den säkra kärnan, men i och med att denna är så liten är sannolikheten stor att Microsoft granskat koden väl.

De stora risker han anser att finns hos denna teknologi är att så mycket av funktionaliteten bygger på att Secure Boot och att den säkra kärnan är beroende av den traditionella kärnan. Hittas ett säkerhetshål i Secure Boot eller UEFI faller hela strukturen i och med att hypervisorn kan kontrolleras eller ersättas med en egen skadlig hypervisor. Som tur kan UEFI uppdateras genom Windows uppdaterings funktion och Microsoft har satt press på hårdvarutillverkarna för att få kända säkerhetshål fixade.

Att den säkra kärnan är beroende av den traditionella kärnan innebär att man teoretiskt kan utföra en denial-of-service attack mot systemet genom att belasta någon av kärnorna. Ionescu påpekar dock att en dylik attack är svår att genomföra och att den som utför attacken har svårt att komma åt någon information på systemet. [9]

Malwaretech har även analyserat och testat Device Guard och dom kom till samma slutsatser. Malwaretech nämner även att Device Guard effektivt blockerar sociala attacker där en systemanvändare luras att köra skadlig kod. Malwaretech påpekar även att även att det handlar om ny teknik som inte ännu testats ordentligt i den verkliga världen och jag har själv hittat exempel på attacker mot liknande system. [13] [14]

På det stora hela anser jag att Microsoft lyckats höja ribban för vad som krävs för

att attackera ett system som skyddas av Device Guard. Om man kompletterar Device Guard med andra säkerhets åtgärder som till exempel antivirusprogram når man en helhet som är svår att knäcka. Att konfigurera Device Guard samt se till ens hårdvara stöder funktionaliteten tar givetvis tid. Men jag anser att denna investering ändå betalar in sig med tiden.

KÄLLFÖRTÄCKNING

[1] Ponemon Institute LLC, "The Cost of Malware Containment," 2015.

[2] C. Hallum, 21 mars 2015. [Online]. Available: https://blogs.windows.com/business/2015/04/21/windows-10-security-innovations-at-rsa-device-guard-windows-hello-and-microsoft-passport/.

[3] Microsoft, ”Device Guard overview,” 14 januari 2016. [Online]. Available: https://technet.microsoft.com/en-us/library/dn986865(v=vs.85).aspx.

[4] H. S. K. a. Orathai Sukwong, "Commercial Antivirus Software Effectiveness: An Empirical Study," IEEE Computer Society, 2011.

[5] B. I. Haffejee J., "Testing antivirus engines to determine their effectiveness as a security layer," in *Information Security for South Africa (ISSA), 2014*, Johannesburg, 2014.

[6] Eset, ”Eset VIRUS RADAR,” 2 februari 2016. [Online]. Available: http://www.virusradar.com/en/glossary/packer-crypter-protector.

[7] A. M. Arne Swinnen, ”One packer to rule them all: Empirical identification, comparison and circumvention of current Antivirus detection techniques,” tekijä: *Black Hat USA 2014*, Las Vegas, 2014.

[8] Microsoft, ”Device Guard deployment guide,” 28 Januari 2016. [Online]. Available: https://technet.microsoft.com/en-us/library/mt463091(v=vs.85).aspx.

[9] A. Ionescu, *Battle Of The SKM And IUM: How Windows 10 Rewrites OS Architecture,* Black Hat 2015, 2015.

[10] Sslshopper, ”What is Code Signing?,” 29 Mars 2016. [Online]. Available: https://www.sslshopper.com/what-is-code-signing.html.

[11] National Security Agency Attention: Information Assurance Solutions Group, ”Defense in Depth,” *A practical strategy for achieving Information Assurance in today’s highly networked environments.*