Tel. 1843
El. p. cert@nksc.lt
https://www.nksc.lt

NACIONALINIS KIBERNETINIO SAUGUMO CENTRAS
PRIE KRAŠTO APSAUGOS MINISTERIJOS

INFORMACINIS BIULETENIS
ATVIRKŠTINĖ KENKĖJIŠKO KODO INŽINERIJA – I DALIS

2023 m. gruodžio 22 d.

Nacionalinis kibernetinio saugumo centras, vykdydamas kibernetinių incidentų tyrimus, esant poreikiui atlieka atvirkštinės kenkėjiško kodo inžinerijos veiksmus. Virusų, logerių, „Trojos arklių“ ir panašaus pobūdžio kenkėjiško kodo kūrėjai stengiasi kaip įmanoma labiau paslėpti pėdsakus, užmaskuoti tam tikrus raktinius loginius elementus, sąsajų nuorodas (angl. *interface references*) ir kodo identifikacijos elementus. Įvairiais metodais siekiama apsunkinti tyrėjo, siekiančio nustatyti kodo paskirtį ir atliekamas operacijas, darbą. Be to, kodas kuriamas taip, kad kuo ilgiau būtų išvengiama apsaugos sistemų identifikacijos ir paleistas galėtų veikti operacinės sistemos netrukdomas, o kartais nepalikdamas joje jokių pėdsakų.

Dėl didesnės nei įprasta apimties suskaidėme temą į tris dalis.

I dalis – analizės pagrindai:

- Analizės metodų apžvalga;
- Statinės analizės pagrindai, pirminė analizė;
- Naudingi įrankiai.

II dalis – pagrindiniai slėpimosi metodai:

- Įkrovėjai (angl. *loaders*);
- Pakeriai (angl. *packers*);
- Obfuskacija, šifravimas, kodavimas;
- Polimorfinis, metamorfinis kodas.

Tel. 1843
El. p. cert@nksc.lt
https://www.nksc.lt

III dalis – gilioji analizė:

- Dinaminės analizės įrankiai ir metodai;
- Mišrios analizės atvirkštinė kenkėjiško kodo inžinerija.

Biuletenių tipas – techninio pobūdžio. Šis informacinis biuletenis yra pirmoji dalis iš trijų.

_____

Analizės tipai ir įrankiai

Atvirkštinės inžinerijos metu specialistas atlieka kenkėjiško kodo analizę, paprastai skirstomą į du pagrindinius tipus – statinę ir dinaminę. Praktikoje patartina naudoti abiejų tipų metodus tokiu būdu atsveriant jų trūkumus.

Statinė kodo analizė	Dinaminė kodo analizė
Tai analizavimo procesas nepaleidžiant kodo.	Kodas paleidžiamas uždaroje monitoringo aplinkoje.
Reikia turėti kompetencijos daugelyje pagrindinių programavimo kalbų.	Dinaminės analizės metu ieškoma pokyčių failų sistemoje, registruose, procesuose ir tinklo komunikacijose.
Prie šios tipo priskiriama failų atpažinimas (angl. *fingerprinting*), virusų skenavimas, paleidžiamojo (angl. *binary*) failo atvirkštinė inžinerija, failų obfuskacijų ir pakerių nustatymas bei derinimas (angl. *debugging*).	Efektyvi prieš daugelį kenkėjiško kodo tipų, tačiau pažengusios kenkėjiškos programos sugeba atskirti aplinką kurioje yra paleidžiamos ir izoliuotoje aplinkoje/virtualizacijose save sustabdo.
Neefektyvi prieš labiausiai pažengusias kenkėjiško kodo programas ir kodą.	

Tyrimams, pvz. skirtingų kenkėjiško kodo iteracijų susiejimui, atakuojančių grupuočių susiejimui ir pan., taip pat ir atvirkštinės inžinerijos praktikavimuisi, galima naudotis pavyzdžiais (angl.

Tel. 1843
El. p. cert@nksc.lt
https://www.nksc.lt

samples) iš atvirų duomenų bazių internete. Kenkėjiško kodo pavyzdžių šaltiniai, kuriuose pavyzdžiai prieinami užsiregistravus ar/ir viešai pateikus savo pačių pavyzdžius:

1. „VirusShare“ (https://virusshare.com)
2. „Malware“ (https://bazaar.abuse.ch)
3. „MalShare“ (https://malshare.com)
4. „Any.Run“ (https://app.any.run)
5. „Hybrid Analysis“ (https://www.hybrid-analysis.com)

Nemokami pavyzdžių internetiniai šaltiniai:

1. „TheZoo resources“ (https://github.com/ytisf/theZoo)
2. „Malware traffic analysis“ (https://www.malware-traffic-analysis.net)
3. „Tek Defense“ (http://www.tekdefense.com/downloads/malware-samples)
4. „Awesome Open Source“ (https://awesomeopensource.com/project/InQuest/malware-samples)
5. „Contagio Dump“ (http://contagiodump.blogspot.com)
6. „Malware Samples“ (https://github.com/fabrimagic72/malware-samples)

Įrankiai kenkėjiško kodo išgavimui:

7. Iš disko atvaizdo: FTK Imager, Belkasoft, Magnet;
8. Iš atminties: DumpIt , MemDump , FTK Imager, Belksoft Ram Capturer, Magnet Ram Capturer;
9. Tinkle: Wireshark, NetworkMiner;
10. Kiti: Kape (https://www.kroll.com/en/services/cyber-risk/incident-response-litigation-support/kroll-artifact-parser-extractor-kape).

Atkreiptinas dėmesys, jog minėtieji pavyzdžiai yra tikros kenkėjiškos programos, kurios gali padaryti žalos jūsų sistemai, taigi jų paleidimą reiktų atlikti tik specialiai tam paruoštoje aplinkoje (pvz. virtualioje sistemoje).

Tel. 1843
El. p. cert@nksc.lt
https://www.nksc.lt

Statinė analizė

Apie statinius metodus

Pirminės statinės analizės metu siektina nustatyti ar failas yra kenkėjiškas išsiaiškinant pagrindines failo savybes, tokias kaip preliminarų funkcionalumą, sertifikatus, importus, kodo kompiliavimo datą ir pan. Pagrindinis šių veiksmų tikslas – pagal surinktą informaciją sukurti indikatorių (angl. *Indicator of Compromise*, trump. *IOC*) sąrašą ir jį toliau naudoti tyrimuose.

Gilioji kodo analizė skirta išnagrinėti išeities kodui ir detaliai nustatyti funkcionalumą. Detalios analizės pavyzdį pateiksime trečiojoje dalyje.

Statinės analizės privalumai:

- Galima ištirti visas paleidžiamojo failo dalis;
- Užvėlintų funkcijų aptikimo galimybė;
- Nereikia laukti sinchronizacijos ar šalutinių paleidimų;
- Galima stebėti kintamuosius skirtingose paleidžiamojo failo vietose;
- Nepriklausoma nuo aplinkos: tinklo, sąsajų, operacinės sistemos;
- Dažnai greitesnis nei įprastas paleidimas.

Trūkumai:

- Dažnai nėra konkrečių verčių ar atminties turinio;
- Sudėtinga išgauti informaciją apie komunikacijas;
- Kartais analizė gali užimti ypač daug laiko;
- Reikalinga daug ekspertinių žinių.

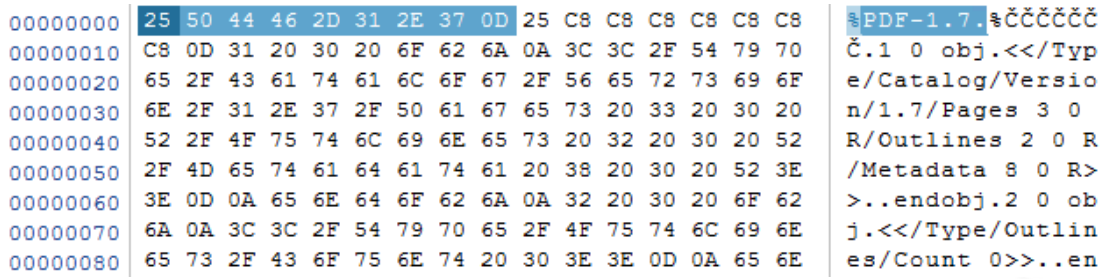
Tel. 1843
El. p. cert@nksc.lt
https://www.nksc.lt

Pirminė statinė analizė

Failų identifikacija vykdoma pirmiausia nustatant jo signatūras, tipus bei identifikacines sumas tam panaudojant maišos (angl. *hash*) funkcijas, pvz. MD5, SHA-1, SHA-256. Identifikavus failą, analitikas pasirenka kokius įrankius naudoti tyrime. Failai skirstomi į dvi pagrindines kategorijas: tekstiniai (angl. *plain text*) ir struktūriniai (angl. *binary*). Tekstiniai failai struktūros neturi, o jų turinys yra tiesiog tekstas kurį galima perskaityti pasinaudojant tekstiniu redaktoriumi. Struktūriniai failai turi griežtą binarinę struktūrą.

Pvz. PDF failai standartiškai turi tokią struktūrą:

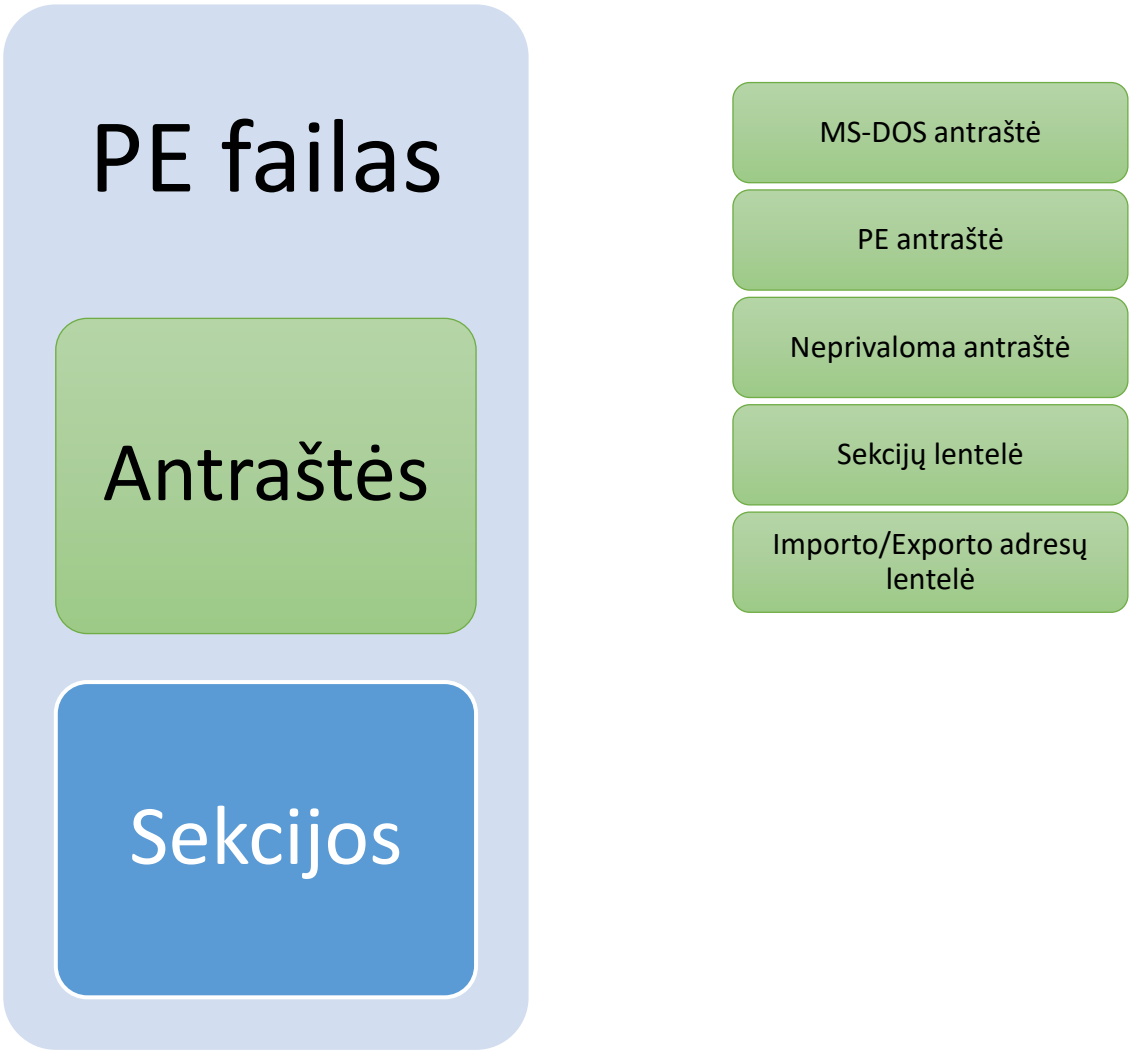
- Antraštė
  - Versijos numeris
- Korpusas
  - Puslapio objektai
  - Paveikslėlių objektai
  - Šrifto objektai
  - Skirtukų objektai
  - Formų objektai
  - ...
- Turinio lentelė
  - Failo objektų lentelių vietos – laisvosios kreipties atminčiai (RAM)
- Galūnė
  - Kai kurių objektų vieta failo korpuse
  - Turinio lentelės vieta faile



Tel. 1843
El. p. cert@nksc.lt
https://www.nksc.lt

Analitikas signatūras rankiniu būdu gali identifikuoti naudodamasis signatūrų sąrašu kurį galima rasti: https://en.wikipedia.org/wiki/List_of_file_signatures

Tuo tarpu paleidžiamojo failo (PE) struktūra standartiškai yra tokia:



Tel. 1843
El. p. cert@nksc.lt
https://www.nksc.lt

Svarbiausia informacija kurią galima išgauti iš PE antraštės yra:

- Importai – funkcijos iš kitų bibliotekų kurias naudoja kenkėjiškas kodas;
- Eksportai – funkcijos kurias kenkėjiškas kodas išveda panaudoti kitų programų ar bibliotekų iššaukimui;
- Laiko žymė – laikas kai programa buvo sukompiliuota;
- Sekcijos – failo sekcijų vardai ir atminties rėžiai diske;
- Subsistema – nurodo ar programa yra komandinės eilutės tipo ar naudoja grafinę sąsają;
- Resursai – atveria eilutes (angl. *strings*), ikonas (angl. *icons*), meniu ir kitą informaciją esančią faile.

.data sekcija | .text sekcija
.rdata sekcija | .bss sekcija
.edata sekcija | .reloc sekcija
.idata sekcija | .rsrc sekcija

Tuo tarpu svarbiausios sekcijos PE faile yra šios:

- .text – talpina paleidžiamąjį kodą;
- .rdata – globalių skaitymo tipo (angl. *read-only*) kintamųjų duomenys prieinami visai programai;
- .data – saugomi globalūs duomenys;
- .idata – čia kartais saugoma importo funkcija. Jei šios dalies nėra, ieškokite importų informacijos .rdata sekcijoje;
- .edata – kartais čia saugoma eksporto funkcija. Jei šios dalies nėra, eksporto funkcija saugoma .rdata sekcijoje;
- .pdata – ši sekcija būna tik 64-bitų PE failuose ir saugo išimčių tvarkymo informaciją (angl. *exception-handling*).

Kitas identifikacinis metodas – nustatyti maišos funkcijų sumas. Tam gali būti panaudojami komandinės eilutės įrankiai md5sum, sha1sum ir sha256sum ar įrankiai su grafine sąsaja, pvz. HashMyFiles (nirsoft) ir Hashe (Eric Zimmerman). Tačiau dažnai kenkėjiškas kodas būna polimorfinis, t.y. jo struktūra keičiasi kodui plintant / platinant. Tokiu atveju reikėtų naudoti

Tel. 1843
El. p. cert@nksc.lt
https://www.nksc.lt

„Fuzzy Hashing“ metodus, kurių esmė, jog kompresijos funkcijos apskaičiuoja panašumus tarp failų. Šios funkcijos išlaiko tam tikrą toleranciją pokyčiams ir lyginant jų išvestį galima identifikuoti pakitusį kodą. Naudotinas komandinės eilutės įrankis „SSDEEP“, kurį galima rasti „Github“ projekte https://ssdeep-project.github.io/ssdeep/index.html.

Apskaičiavus maišos sumas jos tikrinamos viešai prieinamose duomenų bazėse, pvz.:

- Team Cymru: https://hash.cymru.com/
- VirusTotal[1]: https://www.virustotal.com/
- AntiScan: https://antiscan.me

Naudinga informacija kurią reikėtų surinkti analizuojant failą:

- Pakerio identifikacija
- Įtrauktieji resursai (angl. *embeded resources*)
- Kompiliavimo data
- Importai ir eksportai
- Skaitmeniniai sertifikatai
- Svarbios eilutės (angl. *strings*)

Eilutėms išgauti naudojame įrankius tokius kaip „Strings2“ ir „BinText”. Jeigu eilutės užkoduotos (angl. *encoded*), tuomet naudojame „FLOSS“ įrankį, kuris turi funkcionalumą kombinuoti ir automatiškai dekoduoti eilutes naudojant įvairias technikas. Šio biuletenio sudarymo metu „FLOSS“ galima parsisiųsti nuoroda https://github.com/mandiant/flare-floss.

Kitą naudingą informaciją išgauname naudodami įrankius kaip:

- ExeInfoPE – skenuojami failai ir aplankai, informacija apie pakerį, kompiliatorių, GUI versiją. Yra priedų (angl. *plugins*) galimybė;

[1]Būkite atsargūs naudodami „VirusTotal“ tyrimuose ir pratybose, nes šios paslaugos tiekėjas pateikiamus failus ir nuorodas indeksuoja ir saugo duomenų bazėse. Tuomet šie failai tampa prieinami viešu API, kurį gali naudoti ir patys kenkėjai.

Tel. 1843
El. p. cert@nksc.lt
https://www.nksc.lt

- PeStudio – atvaizduoja PE savybes, pažymi anomalijas, apskaičiuoja failo ir jo dalių maišos sumas, išdalina (angl. *parse*) PE failą. Naudingas greitam IOC gavimui;
- CFF Explorer – naudingas atstatyti failo antraštę ar kitas dalis;
- Pescanner, Peframe – Linux sistemų komandinės eilutės įrankiai;
- Signsrch – įrankis skirtas aptikti kriptografiniams metodams, suspaudimui ir kitiems įtartiniems veiksmams, kurie signalizuoja kenkėjiško kodo būvį.
- Pescan – identifikuoti įtartinus PE failus;
- MASTIFF – išgauti informaciją iš kenkėjiško kodo programos žurnalinių įrašų pavidalu; https://git.korelogic.com/mastiff.git/
- Exfiltool – atvaizduoti failo metaduomenis;
- AnalyzePESig – analizuoti paleidžiamųjų failų signatūras;
- Objdump – GCC įrankių rinkinio dalis, skirta išgauti sekcijų, importų, įkrovimo ir kt. informaciją. Komandinės eilutės įrankis. Turi išardymo (angl. *disassembly*) galimybes;
- Dumpbin – panašus į įrankį „Objdump“, tačiau įeina į „MS Visual Studio“ įrankyną. Išardymo (angl. *disassembly*) galimybių neturi.

Siūlomas praktinis uždavinys specialistui

1. Išsirinkti bet kokį failą savo sistemoje (arba dirbant izoliuotoje saugioje (virtualioje) aplinkoje parsisiųsti kenkėjiško kodo pavyzdį);
2. Pasinaudojant pasirinktais įrankiais išgauti IOC;
3. Sudaryti YARA taisyklę skirtą atpažinti failui pagal indikatorius. Daugiau informacijas apie tai: https://www.varonis.com/blog/yara-rules/ ir https://github.com/Yara-Rules/rules
4. Pasinaudoti LOKI skeneriu failo aptikimui pagal anksčiau sudarytą YARA taisyklę https://github.com/Neo23x0/Loki.