Bezpečnost v IT

6. přednáška

Radek Janoštík

Univerzita Palackého v Olomouci

21. 3. 2024

Outline

- Aktuální (kyber)bezpečnostní situace
- Škodlivý software
- Prevence před škodlivým softwarem
- Nedůvěryhodná fyzická zařízení

Aktuální (kyber)bezpečnostní situace

- Google Chrome chce rozšířit funkcionalitu Safe Browsing
  - Přesun „lokálních blacklistů“ na servery Google (krátká životnost podvodných webů)
- Francouzský úřad práce napaden – odcizeny osobní údaje za 20 let (43mil osob)
  - Jména, data a místa narození, adresy, emaily, telefony, čísla soc. pojištění, ...
- Projekt Tor přichází s rozšířením WebTunnel
  - Maskování provozu Tor do HTTPS provozu $\Rightarrow$ horší detekce a cílené blokování
- Byly hacknuty emaily Mezinárodního měnového fondu
  - Zatím tutlají podrobnosti
  - Hostovali u Microsoft 365 – více úniků v posledním roce (samotný MS, HPE), údajně ruská skupina MidnightBlizzard
- Správce tržnice s odcizenými údaji E-Root Sandu Boris Diaconu odsouzen k 42 měsícům
- Člen skupiny LockBit Michail Vasiljev (zadržen 2022) musí zaplatit pokutu 860 tisíc dolarů
  - Přiznal se, našli u něj seznamy potenciálních obětí, screenshoty, ...

Fungování software

- Dnešní software jsou velmi rozsáhlé balíky se stovkami binárních souborů
- Jak můžeme zaručit, že software dělá jen to, co uživatel chce?
- Můžeme například ověřit původ software (digitálně podepsané binární soubory, dll)
- I přes to není zaručeno, že SW bude dělat co má
  - Může nadměrně vytěžovat zdroje
  - Být nestabilní
  - Obtěžovat uživatele dialogovými okny, ...
- Důležitá je příčina nezamýšleného chování
  - Chyba v SW, špatná optimalizace
  - Špatný návrh SW (UI, „flow“)
  - (Zlomyslný) záměr vývojářů
  - Napadení SW, podvržení

Malware

- Dnes se budeme zabývat pouze úmyslnou (nechtěnou) funkcionalitou
- Malicous Software $\Rightarrow$ Malware
  - = SW, který dělá něco, co uživatel nechce (by nechtěl), aby dělal
- Co vše může škodlivý software dělat?
- Vše, na co má oprávnění uživatel, který jej spustil
  - Síťová komunikace
  - Zatěžování zdrojů (dnes populární těžba kryptoměn)
  - Zobrazování reklamy, vyskakovací okna
  - Obtěžování uživatele (zabíjení programů, „psaní na klávesnici“, . . .)
  - Čtení/mazání disků
  - Odposlech (snímky obrazovky, stisky kláves, běžící programy, . . .)
- Spousta lidí si neuvědomuje, jakou důvěru v SW vkládají

Typy škodlivého software (1 / 2)

- Vir – program, který modifikuje ostatní programy
  - Vkládá do nich škodlivý kód
  - Často i sebe sama $\Rightarrow$ šíření
  - Existence po dobu nakaženého (*transient*) nebo samostatně (*resident*)
- Trojský kůň – za chtěnou funkcionalitou schovává ještě nějakou nechtěnou
  - Často malé utilitky (editace videa, stahování, . . .)
  - Ale jiný záměr – šíření jiných virů, odposlech, . . .
- Zadní vrátka(backdoor) – skrytá funkcionalita programu
  - Obejití bezpečnostních mechanismů, nečekaný přístup
  - Většinou implementovány vývojáři nebo potají přidány do zdrojových kódu
- Červ – vir, který šíří své kopie přes síť
- Spyware – špehovací SW (klávesnice, zvuk, kamera!, citlivé fotky)

Typy škodlivého software (2 / 2)

- Adware – zobrazování reklamy uživateli
  - Často napadení prohlížeče – modifikace kliku na odkaz
  - Vkládání kódu do stránek – uživatel nepozná zdroj reklam $\Rightarrow$ zisk k útočníkovi
- Ransomware – způsobí škodu uživateli a požaduje výkupné za napravení
  - Dnes velmi populární – platba v kryptoměnách (napadení společnosti Canon 2020)
  - Otázkou je, zda ransomware „dodrží slovo“
  - Často se pouze tváří, že provedl napravitelnou operaci
  - I jiné výkupné než peníze (Nvidia 2022)
- Kontrola počítače – často tiché vyčkávání na *trigger*
  - poté spuštěn škodlivý kód – botnety
- *Jokeware* – software pro „vystřelení si z kolegy“ – nekomerční původ virů
  - Např. *fork bomba*
- Velmi často je škodlivý SW kombinací více typů – šíření + škoda (+ touha po zisku)

Stručná historie malware

- Myšlenka škodlivého SW se nezrodila v 90. letech
- Známé i výrazně dříve
  - Fred Cohen 1984 – článek COMPUTER VIRUSES: THEORY AND EXPERIMENTS
  - $\approx$ 1979 zmínky o škodlivém kódu, který vkládá kompilátor (U.S. Air Force)
- V 90. letech pouze rozšíření (boom počítačů)
  - Později boom s příchodem internetu
  - Dnes internet hlavním kanálem pro šíření
- 1949 – John Von Neumann – první program, který reprodukuje sebe sama
- Existuje nějaký teoretický základ pro „reprodukci programu“?
  - Věta o rekurzi – 1939 – Stephen Kleene
  - Quine – program, který vypíše svůj kód a skončí

Kvalitní virus (z pohledu útočníka)

- Obtížná detekce – uživatelem i antivirovými programy
  - Vytížit všechny zdroje? Nebo jen někdy?
  - Jak moc okatě dělat svůj záměr?
- Obtížné zneškodnění – stačí odstranit „pár“ instrukcí z binárního souboru?
  - Morfovací techniky, „zažrání se do systému“
- Rychlé šíření (lokální i síťové)
- Náročnost vytvoření
  - Jednoduchý vir na milion počítačů („třeba to vyjde“)
  - Komplexní vir cílený na specifickou platformu
- Platformová nezávislost – omezení OS, HW architektury
- (Hlavní) užitečnost pro tvůrce (monetizace)

Šíření malware

- Dříve vyměnitelná media (pirátské kopie programů)
- Přelom tisíciletí – email – přílohy (`prezentace.exe`, `lechtiveVideo.exe`, ...)
  - Lehká blokace – spam filtr, emailový klient
  - I přes to velmi populární a účinné (i dnes – PDF přílohy)
- Pirátský SW přes internet
- Sociální inženýrství – přesvědčení osob, že jste někdo jiný
  - Phishing – falešná technická podpora
  - Falešná instalační média (fyzický disk poštou)
- Šíření přes sdílené disky (Samba)
- Chyby v síťových aplikacích
  - Bezpečnostní chyby ve webových aplikacích
  - Chyby v síťových démonech

Šíření viru v systému

- Samotný virus na disku nedělá nic – musí se spustit
  - Důležité je první spuštění
  - Pak už může virus pracovat na svém šíření
- Připojení svého kódu před cílový program (obrázek)
  - Stačí rozumět hlavičkám binárních souborů
  - Naleznou první instrukci cílového programu a vložit před ni svůj kód
  - Ideální po svém kódu spustit i cílový program (uživatel nemusí poznat nákazu)
  - Útočník nemusí znát cílový program – univerzální
- Obklíčení cílového programu – škodlivý kód při spouštění a ukončení
  - Cíl: snížení detekce viru – může být v šifrovaném souboru
  - Pří spuštění pouze obecný kód pro rozšifrování
  - Načten do paměti, spuštěn, smazán z disku
  - Při ukončovaní – uložení šifrované kopie na disk
- Integrace do aplikace
  - Horší detekce, nutná znalost konkrétní aplikace

Oblíbené cíle virů

- Boot sektor – OS je také program na disku – něco jej musí spustit
  - Firmware počítačů načte fixní délku kódu z boot sectoru a spustí jej
  - Normálně by se začal načítat OS a všechny jeho komponenty
  - Nakazí-li virus boot sektor může převzít absolutní kontrolu nad strojem
    - Může přemostit jakékoliv systémové volání
    - Simulace zdravého boot sektoru (obtížná detekce)
- *Resident routines* – funkce OS, které jsou trvale načtené v paměti
  - Nenačítají se z disku znovu a znovu (šetření času)
  - Např. obsluha klávesnice, ošetření chybových stavů, ...
- *Excel makro viry* – krátký vir, spouštějící dlouhé makro
- Knihovny – některé široce rozšířené (kauza log4j)
- Samotný operační systém (infiltrace škodlivého kódu do linuxového jádra)

Detekce malware

- Jak rozpoznat, že daná aplikace je škodlivá či nikoliv?
- Jak byste programovali antivir?
- Kód viru musí být někde uložen a po spuštění musí být v paměti
- *Nějak* jsou spouštěny, *nějak* se šíří
- *Vzor*, *signatura* specifických virů $\Rightarrow$ můžeme prohledávat disk, paměť
  - Pro čtení paměti je potřeba administrátorské oprávnění
  - Často stejné umístění v aplikacích (prvních x bajtů)
  - Kontrola změny velikosti/kontrolního součtu aplikací
  - Podezřelá první instrukce – `JUMP`
- Nutná aktuální databáze *vzorů, signatur*
- Často těžké rozlišit od chtěné aplikace

Odstranění malware

- Po detekci je nutné zastavit všechny běžící instance (zastavit šíření)
  - Kontrola v *nouzovém režimu*
  - Kontrola ještě před bootem OS
- V některých případech triviální – odstraníme několik bajtů z binárního souboru
  - Původní aplikace zůstane plně funkční
- Co když je vir „zažraný“ do více částí aplikace?
  - Podaří se nám odstranit všechny kusy?
  - Bude pak aplikace funkční?
- Nejlépe pak přijít na cestu, kudy se vir do systému dostal

Prevence

- Zodpovědné chování uživatelů na síti (správců/adminů)
  - Technicky můžeme být zabezpečení jakkoliv, ale nejslabším článkem jsou uživatelé
  - Hesla na nástěnce vedle PC, půjčování přístupových karet, „zastupitelnosť“
- Používání sotfwaru z důvěryhodných zdrojů (podepsané)
  - *Reproducible builds* otevřeného software
  - vs. obrazy dockeru, flatpaku, . . .
- Testování méně důvěryhodného softwaru na odděleném stroji (k tomu určeném)
  - Bez disku, sítě
- Kvalitní antivirový SW – dnes už není „hloupý“ skener souborů
  - Aktivní skenování paměti, systémových volání
  - Firewall
  - Napojení se do webového prohlížeče, mailového klienta
  - Pozor! Někdy dělají i HTTPS MITM útok
- Co když je antivir virem?

Klasifikace malware

- Studium virů a malware je vědecká disciplína
  - Antivirové společnosti mají velké týmy bezpečnostních expertů
  - Aplikace zpětného inženýrství → pochopení fungování viru → snazší detekce a odstranění
- Ustanovení taxonomie a klasifikace virů a jejich pojmenování
- `https://docs.microsoft.com/en-us/microsoft-365/security/intelligence/malware-naming`
- vs. různé zlidovělé pojmenování

Taktiky proti detekci

- Po rozšíření prvních antivirů na ně začali tvůrci virů reagovat
- Hon na kočku a myš odstartoval
  - Vynalezeny různé taktiky maskování škodlivého kódu
  - A proti-taktiky, ...
- Šifrování viru (na disku)
- Polymorfní viry – mění(mutují) sami sebe
  - Možná mutace své *signatury* = horší detekce
  - *signatura* mutování?
  - Obfuskace kódu (vkládání zbytečných operací)
  - Změna fixních řetězců, pořadí operací

Intrusion Detection System (IDS)

- Detekční systémy podezřelého provozu na síti/stanici
- Host-based – běžící a sledující konkrétní stroj
- Network-based – monitorují celou síť (např. HW sonda)
- Kontrola používání konkrétních portů
- Kontrola vytížení stanice
  - Použití podobných mechanismů jako antiviry (signatura chování)
  - Většinou možné definovat vlastní pravidla
  - V případě aktivace pravidla informování správce
- *Intrusion Prevention Systems* – mohou i aktivně zasáhnout (zavřít porty, zabít aplikaci)
- Dopad režie kontrolního systému na propustnost

Honeypot

- Falešná aplikace/zařízení sloužící jako návnada „hrnec s medem“
- Plně simuluje prostředí a nechá se napadnout
  - Získává informace o činnosti viru
  - Napomáhá v budoucí obraně
- Český projekt *Honeypot as a Service(HaaS)*
  - `https://haas.nic.cz/`
  - Počátky při projektu Turris
  - Spuštění proxy, útočník komunikuje s jiným serverem
- Další systémy např. `Pentbox`, `OpenCanary`

Nedůvěryhodná zařízení – motivace

- „Can someone tell me how the hell he got into our system?“

Obrázek: `https://www.youtube.com/watch?v=aApTVqeGJMw` – Skyfall

Nedůvěryhodné zařízení – motivace

- „And Bond, do you know where this has been?“
- „Anywhere I should imagine.“
- „Into the sandbox!“

Obrázek: `https://www.youtube.com/watch?v=ycvo4xXa538` – No Time To Die

Nedůvěryhodná zařízení

- Velmi často se zapomíná na fyzické zabezpečení systémů
- Můžeme mít hesel a šifrování kolik chceme, ale dostane-li se útočník k odheslovanému stroji, je zle
- Anketa – kdo má zaheslovaný BIOS? Kdo má OS bez hesla?
- Najdete flashdisk na ulici. Co budete dělat? Kdo se odváží jej vyzkoušet na svém PC?
- S dnešní mírou integrace se funkcionalita flashdisku vleze na velmi málo místa
  - Zbytek se dá využít i připojení dalších (zajímavých) čipů
- USB killer – „flashdisk“ plný kondenzátorů – fyzická likvidace stroje
  - `https://hackaday.com/2015/10/10/the-usb-killer-version-2-0/`
  - `https://usbkill.com/`

Rubber Ducky

- „Obyčejný flashdisk“, ale je to i klávesnice
- `https://shop.hak5.org/products/usb-rubber-ducky-deluxe`
- Můžeme do něj nahrát sekvenci klávesových kódu, které se po zastrčení mají „přehrát“
- Programování pomocí *DuckyScript* `https://github.com/hak5darren/USB-Rubber-Ducky/wiki/Duckyscript`
- Kompatibilní se všemi platformami
- Ukázka

Další zařízení

- USB redukce s integrovaným keyloggerem a vysíláním přes Wi-Fi
  - `https://shop.hak5.org/collections/mischief-gadgets`
- Simulace více zařízení (síť, klávesnice, disk, . . .), skriptovíní – Bash Bunny
  - `https://shop.hak5.org/products/bash-bunny`
- Ethernetový (penetrační) tester jako klíčenka – Shark Jack
  - `https://shop.hak5.org/products/shark-jack`
- Ethernetový (hardwarový) packet sniffer
  - `https://shop.hak5.org/products/bug`
- Toto jsou komerční zařízení „za pár korun“
- Co by se dalo pořídit, vymyslet, vyrobit s mnohem větší motivací

Doporučená četba

- Pfleeger C. Security in Computing second edition, Prentice Hall. 2000. ISBN 0-13-337486-6
  - Kapitola 5 – Program Security (177–226)

- Elisan C., Davis M., Bodmer S., LeMasters A.. Hacking Exposed: Malware and Rootkits: Security & Solutions, second edition, McGraw-Hill Education, 2010, ISBN 978-0-07-182307-4
  - Celá kniha

- Sipser M.. Introduction to the Theory of Computation (3rd edition), Thomson South-Western 2012, ISBN 1133187811, 9781133187813
  - Kapitola 6.1 – Věta o rekurzi (245 – 252)