VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INTELLIGENT SYSTEMS

KRYPTOVIROLOGIE

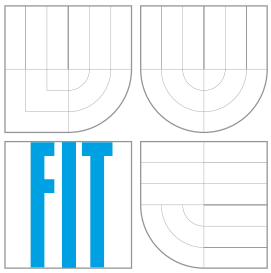
DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. PAVEL KUBÍK

BRNO 2008

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INTELLIGENT SYSTEMS

KRYPTOVIROLOGIE

CRYPTOVIROLOGY

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. PAVEL KUBÍK

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. PETER PECHO

BRNO 2008

Abstrakt

Tato diplomová práce se zabývá studiem relativně nové oblasti počítačové bezpečnosti, která se specializuje na propojení kryptografie s počítačovými viry – Kryptovirologie. Jsou zde analyzovány techniky spojení počítačových virů s moderními kryptografickými algoritmy, způsoby šíření virové nákazy počítačovými sítěmi a také možnosti současných virů a podobných hrozeb. V textu je diskutována nejen problematika kryptografie a klasických virů, ale také metody používané při designu kryptovirů, včetně technik, na nichž je typický kryptovirální útok založen. Jako důkaz proveditelnosti a reálnosti zde popisovaných hrozeb vznikl demonstrační program, který byl implementován s ohledem na splnění základních požadavků kladených na kryptovirus.

Klíčová slova

virus, šifra, kryptografie, algoritmus, kryptografický klíč, bezpečnost, počítačová síť

Abstract

This thesis is focused on a relatively new branch of computer security called Cryptovirology. It uses cryptography and its principles in conjunction with designing and writing malicious codes (e.g. computer viruses, trojan horses, worms). Techniques such as viral propagation through computer networks, capabilities of current viruses and similar threats are described. Beside cryptography and computer viruses, design of the cryptovirus and methods of a cryptoviral extortion attack along with their related potential are also analyzed below in this paper. As a proof of the concept in the given area of cryptovirology, a demonstrational computer program was written. The program was implemented with the respect to the satisfaction of the essentials set to the cryptovirus.

Keywords

virus, ciphertext, cryptography, algorithm, cryptographic key, information security, computer network

Citace

Pavel Kubík: Kryptovirologie, diplomová práce, Brno, FIT VUT v Brně, 2008

Kryptovirologie

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením Ing. Petera Pecha. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Pavel Kubík
16. května 2008

Poděkování

Děkuji Ing. Peteru Pechovi za odborné vedení, cenné rady a za ochotu a vstřícnost při poskytování konzultací v průběhu řešení a zpracování této diplomové práce. Děkuji také svým rodičům za jejich nekonečnou podporu během studia.

© Pavel Kubík, 2008.
Tato práce vznikla jako školní dílo na Vysokém učení technickém v Brně, Fakultě informačních technologií. Práce je chráněna autorským zákonem a její užití bez udělení oprávnění autorem je nezákonné, s výjimkou zákonem definovaných případů.

Obsah

Kapitola 1

Úvod

Bezpečnost informačních technologií je v současnosti velmi skloňovaným pojmem. Na rozvoji bezpečnosti má zásadní vliv vývoj Internetu. Ten dnes mimojiné slouží jako skladiště informací, zábavní centrum a celosvětový komunikační kanál, a v mnoha případech také jako ideální médium k provozování obchodu. Jeho celkový potenciál lze i dnes jen stěží odhadnout.

Pro využití k tolika účelům nebyl Internet dostatečně připraven. Nepředvídatelná dynamika rozvoje a nárůstu celosvětového vlivu způsobila jeho uživatelům mnoho problémů a překvapení. Bezpečnostní incidenty s ohromujícím dopadem a související velké finanční ztráty poškozených firem způsobily, že informační bezpečnost se za posledních deset let dostala do popředí zájmu. Statistiky firem o každoročním navyšování výdajů na bezpečnost jen dokládají rostoucí význam bezpečnosti, jako oboru informačních technologií.

Mezi obory spadající do bezpečnosti bezesporu patří také kryptografie. Ta je dnes součástí mnoha aplikací informačních technologií, aniž by si to běžní uživatelé vůbec uvědomovali. Přítomnost kryptografie v moderních operačních systémech, existence mnoha externích knihoven kryptografických funkcí nebo implementace takovýchto algoritmů do zařízení každodenní potřeby s sebou přináší možnosti, o kterých uživatelé před dvěma desetiletími neměli ani představu.

Příkladem uveďme několik možností, které, více či méně, využívá každý z nás: bezpečná komunikace mezi počítači pomocí tunelového spojení s šifrováním obsahu (SSH), připojování z domácího počítače do školní či firemní sítě (VPN), posílání šifrovaných zpráv elektronickou poštou (PGP), zaručený elektronický podpis sloužící k podepisování elektronických dokumentů, jež některými svými vlastnostmi dokonce, z pohledu bezpečnosti, předčí klasický vlastnoruční podpis, šifrování jednotlivých souborů nebo celých obsahů pevných disků, popř. přenosných USB zařízení (TrueCrypt), používání platebních a kreditních čipových karet v bankovnictví i jinde, a konečně také například telefonování mobilním telefonem v sítích GSM . . .

Ve všech uvedených příkladech našla kryptografie své místo. A mnohé ze zmíněných aplikací by dokonce bez kryptografie nemohly z principu fungovat.

Nyní se na bezpečnost informačních technologií podívejme z jiné strany. Bezpečnostní incidenty, podílející se na zvýšeném významu bezpečnosti v současnosti, jsou vždy způsobeny cílenou činností jistých osob (budeme je v této práci nazývat *útočníci*).

Útočníky jsou zde myšleni nejrůznější narušitelé bezpečnosti, autoři virů nebo podobně nebezpečných programů, vesměs velmi dobře „počítačově“ vzděláni, kteří svou činností sledují různé cíle. Může se tak jednat o pokus nezletilého školáka dokázat si vypuštěním nového viru svoje programátorské schopnosti, ale stejně tak o organizovanou trestnou činnost za

účelem podvodného získání finanční hotovosti od uživatelů elektronického bankovnictví. Svůj podíl na četnosti bezpečnostních incidentů má jistě také konkurenční boj a průmyslová špionáž.

V době, kdy byl Internet ještě jednoduchou počítačovou sítí na akademické půdě, měly první viry v plnění svého poslání snadnou cestu k cíli. Nikdo zpočátku nepočítal s možností nějakého zneužívání takového způsobu propojení pracovních stanic. Naivita prvních uživatelů Internetu v raném stádiu jeho vývoje, nedostatek zkušeností vyplývající z neexistence dřívějších hrozeb – to vše způsobilo ohromný úspěch prvních virů, přestože to byly viry jednoduché, co se strategie šíření týče, nikterak zvláště dobře navržené.

V historii současného Internetu mají počítačové viry „na svědomí“ nemalé finanční ztráty způsobené narušením normálního fungování postižených společností. Ač by se mohlo zdát, že riziko napadení počítače připojeného do sítě je dnes při použití antivirových programů a firewallů minulostí, opak je pravdou. Dokládá to stále rostoucí poptávka po těchto produktech na kompletní zabezpečování počítačových systémů.

Nově vznikají programy spojující výhody kryptografie se způsoby šíření a činnostmi klasických virů a jiného škodlivého softwaru. Představují novou generaci hrozeb, se kterými se musíme naučit vypořádávat. Bývají často důmyslně navrhovány tak, aby dokázaly nepozorovaně vyhledávat a později zneužít cenné a citlivé informace.

Od původních virů, znepříjemňujících svým obětem život, se tak postupně dostáváme až ke kryptovirům, které mohou například po zašifrování určitých dat na disku požadovat zaplacení „výkupného“ výměnou za jejich opětovné zpřístupnění. Namísto relativně málo nebezpečných pokusů o převzetí kontroly nad slabě zabezpečenými systémy, popř. zneužití jejich výpočetních i jiných zdrojů, čelíme činnosti organizovaných počítačových zločinců (expertů), jejichž silnou motivací bývá značný finanční zisk.

1.1 Cíle práce

Tato práce nahlíží na problematiku bezpečnosti z perspektivy tvůrce počítačových virů a zkoumá, jaký bezpečnostní dopad může mít propojení know-how útočníka a moderní kryptografie, a jaká bezpečnostní rizika z tohoto spojení plynou pro uživatele.

Cílem práce bylo studium základů moderní kryptografie s popisem principů používaných ve spojení s počítačovými viry, seznámení se s mechanismy působení virů a způsoby šíření virové nákazy počítačovými sítěmi.

V textu práce je dále obsažen přehled druhů současného škodlivého softwaru, včetně poznatků ze studia okrajových oblastí kontextově zasahujících do záběru tématu. Je zde detailně představen a popsán princip samotného kryptoviru, jakožto relativně nového druhu viru, včetně některých zajímavých vlastností, jimiž se od klasických virů odlišuje.

Získané teoretické poznatky z celé problematiky vedly k naplnění praktické části práce. Jako důkaz proveditelnosti a reálnosti popisovaných hrozeb vznikl demonstrační počítačový program, který má vlastnosti kryptoviru. Tento program je důkazem toho, že lze pomocí dostupných prostředků běžně používaných operačních systémů zneužít jejich velkého potenciálu k aktivitám namířeným proti vlastníkům a uživatelům těchto systémů.

1.2 Struktura a členění textu

Následující kapitola 2 je věnována kryptografii. Jsou zde definovány klíčové kryptografické pojmy, zmíněny hlavní bezpečnostní funkce, stručně popsána její historie a srovnání

s využitím kryptografie v současnosti. Následuje rozdělení kryptografie a popis některých kryptografických nástrojů, kterých se v praxi běžně využívá.

V kapitole 3 je vysvětlen význam slova malware. Budeme se zabývat jeho životním cyklem a definujeme si jednotlivé kategorie, do kterých se malware dělí. Dále je v této kapitole nastíněn pohled na škodlivý software z pohledu antivirových společností, včetně technik, které do svých produktů určených k vyhledávání malwaru implementují. Druhá část kapitoly se bude věnovat modelům a mechanismům šíření virů a analýze nebezpečnosti současných i nově vznikajících virů.

V další kapitole (4) si představíme samotný kryptovirus s charakteristickými vlastnostmi a možnostmi, kvůli nímž je pro tuto práci tolik zajímavý. Srovnáním s klasickým virem naprogramovaným ke stejnému účelu, tj. kryptovirálnímu útoku, ukážeme přednosti kryptoviru a důvod použití právě asymetrické kryptografie u tohoto zvláštního typu malwaru.

Popis průběhu praktické části, tzn. popis analýzy, návrhu, implementace a konečné funkčnosti demonstračního programu, je obsažen v kapitole 5. Protože byla celá kryptografická část implementace postavena na možnostech knihovny Microsoft Cryptography API, bude použitým funkcím věnována celá podkapitola. Dále bude v této kapitole také zhodnocení dosažené efektivity použitých i nově implementovaných funkcí a jejich vliv na chování programu, zmínka o problematice bezpečného mazání souborů a závěrečné zhodnocení celé implementační fáze.

Kapitola 6 bude na téma této práce nahlížet globálně z pohledu bezpečnosti. Často diskutovaným problémem je v bezpečnosti pojem anonymita a jeho chápání. Toho se přímo dotýká oblast narušování bezpečnosti, ať už skrze zneužívání slabých míst v používaných komunikačních protokolech nebo jinými technikami. Na konci kapitoly budou popsána ještě některá bezpečnostní doporučení, kterými lze mnohým incidentům částečně předcházet nebo se jich úplně vyvarovat.

Poslední kapitolou je Závěr. Zde je celkové zhodnocení dosažených výsledků, diskuse o dalších možnostech, budoucnosti a směru vývoje kryptovirologie.

Tato diplomová práce navazuje na stejnojmenný semestrální projekt, kde byly podrobně studovány klíčové oblasti související s kryptografií a problematikou počítačových virů a malwaru obecně. Výsledky práce na semestrálním projektu jsou obsaženy v kapitolách 2 a 3. V implementační části práce jsem se pokusil také ověřit reálnost hrozeb vyplývajících z přítomnosti kryptografických nástrojů v moderních operačních systémech platformy Microsoft Windows.

Kapitola 2

Kryptografie

Studium odvětví souvisejících s počítačovou, resp. informační, bezpečností vyžaduje určitou znalost základních pojmů z kryptografie. V této kapitole budou definovány často používané pojmy a některé principy s nimi úzce související. Některé poznatky k této kapitole byly čerpány z publikací [11, 18, 19].

Jelikož má tato diplomová práce kryptografii částečně již ve svém názvu, je zřejmé, že zde hraje kryptografie velmi podstatnou úlohu.

2.1 Zavedení problematiky

Kryptografie, neboli jinak řečeno šifrování (*encryption* 2.1), se zabývá utajováním obsahu zpráv pomocí šifrovacího klíče do takové podoby, která je čitelná pouze s pomocí určité znalosti (tajemství). Tajemstvím je zde myšleno vlastnictví klíče pro dešifrování. Dešifrování je opačný postup k šifrování (*decryption* 2.2). Je to v podstatě převod zprávy zpět do původní podoby.

Zprávou M (*plaintext, message*) jsou taková data, která jsou normálně čitelná – jsou v nezašifrované podobě. Zašifrovanou zprávou C (*ciphertext*) jsou pak data vzniklá transformací zprávy pomocí šifrovacího klíče K (*key*). Kryptografie slouží především k prevenci a detekci podvodného jednání a jiných podezřelých aktivit.

$$C = E_K(M) \tag{2.1}$$

$$M = E_K^{-1}(C) \tag{2.2}$$

Pomocí kryptografie se snažíme zajistit následující čtyři bezpečnostní funkce [11]:

1. **důvěrnost**
2. **integritu dat**
3. **autentizaci**
4. **nepopiratelnost**

Důvěrnost

Zaručuje, že obsah zprávy bude utajen všem příjemcům, kromě takových, jež jsou oprávněni takovou zprávu přijmout. Přístupů, jak zajistit bezpečnost, existuje několik, od fyzické ochrany, až po matematické algoritmy, jejichž aplikací jsou data transformována.

Integrita dat

Zaručuje, že data nebyla před doručením žádným způsobem modifikována. K zajištění této bezpečnostní funkce musíme mít možnost nějakým způsobem zjistit, zda bylo s daty manipulováno. Manipulaci zde chápeme jako přidání, smazání nebo nahrazení části nebo celé informace.

Autentizace

Funkce autentizace se vztahuje jak na komunikující entity, tak i na samotnou přenášenou informaci. Zpráva přijatá komunikačním kanálem by měla být autentizována, stejně tak jako původce zprávy.

Nepopiratelnost

Nepopiratelností je zde myšlena skutečnost, že komunikující entita po provedení dané akce nemůže tuto aktivitu později popřít.

Definice 2.1.1 *Kryptografie* je věda zabývající se studiem matematických technik souvisejících s aspekty informační bezpečnosti, jako jsou důvěrnost, autentizace, integrita a nepopiratelnost.

2.2 Historie vs. současnost

Původ slova kryptografie je odvozen z řeckého *kryptós* – skrytý a *gráfo* – psát. Použití šifrovacích algoritmů k utajování informací není módní záležitostí posledních několika desetiletí. Nejstarší důkazy o používání kryptografie, byť jen v omezené podobě, pocházejí z Egypta a datují se někdy kolem roku 4 000 př. n. l.

Během několika posledních desítek let však dochází k mohutnému vývoji a použití kryptografických algoritmů. V sedmdesátých letech byl ve Spojených státech amerických uznán algoritmus DES (*Data Encryption Standard*) určený k šifrování neklasifikovaných informací. DES se používá dodnes po celém světě, např. v bankovním sektoru.

Největší tempo nabral vývoj algoritmů v roce 1976. Tehdy Whitfield Diffie a Martin Hellman publikovali svůj článek *New Directions in Cryptography*, kde představili zcela revoluční koncept kryptografie veřejným klíčem. V roce 1978 Rivest, Shamir a Adleman publikovali první použitelný princip pro šifrování veřejným klíčem a elektronické podepisování dokumentů, RSA. Později se objevil algoritmus ElGamal, představený Taherem Elgamalem v roce 1985. ElGamal je algoritmus založený na zcela jiném matematickém problému, než je RSA, který také zaručuje bezpečnost. ElGamal je založený na algoritmu dohodnutí klíče Diffie-Hellman.

Kryptografie se stále více dostává do popředí zájmu, začíná pronikat i do běžného života lidí. Své uplatnění nachází kryptografie všude tam, kde je potřeba přenášet důvěrné informace přes nezabezpečené a nedůvěryhodné prostředí – určené ke čtení pouze oprávněným

subjektům. Takovým nedůvěryhodným prostředím Internet bezesporu je. V historickém kontextu šlo v kryptografii typicky vždy víceméně o vojenské, diplomatické nebo vládní využití. Kryptografie zde slouží např. k předávání strategických informací, koordinaci, komunikaci mezi spojenci, transportu utajovaných skutečností...

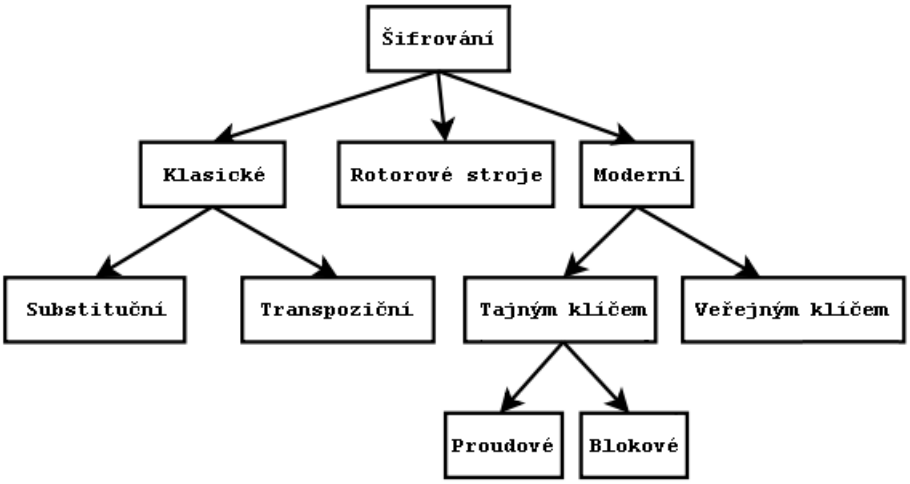
Podobně je tomu s uplatněním kryptografie v současnosti. Algoritmy a jejich aplikace začínají stále více pronikat do civilního sektoru a stávají se nedílnou součástí života lidí i nepřímo pracujících s informačními technologiemi.

2.3 Rozdělení

Někdy se v souvislosti s kryptografií uvádí pojem kryptoanalýza, což je obor zabývající se transformací šifrovaného textu na otevřený, ovšem bez znalosti dešifrovacího klíče. Kryptografii jako takovou rozdělujeme podle typu šifrovacích algoritmů a způsobu použití na kryptografii klasickou (historickou) a moderní.

V klasické kryptografii se používaly jednoduché šifrovací techniky jako substituční, transpoziční a kombinované šifry. Tyto jsou ale při dnešní výpočetní síle strojů velmi snadno dešifrovatelné. Bylo proto třeba hledat moderní postupy, jak zajistit odolnost šifer v současných podmínkách, s výhledem na dobrou použitelnost do budoucna až na desítky let.

Obrázek 2.1 ukazuje rozdělení jednotlivých šifer podle typu a způsobu fungování.



Obrázek 2.1: Typy šifer

Moderní kryptografie využívá v algoritmech sofistikovaných matematických vlastností funkcí, kdy rozluštění šifrované zprávy bez znalosti dešifrovacího klíče je stále natolik výpočetně náročné, že mohou být tyto algoritmy považovány za přijatelně bezpečné.

Podle typu klíče se kryptografie dále dělí na symetrickou kryptografii neboli kryptografii tajným klíčem a na asymetrickou kryptografii, která bývá často označována jako kryptografie veřejným klíčem. Rozdělení podle typu klíče je v kontextu kryptovirologie velice podstatné, proto si symetrickou a asymetrickou kryptografii představíme do podrobnějších detailů.

2.3.1 Symetrická kryptografie

V symetrické kryptografii existuje pouze jediný klíč, který je zároveň jediným tajemstvím navzájem komunikujících stran. Zásadní otázkou je, jak tento tajný klíč bezpečně předat protistraně, se kterou chceme navázat šifrovanou komunikaci. Problém zajištění bezpečné výměny klíče se řeší pomocí bezpečnostních protokolů. Jedním takovým algoritmem je DH (Diffie-Hellman). DH je založen na problému diskrétních logaritmů modulo n. Princip dohody dvou stran (tj. *Alice* a *Bob*) na symetrickém klíči relace je znázorněn posloupností následujících několika kroků:

1. Obě strany, Alice i Bob, zvolí stejné velké prvočíslo n a číslo g, které nedělí n.
2. Alice vygeneruje náhodné přirozené číslo a, Bobovi pošle $X = g^a \, mod \, n$.
3. Bob vygeneruje náhodné přirozené číslo b a Alici pošle $Y = g^b \, mod \, n$.
4. Alice si vypočítá $K = Y^a \, mod \, n$.
5. Bob pak vypočítá $K = X^b \, mod \, n$.
6. Protože platí, že $g^{ab} = g^{ba}$, Alice i Bob se právě dohodli na společném tajemství – $K = g^{ab} \, mod \, n$.

V případě algoritmu DH jsou tajná pouze čísla a a b, každé z nich zná pouze jedna komunikující strana, ostatní hodnoty se posílají komunikačním kanálem v otevřené podobě. Abychom mohli prohlásit nalezení čísla a, resp. b, s pouhou znalostí n, g a $g^a \, mod \, n$, resp. $g^b \, mod \, n$ za výpočetně nezvládnutelné, je nutné volit čísla dostatečně velká (prvočíslo n by mělo mít délku nejméně 300 cifer a čísla a a b alespoň 100 cifer). Naproti tomu nemusí být číslo g nijak velké, v praxi se jako g používají např. hodnoty 2 nebo 5.

V závislosti na konkrétním použitém protokolu může útočník provádět útoky a pokoušet se zjištěnou chybu v návrhu protokolu využít. Existuje celá řada známých útoků na bezpečnostní protokoly, některé z nich jsou popsány v publikacích [13, 20]. Při útoku na protokol výměny šifrovacího klíče se útočník snaží tento klíč získat. Pokud uspěje, stačí už jen následnou komunikaci odposlechnout a získaným klíčem dešifrovat.

Útoky na probíhající komunikaci využívající symetrické kryptografie jsou převážně útoky hrubou silou (*brute-force attack*). Při tomto typu útoku nejde o odchycení klíče, ale o snahu prolomit použitý algoritmus. Tento postup je možné použít prakticky vždy, ovšem při dodržení zásad pro používání symetrické kryptografie a jejích algoritmů je výpočetní a časová náročnost takového útoku velmi velká.

Z bezpečnostních funkcí (uvedených v kapitole 2) zajišťuje symetrická kryptografie pouze důvěrnost. Příklady několika běžně používaných symetrických algoritmů: DES, triple-DES, AES, IDEA, RC4, Blowfish.

2.3.2 Asymetrická kryptografie

Asymetrická kryptografie je založena na principu dvojice odpovídajících si klíčů (soukromý klíč, veřejný klíč). Soukromý klíč existuje v jediné kopii a je uložen u svého majitele – je jeho tajemstvím. Naopak, veřejný klíč je volně přístupný a získatelný pro všechny zájemce o komunikaci. Pokud dvě strany chtějí navzájem komunikovat, každá si musí nejprve opatřit veřejný klíč protistrany a odesílané zprávy se pak šifrují tímto klíčem. Z uvedeného vyplývá, že pouze majitel soukromého klíče odpovídajícího veřejnému klíči, kterým byla zpráva

zašifrována, je schopen ji přečíst. Použití jiného klíče není možné, neboť znalost veřejného klíče nelze použít k odvození klíče soukromého, ani není možné po zašifrování veřejným klíčem získat z výsledku původní zprávu.

Uveďme několik matematických postupů používaných u asymetrických algoritmů a zástupce implementovaného algoritmu: faktorizace čísel (RSA), diskrétní logaritmus (DSA, ElGamal), eliptické křivky (ECDSA).

Algoritmy moderní asymetrické kryptografie jsou postaveny na principech výpočetní složitosti a teorie čísel. Délka klíče zde bývá výrazně větší než v případě symetrických algoritmů. Obecně platí, že čím delší klíč, tím bezpečnější algoritmus. S délkou klíče moderních asymetrických algoritmů, např. 1024 nebo 2048 bitů, výrazným způsobem narůstá bezpečnost šifrované zprávy, tzn. také pravděpodobnost, že nebude prolomena v rozumném čase. Na druhé straně je třeba zvážit nárůst časové náročnosti u implementace s velmi dlouhými klíči. To sice může pro některé aplikace způsobit zvýšení bezpečnosti, ovšem na úkor použitelnosti.

Realita je taková, že ani při dnešní výpočetní síle počítačů není možné použít asymetrickou kryptografii k šifrování větších objemů dat, jako u kryptografie symetrické. Samotné vygenerování dvojice asymetrických klíčů rozumné délky (1024–4096 bitů) je velmi časově náročné.

Asymetrická kryptografie se využívá především pro výměnu klíčů pro symetrickou kryptografii, kterými se pak vzájemná komunikace šifruje. Kryptografie veřejným klíčem zajišťuje bezpečnost při distribuci náhodného klíče protistraně, symetrická kryptografie pak zajistí rychlost při šifrování komunikace dohodnutým klíčem. Rychlost je dána jednoduchostí algoritmu a bezpečnost zaručena (v ideálním případě) zcela náhodně generovaným klíčem.

Jelikož je délka klíčů používaných u symetrických algoritmů výrazně kratší než u asymetrických, což plyne z účelu jejich použití, znamená to, že bývají typicky až o několik řádů rychlejší než algoritmy asymetrické. Nevýhodou je nutnost použití velkého počtu klíčů v případě komunikace mezi více účastníky. Pro N účastníků je potřeba vygenerovat $\frac{N(N-1)}{2}$ tajných klíčů.

Délka symetrického klíče (bitů)	Délka klíče RSA a Diffie-Hellman (bitů)	Délka klíče EC (bitů)
80	1 024	160
112	2 048	224
128	3 072	256
192	7 680	384
256	15 360	521

Tabulka 2.1: Doporučené délky klíčů, NIST

V souvislosti s délkou klíče a výpočetní náročností stojí za zmínku skutečnost, že v případě algoritmů založených na principu eliptických křivek je délka klíče při srovnatelné síle algoritmů výrazně menší než u algoritmů založených na faktorizaci čísel (např. RSA). V tabulce 2.1 jsou uvedeny organizací NIST (*National Institute of Standards and Technology*) doporučované délky klíčů používané v konvenčních šifrovacích algoritmech, jako je DES nebo de facto nový standard AES, společně s nutnými délkami klíčů pro algoritmy RSA, Diffie-Hellman a eliptické křivky tak, aby byla zaručena ekvivalentní bezpečnost [32].

Při použití asymetrické kryptografie k šifrování, tj. v pořadí použití klíčů: veřejný klíč

→ soukromý klíč, je opět zajištěna z bezpečnostních funkcí pouze důvěrnost. V opačném pořadí použití asymetrických klíčů, tj. soukromý klíč → veřejný klíč, které se používá k podepisování (viz dále), jsou zajištěny bezpečnostní funkce integrita, autentizace a nepopiratelnost.

2.3.3 Hybridní kryptografie

Spojením výhod symetrické kryptografie a kryptografie veřejným klíčem vznikají tzv. hybridní kryptosystémy. U hybridních kryptosystémů jsou povolené kombinace symetrických a asymetrických algoritmů předem definovány. Není možná zcela libovolná kombinace. Systémy založené na tomto hybridním přístupu jsou např. SSL, PGP a GPG.

Předmětem této práce je studium možností virové problematiky ve spojení právě s kryptografií. Hybridní kryptosystém je podstatou představeného kryptoviru (kapitola 4).

2.4 Digitální podpis

Digitální podpis se někdy v literatuře uvádí jako zaručený elektronický podpis. Princip obrácení postupu asymetrického šifrování se využívá právě u elektronického podepisování. Nejprve se zašifrováním zprávy (v tomto případě je šifrovanou zprávou hash dokumentu, jenž je elektronicky podepisován) tajným klíčem odesilatele zajistí, že příjemce, který zprávu dešifruje pomocí veřejného klíče odesílatele, bude mít jistotu o původu zprávy. Není tudíž možné podvrhnout zprávu jiným tajným klíčem tak, aby byla dešifrovatelná daným veřejným klíčem.

Na straně příjemce tedy dojde k výpočtu nového hashe z přijatého elektronického dokumentu, ten je porovnán (po dešifrování veřejným klíčem odesilatele) s podepsaným hashem. Porovnáním na shodu obou hashů je zaručeno, že dokument opravdu poslal ten, kdo vlastní tajný klíč odpovídající použitému veřejnému klíči.

Jednou z možných implementací správy asymetrických klíčů je systém kryptografických certifikátů veřejných klíčů (doporučení CCITT X.509). Rozšířenou implementací je protokol LDAP [33]. Veřejné klíče, na rozdíl od tajných, je potřeba distribuovat. Abychom předešli podvržení, zavádíme jejich certifikaci. Příjemce zprávy si ověřuje, že daný veřejný klíč opravdu patří odesilateli. K tomuto účelu slouží certifikáty.

Certifikát je elektronický dokument vydaný důvěryhodnou třetí stranou. Tou je instituce nazvaná certifikační autorita (CA). Je to jakýsi prostředník komunikace. Certifikační autorita připojí k veřejnému klíčí informaci o době platnosti certifikátu a další údaje a podepíše celý dokument svým tajným klíčem. Vytvořený certifikát pošle žadateli (vlastníkovi veřejného klíče).

Certifikát obsahující jak veřejný klíč, tak doplňující informace pro zajištění bezpečnosti může jeho vlastník zpřístupnit volně na Internetu, nechat jeho distribuci na certifikační autoritě, nebo jej posílat společně s každou zprávou, kterou elektronicky podepisuje. Příjemce pak po ověření pravosti certifikátu (ověřování probíhá opět prostřednictvím nějaké CA) použije obsažený veřejný klíč ke kontrole podpisu dokumentu. Spojením symetrické kryptografie k utajení obsahu zpráv a digitálního podpisu využívajícího kryptografii veřejným klíčem můžeme dosáhnout zajištění všech čtyř požadovaných bezpečnostních funkcí.

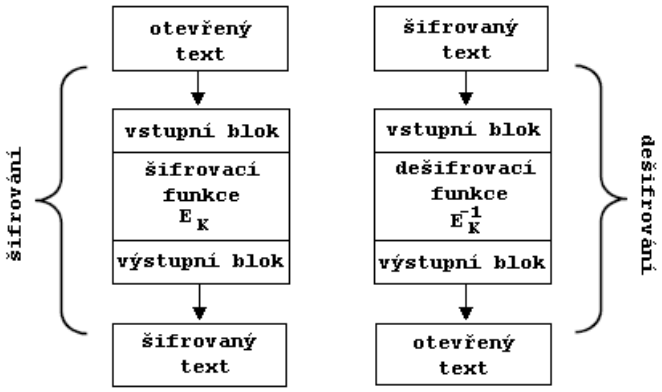
2.5 Režimy blokových šifer

Tato část pojednává o vybraných režimech blokových šifer. Ty se přímo týkají praktické části práce, neboť právě jeden z představených režimů je v implementaci používán. Zmínění dané problematiky je podstatné také z toho důvodu, že nevhodně zvolený režim v aplikacích zaměřených na bezpečnost vede právě k oslabení bezpečnosti, a v důsledku také k nesprávné funkčnosti dané aplikace. Chyby podobného charakteru jsou poměrně závažné, i když nejsou (bez podrobné analýzy návrhu aplikace) ihned patrné.

Blokové šifry pracují nad bloky šifrovaných dat (většinou o velikosti 64 nebo 128 bitů). Pokud velikost dat překročí velikost bloku, jsou data rozdělena do více bloků a šifrována samostatně. Nejběžnější jsou režimy ECB (*electronic codebook*), CBC (*cipher-block chaining*), CFB (*cipher feedback*) a OFB (*output feedback*) [30]. Všechny zde zmíněné režimy blokových šifer poskytují z bezpečnostních funkcí pouze důvěrnost nebo integritu zprávy, ale nikdy obě funkce současně.

2.5.1 ECB

V režimu ECB (obrázek 2.2) je pro daný klíč každý blok dat pevně zašifrován na odpovídající blok zašifrovaných dat. Analogicky jako v kódové knize je šifrováno každé slovo na odpovídající kódové slovo. V tomto režimu jsou stejné bloky dat zašifrovány vždy stejně. Přeuspořádáním zašifrovaných bloků se docílí stejný efekt jako přeuspořádání původních nezašifrovaných bloků dat. Jednotlivé bloky jsou na sobě nezávislé.



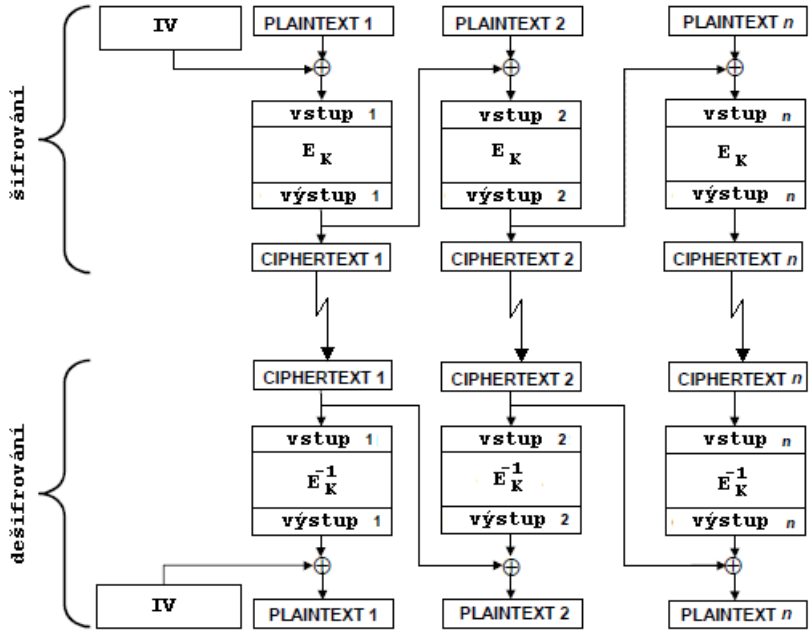
Obrázek 2.2: Režim ECB

Z bezpečnostních důvodů se tento režim nedoporučuje používat pro data s délkou větší, než je délka jednoho bloku, nebo když je klíč používán vícekrát než jen pro jednu jedinou zprávu. Do jisté míry je možné zvýšit bezpečnost ECB režimu přidáním náhodných výplňových bitů (*padding*) do každého nešifrovaného datového bloku.

2.5.2 CBC

Narozdíl od ECB, je v režimu CBC šifra (výstup operace) bloku vždy stejná pro stejná vstupní data jen tehdy, když se kromě stejných vstupních dat navíc použije stejný inicializační vektor (IV) a stejný šifrovací klíč. Utajení dat je zde mnohem lepší, protože zde dochází ke kombinování (řetězení) vstupních dat s výsledkem z předchozí iterace. Změna IV, klíče nebo prvního bloku dat znamená zcela jiný výsledek.

Princip fungování v režimu CBC je na znázorněn na obrázku 2.3. CBC je navíc tzv. samosynchronizující, a to v tom smyslu, že chyba v jednom šifrovaném bloku dat, včetně výpadku celého bloku, ovlivní negativně pouze tento a následující blok. Další datový blok, v pořadí druhý od toho s výskytem chyby, bude již dešifrován korektně.



Obrázek 2.3: Režim CBC

Jako inicializační vektor je vhodné použít náhodně vygenerovanou bitovou posloupnost. Náhodný IV by měl být zcela nepředpovídatelný, tzn. neměl by se lišit od náhodného šumu. Více o generování náhodných posloupností je popsáno v podkapitole 2.6.

Přestože nemusí být inicializační vektor utajen – může být součástí zašifrovaných dat, je doporučeno zajistit alespoň jeho integritu. Modifikace IV umožňuje útočníkovi provést předvídatelné změny jednotlivých bitů v prvním bloku dešifrovaných dat. Utajením IV lze takovým praktikám zabránit.

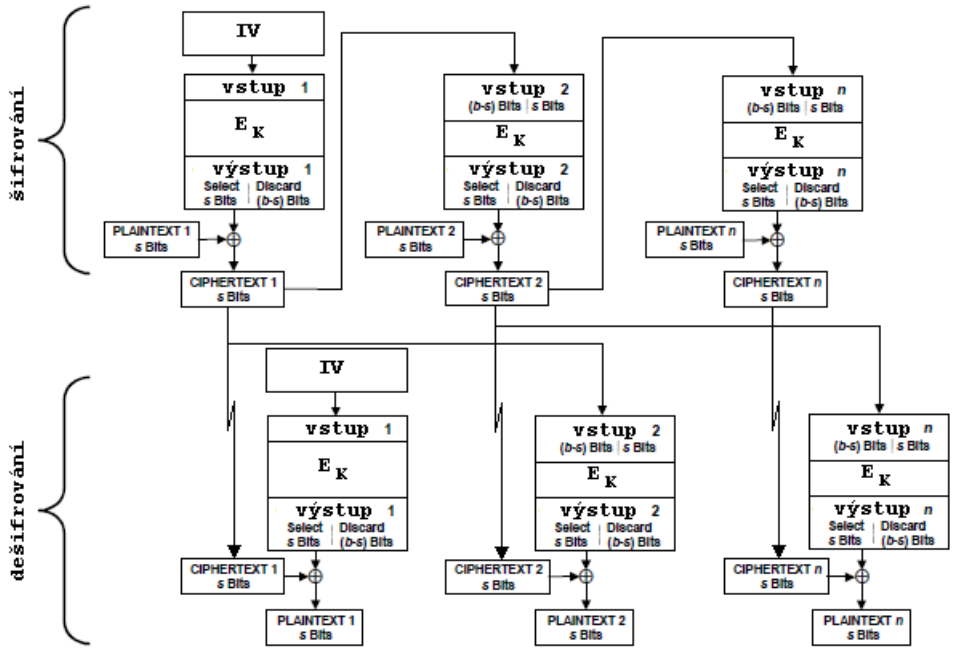
2.5.3 CFB

Režim CFB vkládá opět výstup blokové šifrovací funkce zpět na svůj vstup. Výstupem šifrovací funkce není myšlen přímo výsledek operace – jednoho kola CFB nad blokem dat, ale až výsledek operace XOR tohoto výstupu s otevřeným textem k zašifrování.

Podobně jako CBC, při změně IV dává operace šifrování jiný výstup, i při stále stejných vstupních nešifrovaných datech. Protože je šifrovací funkce v CFB používána zároveň pro šifrování i dešifrování, nesmí být tento režim použit v případě, kdy je blokovou šifrou některý z algoritmů veřejným klíčem. Pro tento případ je vhodné použít namísto CFB režimu CBC. Činnost detailněji ilustruje obrázek 2.4.

2.5.4 OFB

Princip fungování tohoto režimu je velmi podobný CFB s tím rozdílem, že výstup blokové šifrovací funkce je přiveden zpět na vstup. Až poté se provádí XOR s šifrovanými daty.



Obrázek 2.4: Režim CFB

Existují dvě běžně používané verze. První je ISO verze OFB (obrázek 2.5) s úplnou zpětnou vazbou (*full feedback*), která je bezpečnější. Druhou je FIPS verze (starší) OFB s r-bitovou zpětnou vazbou (*r-bit feedback*), kdy se výstup blokové funkce (r nejlevějších bitů) relativně složitě posouvá doprava a pak se z výsledku skládá vstup pro další iteraci.

Za zmínku stojí fakt, že některé operace mohou být předzpracovány, protože výstup blokové šifrovací funkce a šifrovaná data jsou na sobě nezávislá. K předvýpočtům postačuje pouze šifrovací klíč a IV.

2.6 Podpůrné nástroje

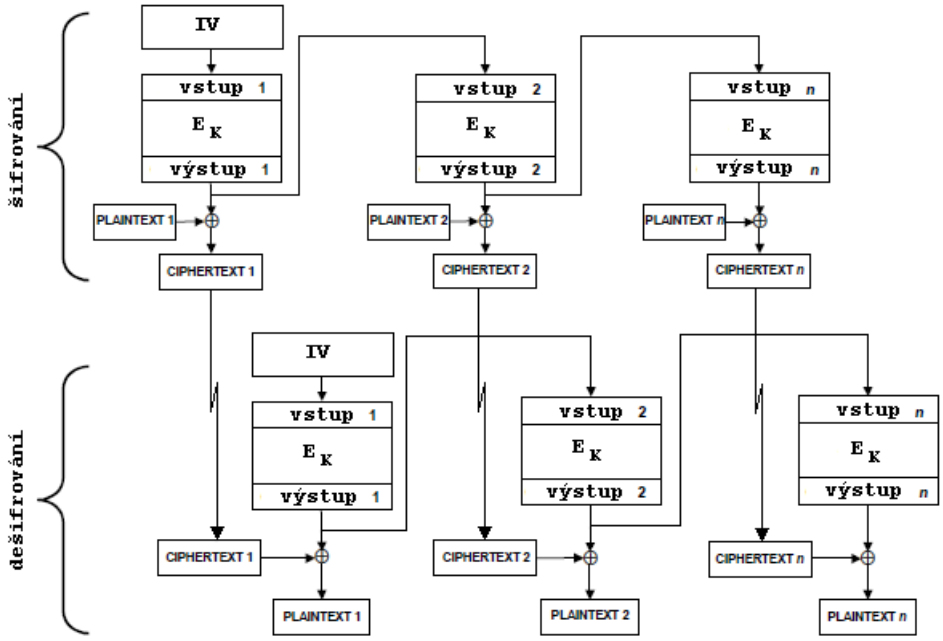
Kryptosystémy a kryptografické protokoly se spoléhají na správnou implementaci podpůrných nástrojů. Nyní se budeme zabývat některými vybranými nástroji, a to takovými, které jsou pro nás z pohledu kryptovirologie nejzajímavější.

2.6.1 Generátory náhodných čísel

Snad nejvíce kritické jsou z pohledu funkčnosti kryptografických algoritmů generátory náhodných bitů – RBG (*Random Bit Generator*). Ty se v kryptografii používají ke generování např. tajných klíčů, inicializačních vektorů nebo paddingu nutného k správné funkčnosti (bezpečnosti) některých algoritmů.

Nezbytnou podmínkou při používání popsaných algoritmů s dosažením požadované bezpečnosti je použití skutečně náhodných posloupností bitů, tedy náhodné výskyty jedniček a nul v generované posloupnosti. Nesprávně generované náhodné posloupnosti mohou vést k uhádnutelným klíčům případným útočníkem.

Spolehlivé RBG pracují na základě získávání dat z náhodné změny fyzikálních veličin. Mezi nejspolehlivější zdroje skutečně náhodných bitů patří zařízení pracující na principu



Obrázek 2.5: Režim OFB

měření radioaktivního rozpadu nějakého izotopu. Každý takový rozpad je zaznamenán elektronickým detektorem, který část energie záření převádí na elektrický signál.

Existující a běžně používané generátory využívají také např. dvojice krystalů – krystalu generujícího takt CPU a krystalu hodin reálného času, typicky jsou oba součástí PC (viz AT & T truerand). Jiným příkladem může být využití chaotických vzduchových turbulencí v pevných discích. Ty mají vliv na dobu vystavení diskových hlav a rotační zpoždění. Kombinací několika podobných hardwarových generátorů a použitím algoritmů na zjišťování, zda entropie generovaných bitů opravdu odpovídá požadavkům na náhodnost, lze dosáhnout spolehlivě náhodných posloupností [25].

Data z RBG se používají jako vstupy pro generátory pseudonáhodných čísel – PRNG (*Pseudo Random Number Generator*). Použití ne zcela náhodných posloupností vede k degradaci bezpečnosti kryptografických algoritmů. Útočník získává jistou šanci uhádnout nebo s určitou pravděpodobností předpovědět následující posloupnost bitů.

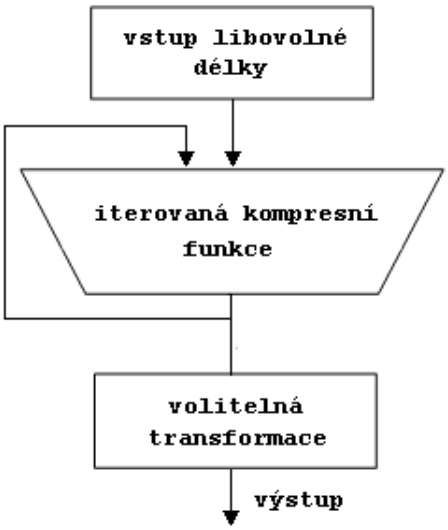
Pouze skutečně náhodná posloupnost může zaručit bezpečnost algoritmů. Chybou, která může vést až k úspěšnému prolomení šifry útočníkem, je použití hashovacích algoritmů (např. SHA-1, MD5, apod.) jako generátorů náhodných posloupností, neboť nelze zaručit míru entropie (viz definice 2.6.1) takto generovaných dat.

Definice 2.6.1 *Entropie* je míra neuspořádanosti, náhodnosti nebo proměnlivosti v uzavřeném systému. Entropie systému X je matematická míra množství informace získané pozorováním systému X. [31]

2.6.2 Hashovací funkce

Hashovací funkce (definice 2.6.2) je speciální funkce, která vrací reprezentativní vzorek daného argumentu. Používá se především u digitálního podpisu a pro zajištění datové integrity. Princip fungování je znázorněn na obrázku 2.6. Zmíněné hashovací funkce jsou veřejně

známé, nepoužívají žádný tajný klíč. Jestliže slouží ke zjištění, zda bylo navíc s daty manipulováno, nazývají se MDC (*modification detection codes*). Příbuzné k těmto funkcím jsou hashovací funkce parametrizované tajným klíčem. Tyto slouží k autentizaci původce zprávy i k zajištění její integrity – rodina algoritmů MAC (*message authentication codes*). Jako zástupce rodiny MAC uveďme algoritmus HMAC.



Obrázek 2.6: Obecný model iterované hashovací funkce

Definice 2.6.2 *Jednocestná funkce* (*One-way function*) je taková funkce f z množiny X do množiny Y, že platí: $\forall x \in X$ musí být výpočetně snadné získat $f(x)$, ale $\forall y \in Im(f)$ je výpočetně nezvládnutelné najít takové $x \in X$, že $f(x) = y$.

Kromě splnění podmínky *preimage resistance* uvedené v definici, klademe na jednocestnou (hashovací) funkci ještě tyto další požadavky:

- Je aplikovatelná na argument (téměř) libovolné velikosti.
- Má výstupní hodnotu konstantní délky n bitů (běžně $n = 160$ nebo $n = 256$).
- Platí *second-preimage resistance*, viz [11].
- Platí *collision resistance*, viz [11].

Algoritmů používaných k získávání jednoznačné řetězcové reprezentace libovolných dat (hashe) existuje několik: MD4, MD5, SHA, SHA-1, RIPEMD aj. U mnoha hashovacích algoritmů byly zjištěny zásadní bezpečnostní nedostatky, proto je do budoucna vhodné používat jejich nástupce. U některých aplikací nemusí být zjištěné nedostatky natolik významné a mohou být dále využívány, toto se týká především aplikací, u nichž nemá relevantnost výstupu hashovacích algoritmů dlouhého trvání (např. síťové protokoly).

Kapitola 3

Malware

V této kapitole se zaměříme na popis různých typů škodlivých počítačových programů. Tak chápeme především počítačové viry, trojské koně, červy a jiné specifičtější druhy. Souhrnně bývá takovýto software produkovaný útočníky nazýván jako malware (*malicious software*).

Aktivity malwaru bývají typicky zacíleny na počítačové systémy s běžícími službami, jež obsahují bezpečnostní slabiny. Jako typický příklad zranitelného systému uveďme stanici s nainstalovaným operačním systémem MS Windows XP bez zahrnutých kritických bezpečnostních záplat. Nezabezpečený systém MS Windows XP, připojený k Internetu, vydrží průměrně 20 minut, než je objeven a úspěšně kompromitován [7].

Nemusí být pravidlem, že škodlivý software se do systému dostane pouze bez vědomí uživatele. Existuje teoretická studie, v níž je dokonce popsán takový typ malwaru (pojmenován jako *Satan Virus*), včetně vlastní propagace, vzájemné komunikace mezi nezávislými instancemi viru, interakce s uživatelem napadeného systému a způsob možné implementace. Program popsaný v této studii je na počátku uživatelem vědomě nainstalován jako software s lákavými funkcemi. Později svoji oběť dokonale ovládá, vydírá ji a nutí jednat podle vlastních pravidel hry [3].

3.1 Co je a není malware

Tato část konkrétněji definuje pojem malware (definice 3.1.1) a poukazuje na rozdíl mezi běžnými a škodlivými programy.

Definice 3.1.1 *Malware* je program vykonávající nějaké škodlivé akce namířené proti uživateli, který byl za tímto účelem vytvořen. Cílem je typické způsobení škody v libovolné podobě, a to na pracovní stanici, serveru nebo počítačové síti.

Skupina softwaru označovaná jako malware tedy reprezentuje všechny obecně známé kategorie škodlivého softwaru, jako jsou viry, červi, trojští koně, rootkity aj. Původně vznikaly tyto programy jako důkaz programátorských dovedností, jako zkouška, kam až je schopen se daný výtvor po síti rozšířit nebo jako žertovné programy k rozptýlení a nevinnému obtěžování uživatele.

Později se objevovaly vandalské programy odstraňující data z disku nebo narušující konzistenci souborového systému. S rozšířením širokopásmového internetu se objevuje stále více malwaru tvořeného s cílem podvodného finančního zisku (spyware, keystroke loggery, dialery).

Poznámka:

Jako malware by neměl být označován software, který obsahuje chyby, ale byl napsán pro legitimní účely.

3.2 Životní cyklus malwaru

Životní cyklus malwaru začíná jeho vytvořením a končí úplným odstraněním z napadeného systému. Následuje popis jednotlivých etap [38]:

1. **Vytvoření**

   K vytvoření nějakého škodlivého kódu postačuje základní znalost libovolného programovacího jazyka a přístup k Internetu. Existují samostatné webové servery sloužící pouze k šíření malwaru nebo vytváření nových verzí z hotových a funkčních verzí.

2. **Replikace a rozšíření**

   Šíření nákazy probíhá více způsoby. Červi využívají e-mail, sdílení souborů nebo IM (*Instant Messaging*). Viry se replikují v rámci hostitelského systému. Trojani se skrývají pod webovými odkazy ke stažení, jsou často součástí e-mailů a pochybných webových stránek.

3. **Aktivace**

   Většina malwaru provádí vlastní činnost po spuštění aplikace. Někdy může být činnost spuštěna automaticky, např. ke zvolenému datu nebo při splnění určité podmínky.

4. **Odhalení**

   Tato fáze typicky následuje po fázi předcházející, avšak nemusí tomu tak být vždy. Po odhalení bývá malware odeslán do ICSA Labs [27], kde je analyzován, zdokumentován a dokumentace o něm distribuována výrobcům antivirových produktů.

5. **Přizpůsobení**

   Antivirové společnosti modifikují svůj vyhledávácí software, aby byl schopen detekovat nový malware. Doba provedení tohoto kroku závisí na jednotlivých antivirových společnostech – pohybuje se od jednotek hodin, dnů, až měsíců.

3.3 Kategorie

3.3.1 Virus

Definice 3.3.1 *Virus* je program, který je spuštěn nějakou akcí uživatele. Poté vytváří své vlastní kopie, případně tyto kopie dokáže i sám modifikovat. Obsahuje nějakou funkci (*payload*), která se vykoná v případě splnění předem definovaných podmínek.

Počítačový virus je tedy zřejmě spustitelný počítačový kód, který je schopen vlastní reprodukce v rámci hostitele. Pokud dokáže virus nějak modifikovat své vlastní kopie – potomky, jedná se o typ tzv. polymorfního viru. Pokud dokáže modifikovat také sám sebe (aktivní instanci programu), nazýváme tento zvláštní druh virů jako metamorfní. Reprodukce, neboli šíření viru z jednoho systému na jiný, je možná pouze jako součást infikovaného systému. Virus se tedy nedokáže šířit zcela autonomně.

Podle typu cílové destinace v hostitelském systému můžeme viry rozlišovat na souborové, boot-sector viry, multipartitní viry atd. Mnohé, vyspělejší formy, dokáží kombinovat zmíněné typy destinací a stávají se pak mnohem odolnější vůči objevení a odstranění z napadeného systému.

Šíření virové nákazy se může ubírat několika cestami – skrze síťové souborové systémy, systémy ke sdílení a výměně souborů (typicky jistý druh P2P sítí), zneužíváním služby WWW (*World Wide Web*) a jistě i mnoha dalšími.

3.3.2 Červ

Definice 3.3.2 *Červ* (*worm)* je program, který, narozdíl od viru, vytváří své vlastní (i modifikované) kopie bez jakéhokoli přičinění uživatele. Propagace nákazy probíhá síťovou cestou, tzn. červ se dokáže šířit z infikovaného systému pouze prostřednictvím jeho síťového připojení. Kód nesený v těle červa může být zcela libovolný.

Červi jsou druh malwaru šířící se samostatně počítačovou sítí. Hlavní rozdíl mezi červy a viry je ten, že první jmenovaná skupina je nezávislá na aktivitě v rámci hostitelského systému.

Jinak řečeno, červi mají autoreplikační schopnost. Posílají kopie sebe sama do sítě na určené síťové adresy, které jsou buď pevně dány nebo jsou náhodně vybrány z celého adresového prostoru. Často se použijí pouze relevantní adresy v rámci sítě, v níž se daný počítač nachází – výběr závisí na implementované logice každého konkrétního typu červa.

V případě šíření prostřednictvím elektronické pošty jsou cílové destinace vyhledávány v souborech na lokálních discích. V minulosti celá řada úspěšných červů tohoto typu dokázala e-mailové adresy vyhledávat v souborech různých poštovních klientů instalovaných na daném počítači, kde byly uloženy kontakty uživatele tohoto počítače.

Obsahem přenášeným v těle červa bývá často jiný, mnohdy zákeřnější kód, který může např. mazat soubory na disku, rozesílat je přes e-mail nebo instalovat tajné přístupové body do systému, tzv. „zadní vrátka“ (*backdoor*). Útočník tak získává navíc možnost se kdykoli v budoucnu k systému vzdáleně přihlásit, téměř neomezeně provádět neautorizovanou činnost a zůstat při tom nepozorován...

Přítomnost červa se může projevovat nedostupností webových stránek výrobců antivirového softwaru nebo únikem licenčních klíčů k instalovaným hrám a aplikacím [38].

Poznámka:

Hranice mezi moderními viry a počítačovými červy se v reálu často stírá. Spojením výhod jednotlivých typů malwaru se jejich autoři snaží posílit schopnosti svých výtvorů. Vznikají stále novější a vyspělejší hrozby. Škody způsobené působením virů, červů a jiných druhů malwaru se odhadují až na 100 miliard $ za posledních dvacet let [9].

3.3.3 Trojan

Definice 3.3.3 *Trojský kůň* neboli trojan (*trojan horse*) je program, který nevytváří vlastní kopie ani se nijak nešíří, pouze provádí činnost, o níž uživatel programu nemá tušení.

Trojský kůň je program, který se snaží předstírat, že je něčím jiným, než ve skutečnosti je. Na první pohled vypadá jako užitečný program, který uživatel jistě náležitě ocení. Po spuštění je realita ale taková, že provádí činnost zcela jinou (nebo může nabízené funkce

opravdu obsahovat a zákeřná činnost probíhá skrytě na pozadí). Poté, co je „nainstalován“ do systému, vykonává činnosti, jako je vytváření zadních vrátek nebo stahování jiného malwaru z Internetu. Nemá replikační schopnosti jako virus, proto je jeho přežití závislé na způsobu, jakým „přesvědčuje“ uživatele, že by měl být spuštěn.

3.3.4 Rootkit

Definice 3.3.4 *Rootkit* je označení programu, který disponuje schopnostmi skrývat svoji přítomnost v systému před uživatelem, ovládat části systému nebo nad ním zcela převzít kontrolu.

Jako rootkity je označována celá skupina programů s podobným chováním. Běžně bývají pomocí antivirových programů nedetekovatelné. Nahrazením originálních bezpečnostních funkcí systému se snaží zajistit vlastní neodstranitelnost. Mezi používané techniky patří skrývání běžících procesů, souborů a dat v operačním systému. Zpravidla s sebou rootkit také zavádí do systému „zadní vrátka“ [37].

3.3.5 Spam

Spam je označení nevyžádané elektronické pošty. V současnosti je to jeden z největších problémů, s nimiž se v Internetu potýkáme. Ačkoli se nejedná o nějaký škodlivý kód, jako předešlé druhy (viz definice 3.3.1, 3.3.2, 3.3.3 a 3.3.4), bývá velmi často k elektronickým útokům využíván.

V době vzniku tohoto označení pro nevyžádaná obchodní nebo podvodná sdělení se spam objevoval převážně v e-mailových schránkách uživatelů. Dnes se problém rozrůstá do několika dalších rozměrů, spam se objevuje už i v diskuzních skupinách, chatu a u komunikačních programů na bázi IM.

Pořadí	Země	Počet známých incidentů
1.	Spojené státy americké (USA)	1 546
2.	Čína	471
3.	Rusko	282
4.	Velká Británie (UK)	206
5.	Jižní Korea	191
6.	Německo	185
7.	Japonsko	160
8.	Francie	157
9.	Indie	126
10.	Kanada	123

Tabulka 3.1: Přehled 10 zemí jako největších původců spamu, k 28. 4. 2008

Dlouhodobé měření procentuálního podílu spamu dokazuje, že okolo 80 % celosvětového oběhu e-mailů je spam. Statistiky dále ukázaly, že nejvíce spamu pochází ze Severní Ameriky, a že asi 200 producentů spamu má na svědomí celých 80 % všech nevyžádaných zpráv mířících do mailboxů uživatelů. Zajímavá data přináší projekt ROKSO (*Register of Known Spam Operations*), což je registr odesilatelů spamu, spamových služeb a známých spamových incidentů [36].

Projekt ROKSO navíc udržuje databázi s klasifikovanými daty, do které mají přístup pouze agentury jako: Scotland Yard CCU (UK); FBI, USSS, US Marshal's Service (USA); OPTA (Nizozemí) a ACMA (Austrálie). Data o nezákonných aktivitách z této DB slouží jako důkazy při soudních procesech. Přehled vybraných zemí podle pořadí v produkci spamu ukazuje tabulka 3.1.

3.3.6 Botnet

Napadený systém, který je pod nadvládou útočníka, se označuje jako *bot* nebo *zombie*. Činností programů, jako jsou červi, trojani, rootkity atd., lze vzdáleně zranitelný systém ovládat. Malware v systému naslouchá na určitých portech počítače. Útočníci mohou kdykoli vzdáleně na tyto porty posílat instrukce (síťové pakety), pomocí nichž činnost svého programu ovlivňují. Počet systémů napadených jedním konkrétním typem malwaru se nezřídka pohybuje v řádech desítek a stovek tisíc kusů. Množina takto ovládaných systémů (botů) se označuje jako botnet [16, 1].

Koordinací celého botnetu dokážou útočníci provádět distribuované útoky na internetové servery DDoS (*Distributed Denial-of-Service*), phishingové útoky nebo rozesílat tisíce podvodných e-mailových zpráv. Kdokoli by se snažil vystopovat původce těchto spamových zpráv, najde spíše nicnetušící oběť jiného útoku než skutečného útočníka. Botnety jsou nedílnou součástí Internetu. Jsou dokonce předmětem obchodování mezi jejich „majiteli" a potenciálními zákazníky, těmi jsou zadavatelé reklamy atp.

Poznámka:

Termín *bot* se někdy používá také pro označení malwaru, který způsobí zmíněné ovládnutí počítače, ovládnutý stroj pak bývá označován jako *zombie*.

3.3.7 Podvodné techniky

Podvodné techniky jsou jakýmsi prostředkem útočníků, jak zneužít zranitelnosti oběti. Slabým místem může být jednak nedostatečné technické zabezpečení, stejně tak ale i povahové rysy člověka sedícího za počítačem.

Mezi podvodné techniky se řadí nejrůznější poplašné zprávy (*hoax*), podvodné stránky snažící se získat přihlašovací jména a hesla nepozorných uživatelů (*phishing*). Procentuální úspěšnost takových technik není zřejmě nikterak vysoká. Porovnáme-li ji však s podobně vyhlížející statistikou úspěšnosti spamu, může být v součtu škoda způsobená i malému procentu všech uživatelů Internetu poměrně vysoká.

3.4 Funkce viru

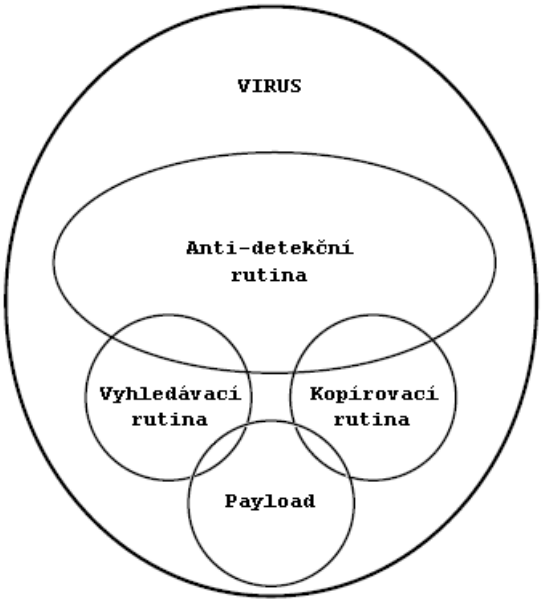
Každý virus se skládá nejméně ze tří, typicky čtyř, částí (tzv. rutin, viz obrázek 3.1) [10]:

- **Vyhledávací**
  Lokalizace dalších potenciálních cílů (souborů) na disku. Rozhoduje o rychlosti šíření virů, zda bude nákaza probíhat na jednom disku nebo více discích. Jako každý program musí v této fázi virus volit optimální poměr – velikost kódu / množství funkcí.

- **Kopírovací**
  Zajištění vlastní reprodukce viru. Složitost této rutiny závisí na typu viru. Čím je sofistikovanější, tím má virus větší šanci splnit poslání a nebýt odhalen.

- **Anti-detekční**
  Definuje chování viru po spuštění. Jistá míra náhodnosti v chování opět snižuje pravděpodobnost odhalení. Zahrnuje rozhodování, zda se má daný soubor nakazit nebo nikoli, s ohledem na jeho velikost a míru diskové aktivity s procesem nákazy související.

- **Charakteristická – *payload***
  Tato (volitelná) část určuje charakteristické chování viru. Zpravidla jde o nějakou zákeřnou činnost (mazání souborů, omezování výkonu nebo funkčnosti OS, obtěžování uživatele, ...).



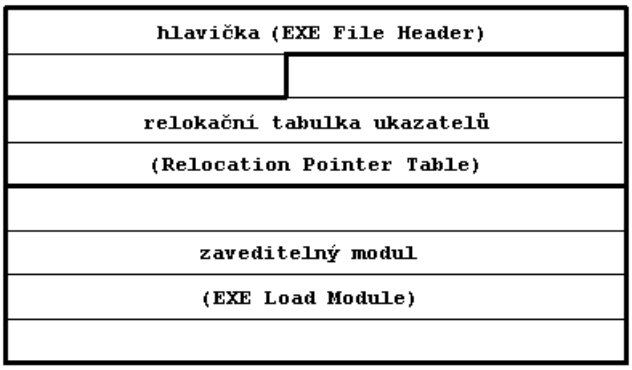
Obrázek 3.1: Funkční diagram viru [10]

3.5 Skrývání a identifikace

Faktem je, že i viry potřebují mít nějaký mechanismus, jak rozpoznat vlastní přítomnost v systému, aby se předešlo opakovanému napadení již jednou infikovaného souboru. Některé dále zmíněné principy tedy v sobě implementují i samotné viry, i když v jejich případě je úkol objevení přítomnosti jiné instance sebe sama o poznání snazší než v případě antivirů. Útočníci před vypuštěním nového viru do světa testují, zda bude jejich výtvor schopen projít přes kontrolu běžnými antivirovými produkty.

Dělení virů do kategorií se provádí na základě mnoha jejich vlastností. Ať už podle cílové destinace (souborové infektory, boot-sector infektory, makroviry, multipartitní viry) nebo způsobu maskování, případně způsobu použití kryptografických algoritmů (klasické, polymorfní, metamorfní, stealth viry, adresářové viry). Z pohledu této práce o kryptovirologii jsou nejzajímavější právě viry, které nějakým způsobem využívají kryptografii.

Typický virus (souborový infektor) napadá spustitelné soubory – nejčastěji typu EXE. Struktura EXE souboru je na obrázku 3.2. Jestliže chce virus napadnout EXE soubor, musí



Obrázek 3.2: Struktura EXE souboru [10]

modifikovat jeho hlavičku (File Header) a relokační tabulku (Relocation Pointer Table) a taktéž přidat vlastní kód do třetí části (Load Module). Virus se může umístit na libovolné místo v souboru. Řekněme, že se v jednom konkrétním případě umístí na konec EXE souboru. Musí zajistit, aby při spouštění souboru byl on jako první, kdo získá kontrolu nad řízením, nikoliv původní program.

3.5.1 Polymorfismus a metamorfismus

Snaha učinit viry co nejméně viditelné si vyžádala vývoj dalších technik, jak zakrýt jejich přítomnost v systému (falšování data a času u souborů, kontrolních součtů). Čím déle dokázal virus skrytě přežívat, tím větší byla jeho šance šířit se a replikovat. Nejenže se takovými technikami snaží skrýt před zraky uživatelů, hlavním jejich cílem je ztížení práce vyhledávacím programům.

Technik ukrývání existuje celá řada, zaměříme se zde podrobněji na jednu z nich – polymorfismus. U polymorfismu jde o to zakrýt vlastní kód tak, aby nebyl identifikován jako virus. Zjednodušený příklad infikovaného souboru je naznačen na obrázku 3.3. Náš virus se umístil na konec souboru, samozřejmě patřičně pozměnil vstupní bod původního programu (*entry point*) [14].

Obrázek 3.3: Běžný virus

Prvním zásadním vylepšením bylo zašifrování celého kódu viru v souboru a umístění před něj pouze malé dešifrovací smyčky – dekryptoru (*decryptor*). Vstupním bodem je pak tato dešifrovací smyčka. Hlavní tělo viru je nyní skryté, je odhaleno až v době spuštění dekryptoru. Polymorfismus je vlastnost mít mnoho různých podob. Proto mají polymorfní viry mnoho typů dešifrovacích smyček. V zašifrované části viru je navíc obsažena funkce, která obstarává polymorfní transformace. Nově nakažený soubor v sobě obsahuje kopii samotného viru, ovšem již v jiné zašifrované podobě. Schéma polymorfního viru je naznačeno na obrázku 3.4.

Obrázek 3.4: Polymorfní virus

Evoluce polymorfních virů vedla ještě dále – k metamorfním typům virů (obrázek 3.5). Jsou to zvláštní typy vycházející z principu polymorfismu, ale způsoby zatemňování kódu a maskování se před antiviry dotahují mnohem dále. Idea spočívá v rozmístění malých „ostrůvků“ virového kódu v různých místech programu. Vstupní bod pak může zůstat nezměněn, virus přebírá řízení až z kontextu spouštění původního programu.

Obrázek 3.5: Metamorfní virus

3.6 Detekce

Vyhledávání a likvidace virů a jiného malwaru antivirovými produkty je hlavní způsob boje proti šířící se nákaze. Základní metody vyhledávání jsou stejně staré jako první počítačové viry. V této podkapitole se budeme věnovat klasickým metodám detekce známých virů. O problematice vyhledávání nových a neznámých virů bude pojednáno v části o heuristických metodách. Přehled způsobů detekce malwaru [15]:

1. **Vyhledávání**
   Jedním z hlavních způsobů, který antivirové produkty běžně používají k detekci známých virů, je metoda porovnávání řetězců (*string matching*). Vyhledávání na tomto principu je v podstatě pouze vyhodnocování nějaké podmínky s výsledkem: *JE TO VIRUS / NENÍ TO VIRUS*. Testuje se shoda charakteristických řetězců, což jsou ve většině případů posloupnosti instrukcí.

   Metoda je relativně časově náročná, protože se musí testovat a vyhodnotit celá databáze virových vzorů pro každý zkoumaný soubor, na který se implicitně pohlíží jako na potenciálně infikovaný virem.

   Optimalizace algoritmů na porovnávání řetězců je velmi žádoucí. Vynaložené úsilí při vývoji se pak projevuje v rozdílné rychlosti antivirového softwaru u jednotlivých výrobců.

   Výhody:

   - jednoduchost
   - spolehlivost
   - úspěšnost léčení

Nevýhody:

- malá rychlost
- problémy s polymorfními viry
- neschopnost detekce nových virů

2. **Skenování**
   Antivirový vyhledávač znaků (*scanner*) je základem každého antivirového softwaru. K pátrání se používá jakýsi soubor charakteristických znaků – virový otisk, hledají se pouze znaky typické pro konkrétní virus.

   Výhody:

   - rychlost
   - úspěšnost léčení

   Nevýhody:

   - malá spolehlivost
   - problémy s polymorfními viry
   - neschopnost detekce nových virů

3.6.1 Heuristika

Heuristika neboli heuristická analýza je způsob detekce virů, které v době vydání antivirového softwaru ještě nebyly napsány. Jelikož je virus program jako kterýkoli jiný, je rozhodování, zda už se jedná o škodlivý kód nebo pořád jen obyčejný program, velmi složité. Stejný problém nastává také u polymorfních virů. Tento typ virů nejenže používá symetrické šifrování obsahu těla viru, ale dokáže měnit i dekryptovací smyčku. V podstatě tak může virus nabývat ohromného množství různých podob, čímž mají antiviry sníženou šanci jej odhalit [14].

U heuristické analýzy antivirus jakoby nahlíží do kódu potenciálního viru, postupuje po instrukcích tak, jako by program spustil a jednotlivé instrukce postupně vykonával. Narazí-li na podezřelé instrukce, varuje o možném nebezpečí uživatele. Ten musí rozhodnout, zda se o virus opravdu jedná, či nikoliv. Schopnost „odhalit“ neznámý virus je závislá na nastavení prahové hodnoty konkrétního antiviru.

Operační systémy poskytují mnoho prostředků (*service routines*), jak usnadnit ostatním programům přístup k periferním zařízením, a naplno tak využívat všech možných funkcí. Pomocí tohoto rozhraní je dosaženo větší úrovně abstrakce pro aplikace využívající hardware počítače. Programy a aplikace smí volat služby operačního systému pomocí mechanismu zvaného softwarové přerušení.

Možnost zasahovat do systému přerušení je způsob, jak zabránit virům v jejich spouštění. Stejnou možnost má ale i virus, ten může úspěšným ovládnutím tohoto mechanismu obcházet detekční antivirové programy. Další (proaktivní) metody a techniky detekce jsou:

1. **Sledování toku řízení**
   Jde o jistou dynamickou kontrolu. Sledují se vstupní body programu při spouštění. Vstupní bod (*entry point*) je místo, kde operační systém předává řízení programu, jenž začíná vykonávat činnost, pro kterou byl naprogramován.

Na základě jistých statistických vlastností kompilovaných spustitelných programů vznikl nástroj PEAT (*Portable Executable Analysis Toolkit*), který dokáže vyhodnocovat anomálie toku řízení v programu [22].

2. **Detekce kryptografického kódu**
   Používanou heuristickou metodou zjišťování virů je detekce kryptografického kódu v programu. Absence kryptografického kódu je jistou zárukou toho, že v systému není přítomen kryptovirus nebo jiný malware používající kryptografii. Viry mívají navíc implementovány nejrůznější algoritmy související s kryptografií – např. test, zda je číslo prvočíslem nebo algoritmus násobení velkých čísel – Karatsuba [28], popř. další speciální algoritmy. Detekování uspořádání nejmenších prvočísel (2, 3, 5, 7, 11, 13, 17, 19 . . .) v programu může znamenat odhalení skrytého viru.

3. **Integritní kontrola**
   Metoda založená na porovnání aktuální informace o programu s uloženou hodnotou, která vznikla v době instalace (nebo ji poskytuje dodavatel příslušného softwaru). K tomuto účelu se výborně hodí CRC (*Cyclic Redundancy Check*). Namísto samotných virů se integritní kontrolou dají zjistit pouze jejich projevy v systému.

   Problémy s integritní kontrolou nastávají v případě legitimní modifikace souboru v čase. Hlášení antiviru o nalezení viru je zřejmě falešný poplach, protože zásahem porušujícím integritu (kontrolní součet) souboru je např. i provedená aktualizace programu pomocí tzv. patchů.

Poznámka:

Zvýšit bezpečnost systému lze svým způsobem i při spouštění programů. Zde hovoříme o ověřování „podepisovacích“ certifikátů softwaru, které se provádí za asistence operačního systému.

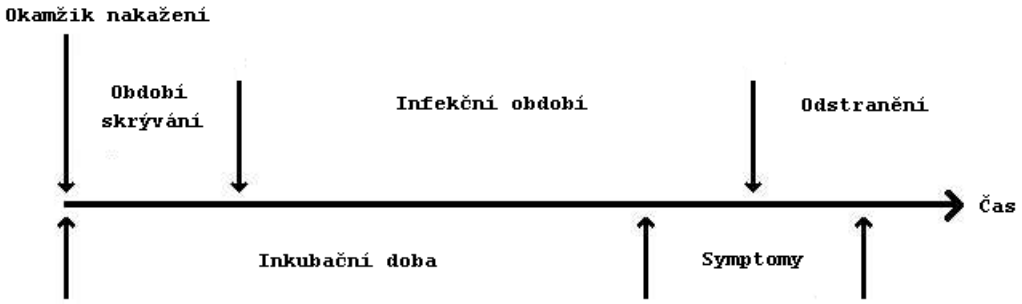
3.7 Mechanismy a modely šíření

Počítačové viry se svým chováním a způsobem šíření příliš neliší od klasických biologických patogenů. Proto se počítačový program napadající zranitelné systémy nazývá také „virus“ (latinsky *virus*, což znamená „jed“, „zhouba“).

Sledování průběhu nákazy biologickým virem v reálném prostředí je velice obtížné. Tímto problémem se zabývalo mnoho studií, dnes se provádí převážně jako počítačové simulace, např. pomocí celulárních automatů – CA (*Cellular Automata*), což jsou diskrétní modely, kterými se zabýváme v teorii systémů, matematice a teoretické biologii. Buňkou v celulárním automatu je konečný automat, FSA (*Finite State Automaton*).

Převratné výsledky v oblasti předpovědí šíření epidemií přinesl projekt nazvaný *Where's George?* [40]. Projekt byl zaměřen na sledování pohybu bankovek v USA. Získaná data odhalila zajímavé statistické zákony o pohybu osob ve Spojených státech amerických a pomohla k vývoji matematického popisu, který může být použit k modelování šíření infekčních nemocí v dané zemi. Autoři projektu (studie) *Where's George?* věří, že takto získané výsledky výrazně vylepší předpovědi o geografickém šíření epidemií [39].

Nakažený hostitel je takový hostitel, který vykazuje příznaky nákazy. Symptomy jsou většinou mechanismy, které viru pomáhají šířit se na další hostitele. Vztahy mezi jednotlivými fázemi infekce jsou na obrázku 3.6 [6].



Obrázek 3.6: Vztah mezi infekčností a virovými symptomy (popisky nad časovou osou popisují infekčnost, zatímco pod osou dynamiku nákazy)

3.7.1 Fáze virové nákazy

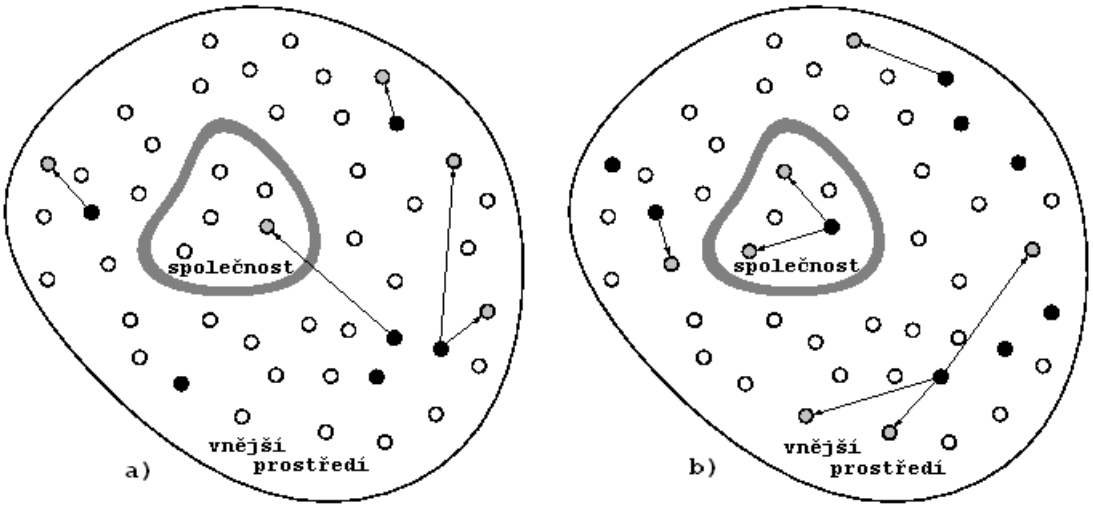
- **Období skrývání**
  V průběhu raného stádia infekce, kdy si virus buduje schopnost šíření na nového hostitele.
- **Infekční období**
  V této fázi je virus již nakažlivý a může se přirozenými mechanismy šíření přenášet na jiné hostitele.
- **Odstranění**
  V závislosti na hostiteli – získání imunity nebo jeho smrt – v této fázi není virus schopen dalšího šíření.
- **Inkubační doba**
  Tato fáze infekce se nemusí u hostitele projevovat žádnými příznaky. Během průniku této fáze a fáze infekčního období se virus šíří nejvíce. Je to dáno tím, že hostitel dosud nemá ponětí o tom, že je přenašečem viru a udržuje normální kontakt s ostatními.
- **Symptomatické období**
  Poslední fází infekce je symptomatické období, kdy jsou již patrné známky nákazy.

3.7.2 Šíření

Modely šíření viru jsou navrhovány z mnoha důvodů. Epidemiologové potřebují jednoduché modely k testování dopadu specifických parametrů na celkové chování epidemií. V počítačové vědě se studiem šíření biologických virů a aplikací poznatků do světa počítačů snažíme předpovídat rychlost šíření a dopady infekce způsobené malwarem. Přestože se místo skutečného světa pohybujeme v prostředí Internetu, dávají výsledky simulací mnohdy překvapivě přesné výsledky [5].

Počítačový virus může být napsán tak, aby se dokázal šířit na libovolnou platformu. Obrázek 3.7 bude sloužit jako modelová situace, kterou si detailněji představíme.

Z pohledu modelové organizace je okolní svět plný počítačových virů, jež se snaží proniknout skrze její polopropustné hranice, které ji oddělují od vnějšího prostředí. Frekvence pokusů o průnik závisí na počtu virových nákaz v prostředí (např. Internetu), dále na počtu počítačů v dané organizaci a na „propustnosti“ hranic organizace (odolnosti bezpečnostních



Obrázek 3.7: Šíření počítačového viru z perspektivy organizace; bílé „objekty“ reprezentují nenakažené počítače, černé nakažené počítače a šedé počítače, jež jsou právě v procesu nakažení virem. **a**): virus proniká za hranice organizace z okolního světa – začátek virového incidentu. **b**): nákaza se šíří na ostatní počítače v rámci organizace – rozsah nákazy je vyjádřen počtem počítačů, na kterých je nákaza objevena a odstraněna.

opatření). Dříve či později si virus najde cestu dovnitř. Okamžik průniku viru přes hranice organizace značí začátek virového incidentu [23].

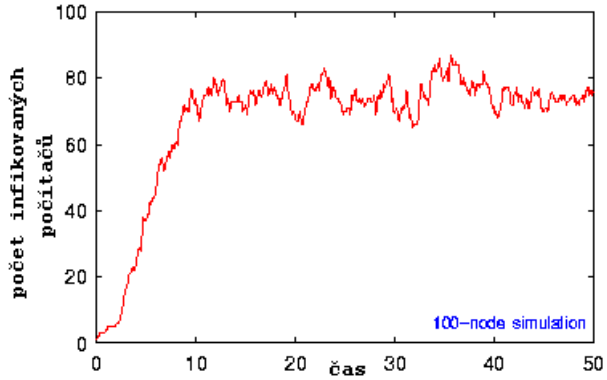
Počet nových virů stabilně roste, ale nezvyšuje se exponenciálně. Míra schopnosti šířit se je u každého viru různá. V teoretických modelech zaměřených na porozumění způsobům, jak se počítačové viry šíří (viz [23]), existují pouze dva stavy sledovaného systému. Systém buď je nakažen virem, nebo není. Pokud je nakažen, existuje jistá pravděpodobnost, že každý den „přijde do kontaktu“ s jinými systémy a dojde k přenosu viru. V současnosti se přenos děje spíše skrze počítačovou síť, dříve to bylo zpravidla prostřednictvím disket.

Tuto pravděpodobnost kontaktu nazýváme „porodnost“ (*birth rate*) viru. Podobně existuje nějaká pravděpodobnost objevení nákazy v systému a její odstranění. Pak hovoříme o tzv. „úmrtnosti“ (*death rate*) viru.

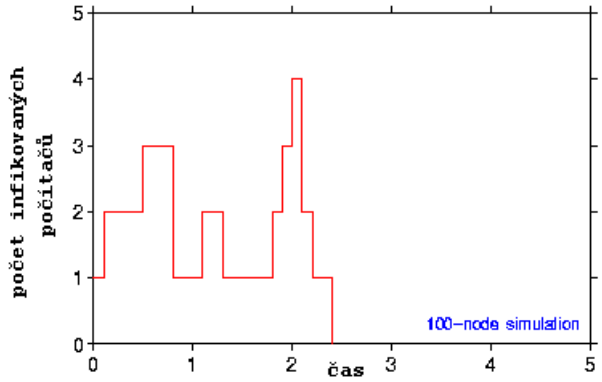
Dvě zmíněné pravděpodobnosti jsou ovlivňovány mnoha faktory. Jednak způsobem, který konkrétní virus používá k replikaci a šíření, dále tím, jak rychle je v systému objeven a eliminován (vlivy: zkušenost uživatele, činnost viru, používání antivirového software). Podle výzkumu, prezentovaného v [23], je zásadní zjištění existence jisté prahové hodnoty epidemie (*epidemic threshold*). Nad touto hodnotou se virus šířit může, pod nikoli.

Pokud je „porodnost“ viru větší než „úmrtnost“, má virus šanci se úspěšně šířit (i když může také zaniknout dříve, než se masově rozšíří). Další pozoruhodný závěr tohoto výzkumu je, že rychlost rozšíření viru může být mnohem menší, než je exponenciální, která bývá předpovídána v jedné publikované teorii [21]. Na obrázku 3.8, zmíněné studie, je znázorněno typické chování v prostředí nad prahovou hodnotou.

Jestliže je „porodnost“ menší než „úmrtnost“ (když je virus v systému objeven a eliminován rychleji, než se šíří), pak nemůže dojít k masovému rozšíření. Může se sice šířit na několik počítačů po jistou dobu, ale poté bude odhalen a eliminován z celé populace (stane se „mrtvým“). Toto chování ilustruje obrázek 3.9.



Obrázek 3.8: Nad prahovou hodnotou rozšíření viru narůstá rychlostí, která závisí na mnoha faktorech, pak se ustálí v určité rovnováze (v této simulaci „porodnost“ viru 5-krát převyšovala „úmrtnost“)



Obrázek 3.9: Pod prahovou hodnotou sice mohou propuknout malé incidenty, ovšem „vyhubení“ nákazy je nevyhnutelné (v této simulaci byla „porodnost“ viru o 10 % menší, než „úmrtnost“; navíc je zajímavé, že horizontální i vertikální stupnice se velmi liší od předchozího případu, viz obrázek 3.8)

3.8 Analýza nebezpečnosti malwaru

Problematika okolo malwaru je poměrně dynamická. Uvádí se, že se denně na Internetu objevuje několik desítek nových aktivních počítačových virů a červů. Při takovém množství je obtížné přesně odhadnout ztráty způsobené útoky. Do kalkulací by se jistě daly zahrnout i mimořádné výdaje firem do dalších bezpečnostních opatření (HW, SW, bezpečnostní politika), které se zavádějí bezprostředně po nějakém významném bezpečnostním incidentu. Intervaly mezi incidenty, jak ukazuje žebříček Top 10 nejvíce devastujících virů, nejsou nikterak dlouhé.

Je otázkou času, kdy přijde další drtivý útok ze strany malwaru. Zdá se, že výše škod napáchaných působením těchto zákeřných programů není shora nikterak omezena. Částky v řádech milionů amerických dolarů, realita před dvěma dekádami, byly mnohonásobně překonány útoky za posledních deset let.

Trh s bezpečnostními produkty je naplněn podobnými softwarovými i hardwarovými

řešeními zacílenými stejným směrem. Žádná ochrana není 100%, ale provozovat aktualizovaný OS s doplňkovými programy (antivirový software, anti-spyware, firewall, bezpečnější internetový prohlížeč) je v dnešní době absolutní nutností. Obecně všechen potenciálně zranitelný software by měl být udržován a pravidelně aktualizován.

Ke složitějším, avšak obdobně důležitým, úkonům patří správná konfigurace (ve smyslu bezpečnosti) všeho, co lze jakkoli připojit k počítačové síti. Pokud je toto vše splněno, zbývá poslední – nadmíru důležitá součástka k tomu, aby celý „stroj“ správně fungoval – tou je konečně správné a uvědomělé chování uživatelů.

Bezpečnosti a s ní souvisejícím oblastem budeme věnovat pozornost ještě v jedné z následujících kapitol.

3.8.1 Top 10 – Nejdestruktivnější viry všech dob

Tato část přináší přehled o desítce nejdestruktivnějšího počítačového malwaru všech dob (řazeno chronologicky) [9].

- **CIH a.k.a. Chernobyl (1998)**
  Odhadovaná způsobená škoda: 20–80 mil. $. Původem z Tchaj-wanu, napadal EXE soubory OS Windows 95/98/ME. Mazal data, přepisoval BIOS a zabraňoval nabootování počítače.

- **Melissa (1999)**
  Odhadovaná způsobená škoda: 300–600 mil. $. Nakaženo až 20 % všech firemních počítačů. Šířil se pomocí programu MS Outlook. Modifikoval dokumenty formátu MS Word.

- **ILOVEYOU a.k.a. Loveletter (2000)**
  Odhadovaná způsobená škoda: 10–15 mld. $. Původem z Filipín. Červ se šířil pomocí e-mailů, přepisoval hudební a grafické formáty souborů. Vyhledával navíc hesla a posílal je e-mailem autorovi viru.

- **Code Red (2001)**
  Odhadovaná způsobená škoda: 8,7 mld. $. Navržen za účelem způsobit maximální škodu. Napadal systémy IIS (*Internet Information Server*). Po dobu 20 dní zobrazoval zprávu: `"HELLO! Welcome to http://www.worm.com! Hacked By Chinese!"` na webových stránkách běžících pod IIS všech infikovaných systémů, poté prováděl útoky typu DoS.

- **SQL Slammer a.k.a. Sapphire (2003)**
  Odhadovaná způsobená škoda: 1,3 mld. $. Cílem nebyly jednotlivé stanice, ale servery. Měl negativní dopad na fungování celého Internetu.

- **Blaster a.k.a. Lovsan (2003)**
  Odhadovaná způsobená škoda: 2–10 mld. $. Napadal systémy OS Windows 2000/XP. Způsoboval vypínání systému napadeného počítače.

- **Sobig.F (2003)**
  Odhadovaná způsobená škoda: 5–10 mld. $. Napadal prostřednictvím e-mailu. Dokázal vygenerovat milion svých vlastních kopií za dobu 24 hodin. Za identifikaci autora nabídl Microsoft odměnu ve výši 250 000 $.

- **Bagle a.k.a. Beagle (2004)**
  Odhadovaná způsobená škoda: Desítky mld. $. Tento červ se šířil opět prostřednictvím e-mailu jako příloha. Existuje až 100 variant tohoto malwaru. V systému zřizoval „zadní vrátka“. Označován za průkopníka – první malware, který byl vytvořen pro finanční zisk.

- **MyDoom (2004)**
  Odhadovaná způsobená škoda: Desítky mld. $. V době nejsilnějšího působení způsoboval zpomalení načítání webových stránek o 50 % a výkon celého Internetu asi o 10 %.

- **Sasser (2004)**
  Odhadovaná způsobená škoda: Desítky mld. $. Nepoužíval k šíření e-mail, ale zneužíval bezpečnostní slabiny v neaktualizovaných OS Windows 2000/XP. Způsoboval pády a nestabilitu napadených systémů.

Kapitola 4

Kryptovirologie

4.1 Viry a kryptografie

Techniky ukrývání, resp. zatemňování vlastního kódu u virů, jsme si představili v kapitole 3.

Jako odpověď na viry ukrývající se pomocí symetrické kryptografie přišli výrobci antivirových produktů s různými podobami implementace tzv. „pískoviště“ (*sandbox environment*). Jde v podstatě o emulátor systému, specifický příklad virtualizace. Tímto se antiviry snaží přesvědčit virus (resp. pouze jistou část – dekryptor), aby se spustil (v domnění, že běží přímo v hostitelském systému), dešifroval a odkryl tak svůj vlastní kód.

Kompletní a věrohodná emulace prostředí antivirem je nezbytná z toho důvodu, že sofistikované viry detekční techniky antivirových společností znají a mívají implementovány obranné funkce. Takové funkce, které dokážou odhalit, že systém, na němž se má dekryptor viru spustit a dešifrovat – odhalit – tělo samotného viru, není skutečný hostitelský systém, ale že jde o snahu nepřátelského softwaru odhalit a zlikvidovat skrývající se virus.

Pokud se antiviru podaří přesvědčit virus, aby se odhalil a spustil, přichází na řadu klasické, dříve popsané detekční metody.

I když jde v tomto případě o využití kryptografie ke skrývání viru v operačním systému, nehovoříme zde stále o skutečném kryptoviru. Pojďme si tento nový pojem představit a definovat.

4.2 Kryptovirus

Definice 4.2.1 *Kryptovirus (cryptovirus)* je počítačový virus, který využívá kryptografii veřejným klíčem. [25]

4.2.1 Koncept

Koncept kryptoviru byl poprvé popsán a představen v roce 1996. Jde o typ počítačového viru, který se snaží získat výhodu (vedoucí k jeho přežití) tím, že určitým postupem donutí oběť přistoupit na daná pravidla hry. Výhody dosahuje především použitím veřejného kryptografického klíče, který si nese ve svém těle. Veřejným klíčem obvykle zašifruje symetrický klíč relace, jímž jsou šifrována důležitá data na disku.

Po úspěšném zašifrování cenných dat kryptovirus zobrazí na obrazovce informační zprávu o provedeném útoku a potřebné informace o způsobu, jakým má oběť viru postupovat, chce-li získat svoje data zpět. Typicky virus za zpřístupnění zašifrovaných souborů požaduje zaplacení výkupného [25, 2].

Jelikož odpovídající soukromý klíč zásadně nikdy není součástí viru, nelze ani po jeho důkladné analýze disasemblováním kódu zašifrovaná data získat zpět. Z definice asymetrické kryptografie plyne, že není možné ze znalosti veřejného klíče získat soukromý klíč. Pokud chce oběť získat svá data nazpět, musí splnit požadavky útočníka. Ten pak poskytne soukromý klíč k dešifrování použitého symetrického klíče relace. Takto vypadá jednoduchý scénář kryptovirálního útoku a jeho následky. Ve skutečnosti není celá procedura ani zdaleka tak jednoduchá a přímočará, jako v tomto uvedeném příkladu.

Nutnou podmínkou úspěšnosti útoku kryptovirem je totiž absence datových záloh. Pokud zálohy existují (zde je zálohou myšleno uložení daných cenných souborů mimo napadený systém – externí úložiště), není pro poškozeného větší problém napadený systém z počítače kompletně odstranit, provést jeho novou instalaci a zálohovaná data obnovit.

Autoři kryptovirů musí řešit mnohé překážky bránící kryptovirům v dotažení útoku do zdárného konce. Pokud by poškozený majitel dat souhlasil s podmínkami útočníka, zaplatil požadované výkupné, a pak získal jeho soukromý klíč potřebný k obnově souborů, mohl by jej posléze poskytnout i dalším obětem téhož viru (tzn. se stejným veřejným klíčem). Útok by tak ztratil na efektivitě.

Z pohledu autora viru je možným řešením použití vlastního generátoru náhodných čísel nebo využití funkce systému oběti a vygenerování náhodného symetrického klíče relace a inicializačního vektoru (problematika diskutována v kapitole 2) pro použitý algoritmus. Tato dvojice (*IV, klíč relace*) se zašifruje veřejným klíčem. Data na disku oběti se budou šifrovat symetrickým algoritmem, pro který se jako symetrický klíč a inicializační vektor použije právě vygenerovaná dvojice.

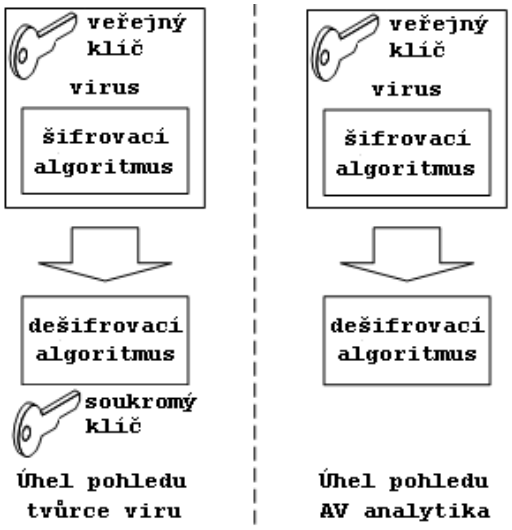
Symetrický klíč k šifrování dat je použit také z toho důvodu, že symetrické algoritmy jsou v šifrování mnohem rychlejší než asymetrické – překážka v podobě nepoužitelnosti šifrování dat asymetrickým algoritmem – veřejným klíčem je tedy odstraněna.

Oběť je kryptovirem informována o nastalé situaci stejně, jako ve scénáři u předchozího zjednodušeného příkladu. Nově je zde ale požadováno zaslání pouze prvotně vygenerované a zašifrované dvojice (*IV, klíč relace*) útočníkovi. Útočník provede dešifrování svým soukromým klíčem a posílá nazpět, opět po splnění požadavků, klíč relace a IV, nyní už v čitelné podobě – nešifrovaně.

Vzájemná spolupráce více obětí pomocí sdílení dešifrovacího klíče je znemožněna, neboť tím je pouze dešifrovací symetrický klíč, který se generuje až při napadení počítače kryptovirem a je tedy v případě každé oběti zcela unikátní.

Podíváme-li se na klasický virus využívající kryptografii ke stejnému účelu, nikoli však ještě kryptovirus z definice 4.2.1, je jasné, že po podrobné analýze bude obsah těla viru z pohledu antivirového analytika stejný, jako z pohledu jeho tvůrce. Virus v sobě obsahuje určitý šifrovací klíč a algoritmus, kterým se pomocí klíče zašifrují určitá data. V tomto případě vždy existuje zpětný mechanismus – dešifrovací algoritmus, kterým lze šifrovaná data převést zpět do původní podoby. Stále hovoříme o symetrické kryptografii, kde stačí pro obě zmíněné operace jediný klíč. To by u viru, s cílem vydírat uživatele a požadovat výkupné prostřednictvím zašifrovaných dat, představovalo poměrně slabé místo. Zjištěním způsobu, jak byla data virem zašifrována, by je bylo možné také obnovit, samozřejmě k nelibosti útočníka.

Naopak použitím asymetrické kryptografie přináší kryptovirus útočníkovi nad antivirovým analytikem velkou výhodu. Virus samozřejmě opět obsahuje šifrovací algoritmus a klíč, zde jde však jde o klíč veřejný. Odpovídající soukromý klíč v těle viru přítomen není (viz obrázek 4.1). Jediná kopie je bezpečně uložena pouze na straně útočníka, může být např. na jeho čipové kartě. Klíč na čipové kartě s sebou přináší pro útočníka další



Obrázek 4.1: Kryptovirus z pohledu tvůrce a antivirového analytika

příjemnou výhodu, a to pro případ soudního procesu po jeho odhalení. Muselo by mu být také dokázáno, že použitý veřejný klíč nebyl falešný atd. Detailně je otázka viny a neviny útočníka v souvislosti s použitím falešného veřejného klíče rozebrána v literatuře, viz [25].

Z definice kryptosystému veřejným klíčem je zřejmé, že neexistuje způsob, jak ze znalosti běžně dostupného veřejného klíče odvodit odpovídající klíč soukromý, kterým se šifrovalo. Antivirový analytik nemůže tedy ani detailním rozborem viru z jeho veřejného klíče získat soukromý klíč útočníka.

Síla kryptoviru spočívá v principech asymetrické kryptografie. Jediným a teoreticky možno proveditelným řešením nepřítomnosti soukromého klíče na straně oběti viru by snad bylo využití slabého místa samotného šifrovacího algoritmu použitého při útoku a prolomení tohoto algoritmu.

4.3 Shrnutí

Popsaný kryptovirus byl poprvé představen relativně nedávno. V porovnání s běžnými počítačovými viry a jiným škodlivým softwarem by se dalo říci, že u kryptovirů zatím nedošlo k takovému rozšíření, aby se tento relativně nový druh stal hrozbou číslo jedna.

Teoreticky nachází kryptovirus nejlepší uplatnění v systému především v tom případě, kdy objeví nezabezpečená, přesto vysoce „cenná“ data. Nedostatečné zabezpečení takových systémů ale může být překonáno dnes zcela běžnými způsoby, proto z pohledu zajištění bezpečnosti nerozlišujeme, zda se jedná o protiopatření zabraňující klasické nebo kryptovirální nákaze.

Kryptografie byla vždy určena spíše k ochraně dat a k zajištění celkové bezpečnosti. Nyní se však ukazuje, že přítomnost kryptografických algoritmů v moderních OS paradoxně může znamenat i velmi vážnou hrozbu. V případě, že neexistují zálohy dat kryptovirem napadeného systému, nevede k jejich obnově zpravidla jiná cesta, než je splnění požadavků držitele soukromého klíče – útočníka. Možnost vymáhání výkupného, pod výhrůžkou zveřejnění získaných utajovaných firemních dat, by mohlo pro společnosti pohybující se ve vysoce konkurenčním prostředí představovat vážný problém.

Kapitola 5

Implementace

Implementační část práce má za cíl posloužit jako důkaz proveditelnosti a použitelnosti mechanismů popisovaných v předchozích kapitolách. Jako nejvhodnější se k tomuto účelu ukázala být platforma MS Windows. Celý vývoj (kódování programu) probíhal konkrétně pod OS Windows XP. Právě Windows XP je v současnosti nejrozšířenějším operačním systémem, proto je velice pravděpodobné, že kryptovirální útok založený na podobném principu, jaký je zde prezentován, by v reálu znamenal vážný bezpečnostní problém.

Na zmíněném OS byl výsledný program také patřičně otestován, nicméně použité funkce by měly zaručit jeho funkčnost (příp. s drobnými úpravami) také na OS Windows 2 000 a nově i na Windows Vista.

Implementačním jazykem je, také z důvodů použití knihovny Windows API, jazyk C. Dobrá přenositelnost v rámci platformy Microsoft Windows je zaručena použitím standardu ISO C99. Jazyk C je vhodným kandidátem také z toho důvodu, že (narozdíl od jazyka C++) má programátor nad svým programem úplnou kontrolu a výsledný kód je relativně malý.

Poznámka:

Implementace využívá několika funkcí a postupů z programu, kterým autoři kryptovirologie Adam Young a Moti Yung již dříve prezentovali možnosti zneužití nástrojů knihovny CryptoAPI. Zdrojové kódy, z nichž i tato implementace ideově vychází, jsou volně dostupné ke stažení na stránkách autorů, viz `http://www.cryptovirology.com/`.

5.1 Implementační prostředky

Softwarové prostředky

- operační systém – Windows XP SP2, verze 2002
- virtuální stroj – VMware Player 2.0.3
- nástroje – MinGW 5.1.3 (*Minimalist GNU for Windows*) s GCC 3.4.5 (mingw special)
- textový editor – PSPad 4.5.3
- knihovny – Windows API a Cryptography API

Hardwarové prostředky

- procesor – AMD Athlon XP mobile 3 000+, 2 200 MHz
- operační paměť – 1 280 MB DDR RAM
- disk – 60 GB, 7 200 RPM

Překlad lze provést pomocí přiloženého `Makefile`, kde jako parametry překladače byly použity volby: `-std=c99 -Wall -W -pedantic -O3`.

5.2 CryptoAPI

Kryptografické nástroje nebyly dříve běžnou součástí distribucí operačních systémů. Realizace útoku s použitím kryptografie veřejným klíčem si v minulosti tedy vyžadovala zvláštní implementaci podpůrných funkcí. Celá záležitost použití kryptografie v aplikacích se velice usnadnila s příchodem Microsoft Windows 95 OEM Service Release 2, s nímž bylo společně distribuováno také aplikační programové rozhraní Cryptography API (CryptoAPI) [24].

CryptoAPI bylo vyvinuto jako prostředek k usnadnění práce programátorům při vývoji aplikací založených na operačních systémech Microsoft Windows a Windows Server.

Program vyvinutý na základě studia kryptovirologie, jakožto součásti této diplomové práce, je celý založený na možnostech plynoucích z přítomnosti CryptoAPI v systémech Windows. Demonstruje, jaké hrozby může existence kryptografických nástrojů v celosvětově rozšířených OS představovat pro jejich uživatele. Jeho relativní jednoduchost ukazuje, jak by mohl vypadat malware budoucnosti – využívající kryptografických funkcí poskytovaných operačními systémy.

5.2.1 CSP

Cryptographic Service Provider neboli CSP je implementace kryptografických algoritmů pro specifické délky klíčů. CSP sestává z jedné nebo více dynamicky linkovaných knihoven, které implementují kryptografické systémové programové rozhraní (CryptoSPI).

CSP může být implementován v softwaru, jindy zase mohou být funkce implementovány hardwarově, např. na čipové kartě. Pokud CSP neimplementuje všechny své funkce, pak slouží jako mezivrstva, která umožňuje operačnímu systému komunikovat s platnou implementací CSP.

Platforma Windows má vlastní množinu zabudovaných CSP poskytovatelů – *Microsoft Cryptographic Service Providers*. Přehled všech kryptografických poskytovatelů uvádí následující výčet [29]:

- *Microsoft Base Cryptographic Provider*
- *Microsoft Strong Cryptographic Provider*
- *Microsoft Enhanced Cryptographic Provider*
- *Microsoft AES Cryptographic Provider*
- *Microsoft DSS Cryptographic Provider*
- *Microsoft Base DSS and Diffie-Hellman Cryptographic Provider*

- *Microsoft Enhanced DSS and Diffie-Hellman Cryptographic Provider*
- *Microsoft DSS and Diffie-Hellman / Schannel Cryptographic Provider*
- *Microsoft RSA / Schannel Cryptographic Provider*
- *Microsoft RSA Signature Cryptographic Provider*

Microsoft Base Cryptographic Provider poskytuje implementaci 512bitového algoritmu RSA a 56bitového DESu. *MS Enhanced Cryptographic Provider* implementuje 1024bitový RSA, 56bitový DES a triple-DES (3DES). *MS Base Cryptographic Provider* bohužel nepodporuje klíče větší než 512 bitů a není podporován ani algoritmus 3DES.

V demonstračním programu (pojmenovaném příhodně `cryptovirus.exe`) je použit právě zmíněný Microsoft Enhanced Cryptographic Provider. Skutečné jméno tohoto poskytovatele je MS_ENHANCED_PROV. Typ použitého poskytovatele je PROV_RSA_FULL. CSP ukládá páry klíčů v nonvolatilní paměti. Softwarově implementované CSP mohou tyto klíče ukládat v šifrované podobě v registrech (*Windows Registry*). U CSP v hardwaru se k tomuto účelu používají tzv. *tamper-resistant* zařízení.

5.2.2 Objekt *key container*

Kryptografický pár klíčů je uložen v logickém datovém objektu zvaném kontejner (*key container*). Pro každého klienta (resp. uživatele) CSP udržuje jeden takový kontejner. Každý klíčový kontejner může uchovávat jeden pár klíčů daného typu, který je ze strany CSP podporován. Microsoft Base CSP například podporuje dva typy klíčových párů: klíče k digitálnímu podepisování a pár sloužící k výměně klíčů.

Kontejnerů může být otevřeno v jednom okamžiku, ze strany programu, i více. Při každém volání CryptoSPI musí volaná funkce specifikovat, který klíčový kontejner má být použit. Specifikace kontejneru se provádí prostřednictvím jednoho z parametrů volané funkce.

Klíč generovaný pro symetrickou blokovou šifru má pro použitý režim CBC (viz kapitola 2) implicitně nastaven nulový inicializační vektor [24]. Nulový IV může, a měl by, být změněn pomocí CryptoAPI funkce `CryptSetKeyParam`.

5.3 Použité CryptoAPI funkce

Při implementaci byly z CryptoAPI použity následující funkce:

- `CryptAcquireContext`
- `CryptReleaseContext`
- `CryptGenRandom`
- `CryptGenKey`
- `CryptDeriveKey`
- `CryptSetKeyParam`
- `CryptEncrypt`

- `CryptDecrypt`
- `CryptExportKey`
- `CryptImportKey`
- `CryptDestroyKey`
- `CryptCreateHash`
- `CryptHashData`
- `CryptDestroyHash`

5.4 Návrh, design a funkčnost programu

V této podkapitole bude podrobněji popsána architektura programu s popisem jeho nejzajímavějších funkcí a činností, tak jak je vykonává od okamžiku spuštění až do chvíle ukončení běhu.

5.4.1 Fáze 1 – spuštění

Po spuštění na hostitelském systému program nejprve nečinně vyčkává po dobu 60 s (aby ihned neprozradil náhlou aktivitou svoji přítomnost).

Když uplyne vyčkávací období, zjistí, zda už v systému neběží jiná instance téhož kódu. Pokud ne, vytvoří pojmenovaný mutexový objekt. Existence takového mutexu se stejným jménem způsobí, že se daná instance kryptoviru ukončí. Tak se zajistí, že data nebudou modifikována dvěma procesy zároveň.

Po testu přítomnosti jiné instance v systému se program pokusí získat debuggovací práva pro běžící proces. Ta jsou výhodná pro další činnost. V případě, že byl kryptovirus spuštěn pouze pod běžnými uživatelskými právy, neměl by možnost zasahovat do systémových registrů, mazat některé typy souborů z disku nebo ovlivňovat běh ostatních programů v systému. Běžnou praxí je detekce procesů běžících antivirových programů a jejich násilné ukončování, popř. také mazání z disku. Dubuggovací práva umožňují kryptoviru provádět některé ze zmíněných praktik.

Předpokládejme, že byl proces spuštěn pod administrátorským uživatelským účtem – tato možnost je velice častá a obvyklá u mnoha uživatelů systému Windows XP.

Dalším krokem je editace hodnot v registru Windows – program se pokusí zablokovat možnost spuštění správce procesů, a to vytvořením a spuštěním kódu v jazyce VBScript. Navíc přidá do registru novou hodnotu, která zajistí jeho spuštění po dalším startu systému. Program se tak stane rezidentním.

5.4.2 Fáze 2 – *payload*

Po spuštění a zajištění opětovného spuštění po restartu systému program zahajuje vlastní zákeřnou činnost (*payload*). Voláním funkce `CryptAcquireContext` získá ovladač (*handle*) ke klíčovému kontejneru vybraného CSP.

Generuje se symetrický klíč relace, kterým se budou data šifrovat. Použit algoritmus 3DES s délkou klíče 168 bitů. Jako režim byl zvolen mód CBC. Funkcí `CryptGenRandom` generuje 8 bajtů, které poslouží jako inicializační vektor. Symetrickému klíči se tento parametr

nastaví zavoláním funkce `CryptSetKeyParam`. `CryptExportKey` zajistí vyexportování symetrického klíče z CSP, klíč je při exportu zašifrován veřejným klíčem. Je převeden z binární podoby do hexadecimálního tvaru ASCII kódování a uložen do souboru. Za zmínku jistě stojí fakt, že dvojice asymetrických klíčů používá algoritmus RSA, délka klíče je 1024 bitů.

Dalším krokem programu je zjištění aktivních jednotek pevných disků v počítači. Aktivní disky se postupně prohledají. Prochází všechny soubory na disku, jeden po druhém, a porovnává jejich typ s množinou specifikovaných typů souborů. Pokud narazí na soubor, který má být zašifrován, zavolá funkci `FileEncrypt`.

`FileEncrypt` vytvoří nový soubor stejného jména, ale navíc připojí k souboru koncovku „*.encrypted*“, pomocí této koncovky se dají pouhým pohledem na obsah adresáře odlišit soubory vzniklé činností kryptoviru. Ihned po zašifrování je cesta a jméno souboru zapsáno do souboru, který slouží jako seznam souborů, které jsou drženy jako předmět výkupného. Na původní soubor je volána funkce `WipePlaintextFile`.

Jelikož není použitý IV žádným tajemstvím, je těchto 8 bajtů vloženo na začátek každého zašifrovaného souboru. Dvojice textových souborů vzniklých v této fázi činnosti kryptoviru, tj. seznam zašifrovaných souborů a veřejným klíčem šifrovaný symetrický klíč relace, je nutná k obnově dat po splnění podmínek útočníka.

Po ukončení této fáze dojde ke zničení klíčového kontejneru CSP, včetně uložených klíčů, k tomuto účelu CryptoAPI poskytuje funkce `CryptDestroyKey` a `CryptReleaseContext`. Program informuje uživatele systému o provedení útoku. Poté přejde do následující fáze – čeká, až uživatel získá dešifrovaný symetrický klíč – zadržovaná data pak mohou být opět zpřístupněna.

5.4.3 Fáze 3 – obnova dat

Nyní se virus nachází ve fázi čekání, až uživatel splní stanovené podmínky a získá dešifrovaný symetrický klíč. Poté, co je klíč k dispozici, kryptovirus jej načte ze souboru a provede import do CSP pomocí funkce `CryptImportKey`.

Když byl klíč úspěšně importován, může dojít k obnově kryptovirem zadržovaných souborů. K tomuto účelu slouží seznam vytvořený v předchozí fázi. Soubory jsou postupně dešifrovány do původní podoby. Pro případ chybného dešifrování souboru zůstávají soubory s příponou „*.encrypted*“ nadále v systému. V případě jakýchkoli komplikací lze tedy postup dešifrování opakovat.

Z bezpečnostních důvodů je soukromý klíč (komplement veřejného klíče v těle kryptoviru), sloužící k dešifrování symetrického klíče relace, chráněn pomocí hesla. Heslo zná pouze útočník a slouží zde jen jako pojistka proti použití soukromého klíče jinou osobou, než je autor kryptoviru (útočník). Hashované heslo (funkcí `CryptHashData`) bylo použito k odvození symetrického klíče sloužícího pouze k zašifrování tajného klíče. Odvození hesla, resp. derivaci z hodnoty parametru, provádí funkce `CryptDeriveKey`.

5.4.4 Rekapitulace průběhu útoku

1. Vygenerování RSA dvojice klíčů.
2. Zakódování veřejného klíče do těla kryptoviru.
3. Uložení soukromého klíče v šifrované podobě na disk (popř. čipovou kartu) útočníka.
4. Spuštění kryptoviru na cílovém systému.

5. Zašifrování dat uživatele náhodně generovaným symetrickým klíčem relace.
6. Poskytnutí šifrovaného symetrického klíče relace uživateli.
7. Požadování výkupného výměnou za dešifrování použitého klíče.
8. Vyjednávání a splnění podmínek útočníka.
9. Dešifrování klíče relace pomocí soukromého klíče na straně útočníka.
10. Obnovení (dešifrování) dat pomocí získaného symetrického klíče relace.

5.5 Efektivnost

Protože je cílem každého malwaru splnit své poslání dříve, než bude v systému objeven a zlikvidován, je pro autora takového škodlivého kódu takřka pravidlem, aby byl jeho program pokud možno co nejmenší, dostatečně rychlý a svou činností v napadeném systému příliš neupoutával pozornost, alespoň do té doby, než dojde k jeho úspěšnému šíření na další hostitelské systémy.

Kryptovirus se od jiných virů liší pouze svou charakteristickou fází činnosti malwaru (funkcí), která se nazývá *payload*. Jiné fáze, resp. funkce malwaru, jsou naprosto totožné pro všechny druhy těchto programů.

Efektivitou zde chápeme hlavně rychlost, jakou dokáže kryptovirus splnit své specifické poslání a množství spotřebovaných systémových zdrojů za dobu působení. Měření efektivnosti implementovaných funkcí probíhalo na referenčním stroji, na kterém probíhala i implementace, některé parametry byly z důvodů bezpečnosti měřeny na virtuálním stroji. Pomocí volání funkce `clock`, resp. konstrukcí `clock()/CLOCKS_PER_SEC`, byla zjišťována doba běhu programu. Rozdíl mezi dvěma časy, před a po zavolání funkce `FileEncrypt`, nám dává dobu trvání (v sekundách).

Testováním efektivnosti implementace bylo zjištěno, že doba potřebná k prohledání celého obsahu pevného disku (cca 300 000 souborů) trvá přibližně 90 sekund, s přihlédnutím k momentálnímu vytížení procesoru a množství spuštěných programů. Naměřená doba 90 sekund by mohla představovat okamžik vyzrazení přítomnosti kryptoviru v systému. Jelikož probíhá šifrování vybraných souborů ihned při nalezení v systému, celková doba vytíženosti CPU bude jistě mnohem delší – závisí na výkonu počítače, počtu souborů na disku a hlavně na tom, kolik souborů splní vyhledávací kriteria programu.

Rychlost samotného šifrování dat bylo testováno na malých (100 KB), středně velkých (2 MB) i velkých souborech (30 MB). Průměrná rychlost pak dosahovala hodnot okolo 8,5 MB/s, což je výsledek poměrně příznivý, protože velikosti cílových souborů (převážně dokumenty) zpravidla nedosahují zvlášť extrémních hodnot a ani se nepředpokládá větší celkový objem nalezených „cenných“ dat. Efektivnost kryptoviru je závislá také na volbě typů souborů, které se budou šifrovat.

K prohledání disku a zašifrování několika desítek vybraných souborů (velikosti jednotek MB) stačilo řádově několik desítek sekund. To je dostatečně krátká doba na to, aby si napadený uživatel stihl uvědomit, že se v systému děje něco nekalého, a stačil zamezit kryptoviru dokončit jeho poslání. Společně s blokováním spouštění některých systémových nástrojů (editací příslušných hodnot v registrech), jako je např. správce procesů, a tak je tímto programem prakticky nemožné včas hrozbu lokalizovat a zabránit tak napáchání škody.

5.6 Mazání souborů

Problematika bezpečného mazání souborů je v daném kontextu velmi důležitá. Aby mohl být útok kryptovirem smysluplně proveden, je kromě úspěšného zašifrování dat na disku také potřeba jejich nenávratné odstranění – smazání. Pokud by k smazání vůbec nedošlo nebo by nebylo provedeno zcela nenávratně, k žádnému útoku typu DoS by v podstatě nedošlo. Pokud by soubory po skončení této fáze kryptovirálního útoku zůstaly v nějakém stavu i nadále na disku, mohly by být tyto nesprávně smazané soubory pomocí dostupných programů opět obnoveny.

Z pohledu útočníka je takový postup neakceptovatelný, proto je nenávratné mazání souborů jednou z klíčových podmínek úspěšnosti kryptovirálního útoku. Klasické smazání souboru neodstraní skutečně jeho obsah z disku, pouze odstraní záznam o souboru z tabulky, kde se udržuje seznam souborů a jejich struktura v rámci souborového systému.

Protože binární data (posloupnost hodnot 1 a 0) jsou fyzicky stále na disku, pouze místo, kde jsou uložena, je v systému označené jako volné – je na něj tedy možné zapisovat jiná data, existuje stále vysoká pravděpodobnost, že půjdou „smazaná“ data obnovit.

Aby se zabránilo takovým praktikám pomocí specializovaných programů, provádí se bezpečné, tzn. nevratné mazání dat. Jediný způsob, jak data trvale odstranit, je jejich několikanásobný přepis zcela náhodnými hodnotami. Opravdu bezpečné smazání dat je netriviální záležitost, viz [8]. Mazání klasifikovaných dat z pevných disků se u některých institucí americké státní správy bere opravdu velmi vážně. Jako standard mazání disků používají až sedminásobný přepis celého obsahu disku náhodnými bity. Po této proceduře navíc následuje ještě fyzická likvidace disku roztavením.

Útočník zřejmě nepotřebuje při kryptovirálním útoku zajít do podobných extrémů. Pro takové potřeby naprosto postačuje jednoduchý, popř. dvojnásobný, přepis původního souboru náhodnými daty.

V implementovaném demonstračním programu `cryptovirus.exe` k účelu permanentního smazání původních souborů oběti slouží funkce `WipePlaintextFile`. Ta byla implementována s ohledem na možnosti kryptoviru. Funkce je volána vždy po každém zašifrování původního souboru.

Program `cryptovirus.exe` zjistí přesnou velikost souboru, ten pak v jeho celé délce přepíše náhodnými bity, které byly vygenerovány pomocí funkce `CryptGenRandom` po spuštění instance tohoto programu.

`CryptGenRandom` generuje náhodná data pomocí kryptograficky bezpečného generátoru náhodných čísel. CryptoAPI ukládá *seed* pro generátor náhodných čísel pro každého uživatele a jeho zdrojem jsou jak události z periferních zařízení (klávesnice, myš), tak i jiná systémová data (ID procesů, ID vláken, systémové hodiny, stav paměti, počet volných diskových clusterů, ...). Tato data se hashují pomocí algoritmu SHA-1 a výstup je použit jako *seed* RC4 datového proudu. Až teprve výsledek celé této procedury se použije k aktualizaci uloženého *seed*u. Data získaná funkcí `CryptGenRandom` lze považovat k danému účelu za dostatečně náhodná.

Takovéto smazání souborů je pro naprostou většinu uživatelů postačující. Obnovení dat dříve přepsaných náhodnými bity je téměř nemožné. Dokonce i pro zkušeného uživatele vybaveného patřičným softwarem. Tím je podmínka pro úspěšně provedený útok kryptovirem splněna.

5.7 Zhodnocení výsledků

Využitím několika funkcí knihoven CryptoAPI a WinAPI vznikl program, který dokáže vyhledávat na disku potenciálně cenné soubory (dokumenty, fotografie, e-cash) a šifruje je veřejným klíčem.

Doplněním o replikační část a implementací protokolu komunikace oběti s útočníkem by se z programu mohla stát potenciální internetová hrozba. Jelikož ale u virů nejde pouze o *payload*, musela by být přidána technika průniku do systému (využitím neopravené bezpečnostní díry SW nebo e-mail spolu s oklamáním uživatele) a nová – originální technika zamezující detekci současnými antivirovými programy, aby se kryptovirus mohl stát skutečnou hrozbou.

Zatím nebyl zaznamenán žádný větší bezpečnostní incident založený na kryptografii veřejným klíčem. Tato práce a program `cryptovirus.exe` nemá za cíl sloužit jako návod, jak podobné útoky provádět – naopak, hlavním posláním je prozkoumat potenciál propojování virů a kryptografie a přimět čtenáře k zamyšlení nad prezentovanými závěry. Pouze porozuměním celé problematice a zajištěním patřičných protiopatření se lze později kryptovirálním útokům ubránit.

Kapitola 6

Bezpečnost

Obor bezpečnost má ve světě informačních technologií relativně široký záběr a problematika spadající do této kategorie je proto značně obsáhlá. Následující kapitola je zaměřena konkrétněji – budou zde probrány oblasti, které se přímo dotýkají tématu této diplomové práce.

6.1 Anonymita v síti

V počátcích masovějšího používání Internetu nebyla možnost anonymity a anonymního používání dostupných služeb brána příliš v potaz. Každá entita připojená do sítě je v rámci této sítě identifikována přidělenou unikátní IP adresou, tím pádem je také snadno lokalizovatelná. To je výhodné z pohledu směrování provozu komunikace, avšak již nikoli z pohledu bezpečnosti (anonymity uživatelů). V dnešní době je možnost anonymního – bezpečného a nevysledovatelného pohybu v síti Internetu v jistých případech velmi žádoucí. Anonymní komunikace v elektronickém světě, resp. získání a udržení takové anonymity, může představovat problém.

Nutno zdůraznit, že potřeba anonymity při pohybu na síti se netýká pouze legitimních činností (např. elektronické volby, anonymní prohlížení webových stránek), ale také těch ilegálních. Útočníci potřebují k „bezpečné“ činnosti stejně silné prostředky.

Souvislost anonymity v síti s kryptovirologií můžeme najít v případě, kdy autor kryptoviru (viz kapitola 4) potřebuje anonymně komunikovat s obětí. Důvodem takové komunikace útočníka s obětí je získání výkupného za obnovení kryptovirem zašifrovaných dat.

6.1.1 Mix sítě

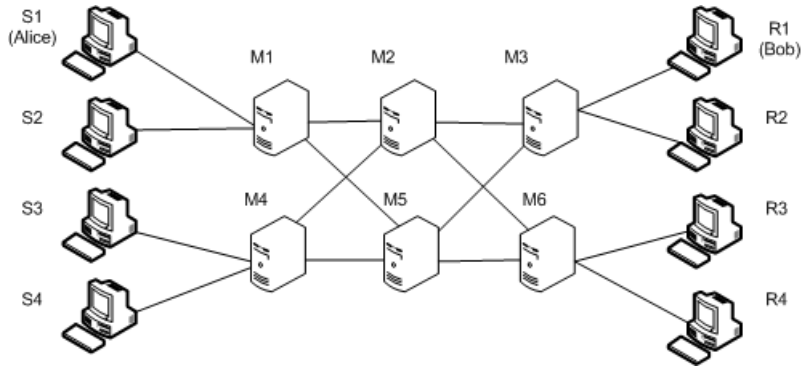
K zajištění anonymity při komunikaci v nezabezpečeném prostředí existuje teoreticky velmi silný prostředek. Jedná se o zvláštní druh logických sítí (fungují nad fyzickou síťovou infrastrukturou), ve kterých je adresace a směrování provozu zcela odlišné, než jak je tomu v klasickém případě.

Vesměs všechny takto specializované prostředky pracují na podobném principu, proto zde budou popsány pouze vlastnosti mix sítí a jejich možné využití.

Mix sítě (*mix networks*) fungují na následujícím principu (viz obrázek 6.1). Uzel sítě (tzv. mix) přijímá zprávy a odesílá je dále na místo určení v takovém pořadí, že vnější pozorovatel (narušitel) nedokáže odvodit žádnou souvislost mezi přijatou a odeslanou zprávou. Zpráva není přeposlána ihned po přijetí uzlem sítě, ale definované množství příchozích zpráv

je vysláno jako jedna dávka v jednom časovém okamžiku. Díky sdružování příchozích zpráv před samotným odesláním je možné provádět tzv. časové přeuspořádání.

Vysílání přeuspořádaných zpráv po dávkách je ochranou před útoky založenými na principu sledování časových souvislostí (*timing based attack*) [26]. Při použití mix sítí by pro případného narušitele mělo být velmi obtížné vysledovat, kdo komunikuje s kým.



Obrázek 6.1: Topologie mix sítě

Pojem anonymita je vhodné si ještě detailněji specifikovat. Anonymitu vztahu (*relationship anonymity*) chápeme jako nevysledovatelnost skutečnosti, že jedna konkrétní strana komunikuje s jinou. Anonymita odesilatele (*sender anonymity*) pak zakrývá fakt, že konkrétní odesilatel zaslal danou zprávu. A konečně, anonymita příjemce (*recipient anonymity*) znamená, že nelze zjistit, kdo je příjemcem dané zprávy.

Z uvedeného jasně vyplývá, že anonymita vztahu a anonymita odesilatele, resp. příjemce, nejsou zcela totožné pojmy. Pouze zajištěním obou typů – anonymity odesilatele i anonymity příjemce – lze dosáhnout fungující anonymity vztahu vzájemně komunikujících stran [17].

Nevysledovatelnost takového způsobu komunikace je zajištěna pomocí řetězení mnoha uzlů sítě (mixů) a hlavně použitím kryptografie veřejným klíčem. Každá zpráva je zašifrována na veřejným klíčem uzlu, na kterém se cestou k příjemci právě nachází. Na každém uzlu je daná „šifrovací obálka“ odstraněna použitím odpovídajícího soukromého klíče tohoto uzlu.

Dešifrováním zprávy tajným klíčem a inspekcí jejího obsahu mix zjistí, kam má být zpráva dále směrována. Anonymita je za jistých podmínek zaručena [17] a funguje i v případě, že až $N-1$, z celkového počtu N, uzlů mix sítě je nějakým způsobem kompromitováno.

6.1.2 e-cash

Jako e-cash (*electronic cash*) se označuje elektronická měna, což je reprezentace fyzické měny hmotného světa. Někdy tento pojem bývá označován jako e-money. Hotovost v elektronické podobě může být uložena na čipové kartě nebo na pevném disku v počítači – neexistuje v hmatatelné formě.

V případě e-cash jde o technologii, jak lze např. realizovat peněžní transakce po Internetu nebo jeho prostřednictvím nakupovat zboží. Anonymita je zde mnohem větší než při použití běžných kreditních karet, kdy při každé takové peněžní operaci vzniká nový záznam obsahující číslo použité kreditní karty a další potenciálně citlivé a zneužitelné údaje. Příkladem systému postaveného na principech elektronické měny je mezinárodní platební systém PayPal.

Útočník může po úspěšném kryptovirálním útoku požadovat zaplacení určité peněžní sumy jako výkupného výměnou za obnovení souborů. Platba bezhotovostním převodem na účet by zcela jistě znamenala riziko vystopování totožnosti útočníka. Nejschůdnější způsob, jak anonymně a bezpečně získat výkupné, je právě použití technologie e-cash. Protože může být e-cash uložen v souboru, na disku počítače, může být samozřejmě také posílán po síti.

Kombinací mix sítí a e-cash získává útočník dobrou šanci na získání kýžené finanční „odměny“ spolu s vysokou mírou anonymity. Prakticky to znamená, že může takto nezákonně generovat zisk a přitom díky zneužití vyspělých informačních technologií vůbec nic neriskovat [4].

6.2 Chování uživatelů

Bezpečnost není jen otázka technického a fyzického zabezpečení, kvality použitého softwaru nebo síly kryptografických algoritmů. Podstatné procento bezpečnostních incidentů by nikdy nenastalo bez selhání lidského faktoru. Ten je, jak se zdá, v boji proti útočníkům a jejich malwaru stále nejslabším článkem.

Neznamená to však, že by incidenty vznikaly s tichým souhlasem nebo přičiněním dotyčných zodpovědných pracovníků. Jde spíše o důsledek jejich nekvalifikovanosti a nedostatečné obezřetnosti při nakládání s důvěrnými informacemi (hesla k účtům, telefonní čísla organizace, jména zaměstnanců, topologie podnikové sítě, ...). Lidé by si měli uvědomit, s jakými informacemi přicházejí do styku a jakou mají takové informace v dnešním světě hodnotu.

Míra ochrany a zabezpečení by především měla odpovídat celkové hodnotě informací nacházejících se v konkrétním počítači. Důležitá data je potřeba chránit také pravidelným zálohováním – pouze tak se lze účinně bránit i před kryptoviry.

6.2.1 Sociální inženýrství

Sociální inženýrství neboli sociotechnika je způsob manipulace s lidmi, který využívá vlivu a přesvědčování k oklamání oběti, že sociotechnik je ten, za koho se vydává, a nikoli tím, kým ve skutečnosti je. Ve výsledku dokáže sociotechnik využít svého vlivu a vymámit z oběti cenné informace, ať už s použitím informačních technologií nebo bez nich [12]. V umění sociotechniky vynikal známý hacker, a patrně také nejznámější příklad exemplárně potrestaného počítačového „zločince“ v historii, Kevin Mitnick. Mitnick dokázal pomocí telefonu i osobně (po krátké přípravě), díky sebevědomému vystupování a použitím sociotechniky, získat mnoho uživatelských jmen a hesel pro přístup do podnikových systémů; a mnohdy dokonce přístup k administrátorským účtům na jinak velmi dobře zabezpečených serverech, jež obsahovaly zdrojové kódy aplikací. Jejich zkopírováním na jiné servery údajně vznikly ztráty v řádech desítek milionů dolarů mnoha předním softwarovým společnostem.

Manipulace byla sociology studována více než padesát let. Na otázku, proč jsou lidé zranitelní vůči sociálnímu inženýrství, dává odpověď právě výsledek tohoto dlouhodobého studia a výzkumu. Následuje výčet šesti základních sklonů lidské povahy, na které se útočníci při manipulaci lidí (sociotechnice) zaměřují [12]:

- **Autorita**
  Lidé mají sklon vyhovět někomu, kdo má v jejich očích dostatečnou autoritu, nebo když jeho žádost působí oprávněně.

- **Náklonnost**
  Lidé mají sklon vyhovět žádosti i cizího člověka, když je tento dokáže přesvědčit, že mají podobné zájmy, názory a stanoviska, nebo když se jim dotyčný jeví jako příjemná a sympatická osoba.

- **Vzájemnost**
  Lidé mají sklon vyhovět žádosti v případě, kdy je jim na oplátku za tuto „laskavost“ slíbeno něco pro ně cenného a významného (rada, pomoc, dar, atd.).

- **Zásadovost**
  Lidé mají zpravidla sklon vyhovět poté, co dají někomu svůj veřejný příslib nebo souhlas s angažováním se v dané záležitosti.

- **Společenské potvrzení**
  Lidé mají sklon vyhovět žádosti, která v důsledku vede k takovému jednání, jež se zdá být ve společnosti běžné a přijatelné, tzn. tím, co ostatní normálně dělají. Chování okolí funguje jako potvrzení legitimity požadované akce.

- **Vzácnost**
  Lidé mají sklon vyhovět, když uvěří, že jistá věc, o niž usilují, se zdá být v omezeném množství a ostatní o ni musí soupeřit; nebo že je její dostupnost omezena pouze na krátký časový interval.

Několika výše uvedených rysů lidské povahy nezneužívá malware (vč. kryptovirů) přímo, ale spíše se těchto poznatků o lidské povaze snaží využít autoři (útočníci) ve fázi návrhu implementace způsobu šíření svých škodlivých programů. Uživatelé si pak pod nejrůznějšími pohnutkami nevědomky nainstalují zákeřný malware stažený z Internetu společně s jiným softwarem. Studie v kapitole 3, viz *Satan Virus*, může být modelovým příkladem převzetí kontroly virem a získání rozhodující výhody nad uživatelem [3].

6.3 Zneužití komunikačních protokolů

Zneužívání slabých míst komunikačních protokolů představuje oblast zájmu mnoha útočníků. Protože jsou komunikační protokoly nedílnou součástí jakékoli počítačové sítě, jsou i v prostředí Internetu přítomny takřka v každé aplikaci. Bez jejich existence by měly počítače značně omezené možnosti.

V této podkapitole se zaměříme na používaný komunikační protokol SMTP a sítě typu P2P. Protokol SMTP bývá velmi často zneužíván tvůrci malwaru jako prostředek šíření nákazy po síti – v podobě e-mailových zpráv. Sítě P2P pak slouží k podobnému účelu, a to také jako distribuční kanál malwaru, přičemž rozdíl mezi oběma způsoby šíření malwaru spočívá v míře vlastního přičinění uživatele vedoucího k dalšímu bezpečnostnímu incidentu.

6.3.1 Protokol SMTP

Protokol SMTP (popsán v RFC 821, resp. RFC 2821) je textově orientovaný protokol určený k přímému zasílání e-mailových zpráv mezi koncovými stanicemi. Pracuje nad transportním protokolem TCP a běží na portu č. 25 [34].

SMTP servery fungující v režimu *open relay* jsou nejjednodušším cílem rozesílatelů spamu. Server v této konfiguraci přijímá zprávy od libovolného odesilatele a posílá je na zvolenou adresu příjemce. Nijak si neověřuje, zda-li je uživatel oprávněn využívat služeb

serveru, jestli je cílová adresa zprávy platná, nebo zda je uživatel skutečně tím, za koho se vydává.

Chybějící autentizace umožňuje činnost malwaru generujícímu ohromné množství spamových zpráv. Malware si nese seznam serverů typu *open relay* k šíření buďto statický – je pevně dán už v době kompilace – nebo si *open relay* servery dokáže vyhledávat v Internetu, až v době úspěšného napadení počítače oběti.

Ne ve všech případech může malware využívat služeb nějakého otevřeného serveru. Extrémním případem je implementace vlastního SMTP serveru neseného v těle viru, pak neexistuje závislost na externích serverech, což navíc snižuje možnost odhalení a blokování IP adresy umístěním na černou listinu. Nevýhoda podobného řešení spočívá v nárůstu velikosti binárního kódu.

Malwarem generované zprávy určené k vlastnímu šíření mají pochopitelně podvrženou adresu odesilatele. Původní implementace SMTP serverů ověřování odesilatele nevyžadovaly. Pro potřeby zajištění alespoň základní autentizace poštovního klienta MUA (*Mail User Agent*) vzniklo rozšíření SMTP-AUTH. Tak je zajištěn základní mechanismus přihlašování klienta na poštovní server. SMTP-AUTH je částečným řešením dřívějšího problému, neboť technikou spoofingu lze také podvrhnout informaci o skutečném odesilateli zprávy nahrazením dané položky v hlavičce adresou jiného, skutečně existujícího uživatele [35].

Na zvolené strategii šíření a činnosti malwaru závisí jeho síťová aktivita. Druhy šířící se pomocí elektronické pošty zpravidla v této fázi negenerují takové množství zpráv, které by vyčerpávalo přenosová pásma linek a docházelo by k jejich zahlcení. Zvýšená síťová aktivita totiž znamená zvýšenou pozornost uživatele, pozornost uživatele pak zvyšuje riziko odhalení. Později, po úspěšném ukončení replikační fáze, může malware generovat spam dle libosti. Mnohé druhy toto dělají jako své hlavní poslání. Snaží se odesílat maximální množství zpráv, jaké jim druh připojení počítače k síti umožňuje. Útok formou zahlcení serveru příjemce množstvím příchozích e-mailových zpráv se nazývá *e-mail bomb* a jde o útok z kategorie DoS.

Protože lze relativně snadno filtrovat příchozí zprávy se stále stejnou odchozí adresou, využívají útočníci k tomuto útoku rozsáhlých botnetů. Proti záplavě milionů e-mailových zpráv z různých IP adres, přicházejících v jeden okamžik, se lze bránit takto již jen stěží. Koordinované útoky za pomocí botnetů spadají do kategorie typu DDoS.

Některé poštovní servery nyní proto bývají konfigurovány tak, aby odmítaly přenosy na portu č. 25, pouze autentizovaným uživatelům na portu č. 587 je dovoleno odesílat zprávy, a pouze na existující adresy. Tím však možnosti k dalšímu zneužívání nekončí.

V minulosti spoléhaly některé druhy malwaru na dostupnost knihovních funkcí určených pro práci s poštovním klientem MS Outlook Express. Ten je zahrnut ve standardní instalaci OS Windows XP. Konkrétně jde o knihovnu MAPI32.DLL. Malware pak mohl k šíření použít nakonfigurovaný účet uživatele této aplikace.

Příklad:

Kód umožňující zneužití knihovny v MS Windows skutečně existujícího malwaru ...

```
HINSTANCE MAPIlHnd;
MAPIlHnd = LoadLibraryA("MAPI32.DLL");
```

Po úspěšném zavolání funkce *LoadLibraryA* stačilo jen několik dalších řádků kódu (v jazyce C), aby byl virus schopen generovat množství poštovních zpráv (posílal se jako spus-

titelná příloha) a přidávat je do fronty odesílaných zpráv k ostatním legitimním zprávám uživatele.

6.3.2 P2P sítě

Největší potenciál P2P sítí, resp. jejich konkrétních implementací, spočívá, z pohledu malwaru, právě v téměř ideálním prostředí k šíření nákazy. Útočníci mohou své výtvory maskovat pod názvy aktuálně v Internetu nejvíce stahovaných programů (nebo malware integrovat do skutečných instalačních balíků legitimního softwaru). Uživatelé si jej pak vlastní vinou „vpouští“ do svých počítačů.

Malware by se teoreticky dokázal šířit pomocí P2P sítí zcela autonomně – vyhledáváním zranitelných systémů sdílení souborů a vytvářením svých vlastních kopií pod jmény zdánlivě užitečných programů. Nakonec by mohl, v závislosti na implementaci služby, generovat vlastní soubory s metadaty (jako je např. torrent u protokolu BitTorrent). Zpřístupněním se pak nasdílet ke stažení potenciálním obětem.

6.4 Bezpečnostní protiopatření

Oblast komunikačních technologií je velmi dynamická – nové možnosti využití se objevují ve velmi krátkých intervalech. Stejnou rychlostí se objevují zákeřné praktiky založené na zneužití slabých míst těchto, co se týče zabezpečení, dosud nevyspělých technologií.

Nejen zneužívání síťových technologií k nelegálním aktivitám ze strany uživatelů, ale hlavně zneužívání těchto vlastností počítačovými útočníky vyžaduje nasazení mnoha bezpečnostních protiopatření.

6.4.1 Klasická protiopatření

Možných bezpečnostních incidentů existuje celá řada – riziko výskytu nějakého problému není nikdy možné stoprocentně eliminovat. Dodržováním základních zásad bezpečnosti při pohybu na síti můžeme docílit relativně vysokého stupně ochrany před malwarem. Nutnou podmínkou je však používání ověřeného a aktualizovaného softwaru, včetně antivirového a anti-spyware softwaru, a firewallu. V neposlední řadě patří mezi nejúčinnější obranné prostředky současnosti hlavně vysoká míra obezřetnosti a nedůvěřivosti ke všemu, co pochází z neznámého zdroje.

Podceňování důležitosti bezpečnosti se může každé organizaci vymstít. Autoři malwaru dnes netvoří zákeřné programy pro vlastní potěšení, ale je to převážně cíleno na generování podvodného finančního zisku. Útoky se odehrávají ve velkých měřítkách, s využitím botnetů. Samotné vytváření botnetů, jejich prodej a využívání za účelem šíření nevyžádané pošty, popř. jiných praktik, spadá do oblasti organizovaného zločinu. V žádném případě není Internet bezpečným místem pro připojení nechráněného počítače. Přítomnost cenných dat na takovém počítači není rozhodující. Samotná výpočetní síla počítače a jeho prostředky hrají důležitou roli. Koordinovanému útoku tisíců ovládnutých počítačů útočníkem se mohou cílové komerční servery bránit jen pomocí extrémních a krátkodobých protiopatření. Ztráty způsobené DDoS útoky jsou vždy prakticky nevyhnutelné.

6.4.2 Kryptovirální protiopatření

Největší síla kryptoviru spočívá v asymetrickém soukromém klíči, který není součástí viru. Mechanismus kontroly nad používáním kryptografie v systému by měl být součástí jádra

OS. Před zašifrováním souboru by jádro systému vyžadovalo od uživatele důkaz, že on je tím, kdo vlastní příslušný soukromý kryptografický klíč (tzv. *zero-knowledge proof*). To by mohlo být dokazováno vždy v době přihlášení uživatele do systému, např. pomocí čipové karty.

Jinou možností, jak dosáhnout podobného cíle, je ověření podpisu veřejného klíče, který patří certifikační autoritě, jenž je pro jádro systému důvěryhodná. Pokud by autor kryptoviru použil certifikovaný veřejný klíč k útoku, stal by se ihned hlavním podezřelým. Celá strategie by fungovala pouze za předpokladu, že jádro systému nebylo modifikováno kryptovirem.

Faktem však i nadále zůstává skutečnost, že veškerá kryptografická funkčnost může být naimplementována do těla viru, tedy být nezávislá na operačním systému [24].

Kapitola 7

Závěr

V této diplomové práci jsem se zabýval kryptovirologií, což je poměrně nový obor počítačové bezpečnosti. Hlavním cílem bylo nastudovat bezpečnostní dopady plynoucí z propojení kryptografických algoritmů a počítačových virů.

Demonstrační program `cryptovirus.exe` vznikl jako ukázka toho, jak by mohly vypadat nově vznikající hrozby šířící se po síti. Program využívá možností běžných operačních systémů platformy Microsoft Windows – CryptoAPI, simuluje chování počítačového viru. Po spuštění v hostitelském systému vyhledává vybrané soubory, zašifruje je symetrickým klíčem a klíč (zašifrovaný ještě asymetrickým veřejným klíčem) zpřístupní uživateli. Ten, aby získal svá zašifrovaná data zpět, musí splnit podmínky útočníka.

Z časových důvodů nebyl protokol komunikace oběti s útočníkem, nutný k předání „výkupného“, v programu implementován. Celá procedura by byla realizována nejspíše s využitím mix sítí a elektronické měny e-cash (zmíněno v kapitole 6). Virus by mohl zvyšovat svoji šanci na přežití v systému aktualizací kódu a přidáváním nových funkcí. K tomuto účelu se nabízí zabudování interakce útočníka s virem, např. pomocí komunikačního kanálu IRC nebo veřejných diskusních skupin, kdy by virus mohl v těchto médiích vyhledávat šifrované instrukce ke svému dalšímu chování. Zde je poměrně velký prostor k dalšímu rozšíření programu.

Pro zajištění bezpečného testování programu nebyla implementována ani replikační procedura viru, takže nemůže dojít k jeho náhodnému úniku. K tomuto účelu by bylo možné použít jeden z klasických postupů šíření malwaru, viz kapitola 3. Při experimentech s replikací a šířením však bylo zjištěno, že poskytovatelé internetového připojení monitorují odchozí datové toky svých klientů, a při podezřelé síťové aktivitě reagují zablokováním některých TCP a UDP portů na straně klienta.

Přínosem této práce je poskytnutí uceleného pohledu na celou problematiku kryptovirologie a poukázání na reálnost nebezpečí plynoucího z volně dostupných kryptografických funkcí OS, o kterých uživatelé běžně ani nemají tušení. Jako důkaz proveditelnosti může posloužit zmíněný demonstrační program.

Kryptografie, jako součást dnešních OS, je velmi mocný nástroj k zajištění větší bezpečnosti při komunikaci v nedůvěryhodném prostředí Internetu, zároveň ale může nesprávným použitím nebo zneužitím útočníky představovat velká rizika. Uživatelé informačních technologií by si měli tato rizika uvědomovat. Také výrobci operačních systémů by měli zneužívání kryptografických funkcí předcházet implementací dodatečných bezpečnostních protiopatření.

Literatura

[1] BÄCHER, P.; HOLZ, T.; KÖTTER, M.; aj.: Know your Enemy: Tracking Botnets. [online], [cit. 2008-04-16].
URL <http://www.honeynet.org/papers/bots>

[2] BALEPIN, I.: Superworms and Cryptovirology: a Deadly Combination. [online], [cit. 2008-01-06].
URL <http://www.megasecurity.org/papers/worms-cryptovirology.pdf>

[3] BOND, M.; DANEZIS, G.: A Pact With the Devil. In *Proceedings of the 2006 workshop on New Security Paradigms*, 2006, ISBN 978-1-59593-923-4, s. 77–82.

[4] CHAUM, D.: Achieving Electronic Privacy. [online], [cit. 2008-05-04].
URL <http://ntrg.cs.tcd.ie/mepeirce/Project/Chaum/sciam.html>

[5] FU, S. C.: Realism in Epidemic Models. [online], 2002, [cit. 2008-05-01].
URL <http://www.csse.uwa.edu.au/~scfu/caepidemic/litreview.html>

[6] FU, S. C.; MILNE, G.: Epidemic Modelling Using Cellular Automata. *Proceedings of the Australian Conference on Artificial Life*, 2003.

[7] GRANNEMAN, S.: Infected in 20 minutes. [online], [cit. 2008-04-29].
URL <http://www.theregister.co.uk/2004/08/19/infected_in20_minutes>

[8] GUTMANN, P.: Secure Deletion of Data from Magnetic and Solid-state Memory. In *Proceedings of the 6 th Conference on USENIX Security Symposium, Focusing on Applications of Cryptography – Volume 6*, USENIX Association, 1996, s. 8–8.

[9] JONES, G.: The 10 Most Destructive PC Viruses Of All Time. [online], [cit. 2008-04-28].
URL <http://www.techweb.com/showArticle.jhtml?articleID=160200005>

[10] LUDWIG, M. A.: *The Little Black Book of Computer Viruses (Electronic Edition)*. American Eagle Publications, Inc., Post Office Box 1507, Show Low, Arizona 85901, 1996, ISBN 0-929408-02-0, 183 s.

[11] MENEZES, A. J.; van OORSCHOT, P. C.; VANSTONE, S. A.: *Handbook of Applied Cryptography*. CRC Press, 1996, ISBN 0-8493-8523-7, 816 s.

[12] MITNICK, K. D.; SIMON, W. L.: *The Art of Deception: Controlling the Human Element of Security*. Wiley Publishing, Inc., 2002, ISBN 0471237124, 352 s.

[13] NEEDHAM, R. M.; SCHROEDER, M. D.: Using Encryption for Authentication in Large Networks of Computers. *Communications of the ACM*, ročník 21, č. 12, 1978: s. 993–999.

[14] PEARCE, S.: Viral Polymorphism. [online], [cit. 2008-03-13].
URL <http://vx.netlux.org/lib/pdf/Viral%20polymorphism.pdf>

[15] PŘIBYL, T.: Pozor! Soubor napaden neznámým virem! [online], [cit. 2008-04-29].
URL <http://www.virusy.sk/clanok.ltc?ID=113>

[16] SCHLUTING, C.: Botnets: Who Really "0wns" Your Computers? [online], 1995, [cit. 2008-04-16].
URL <http://www.enterprisenetworkingplanet.com/netsecur/article.php/3504801>

[17] SHMATIKOV, V.; WANG, M.-H.: Measuring Relationship Anonymity in Mix Networks. In *Proceedings of the 5th ACM Workshop on Privacy in Electronic Society*, 2006, ISBN 1-59593-556-8, s. 59–62.

[18] STALLINGS, W.: *Cryptography and Network Security: Principles and Practice (3rd Edition)*. Prentice Hall, ©2003, ISBN 0130914290, 681 s.

[19] STINSON, D. R.: *Cryptography: Theory and Practice*. CRC Press, 2005, ISBN 1584885084, 616 s.

[20] SYVERSON, P.: A Taxonomy of Replay Attacks. In *Proceedings of the 7th IEEE Computer Security Foundations Workshop*, 1994, s. 131–136.

[21] TIPPETT, P. S.: The Kinetics of Computer Virus Replication: A Theory and Preliminary Survey. *Safe Computing: Proceedings of the 4th Annual Computer Virus and Security Conference*, 1991: s. 66–87.

[22] WEBER, M.; SCHMID, M.; SCHATZ, M.; aj.: A Toolkit for Detecting and Analyzing Malicious Software. In *18th Annual Computer Security Applications Conference, Las Vegas, NV, USA*, IEEE Computer Society, 2002, ISBN 0-7695-1828-1, s. 423–431.

[23] WHITE, S. R.; KEPHART, J. O.; CHESS, D. M.: Computer Viruses: A Global Perspective. *Proceedings of the 5th Virus Bulletin International Conference, Boston*, 1995.

[24] YOUNG, A.: Cryptoviral Extortion Using Microsoft's Crypto API: Can Crypto APIs Help the Enemy? *International Journal of Information Security*, ročník 5, č. 2, 2006: s. 67–76.

[25] YOUNG, A.; YUNG, M.: *Malicious Cryptography: Exposing Cryptovirology*. Indianapolis, Indiana: Wiley Publishing, Inc., ©2004, ISBN 0-7645-4975-8, 392 s.

[26] ZHU, Y.; FU, X.; BETTATI, R.; aj.: Anonymity Analysis of Mix Networks Against Flow-Correlation Attacks. In *Proceedings of IEEE Global Communications Conference, St. Louis, MO, USA*, IEEE Computer Society, 2005.

[27] International Computer Security Association. [online], [cit. 2008-02-15].
URL <http://www.icsa.net>

[28] Karatsuba Multiplication. [online], [cit. 2008-03-03].
URL <http://mathworld.wolfram.com/KaratsubaMultiplication.html>

[29] Microsoft Cryptographic Service Providers. [online], [cit. 2008-05-08].
URL <http://msdn.microsoft.com/en-us/library/aa386983(VS.85).aspx>

[30] NIST: Recommendation for Block Cipher Modes of Operation. [online], [cit. 2008-4-25].
URL <http://csrc.nist.gov/publications/nistpubs/800-38a/sp800-38a.pdf>

[31] NIST: Recommendation for Using Approved Hash Algorithms. [online], [cit. 2008-4-27].
URL <http://csrc.nist.gov/publications/drafts/Draft-SP-800-107/Draft-SP800-107.pdf>

[32] NSA: The Case for Elliptic Curve Cryptography. [online], [cit. 2007-12-15].
URL <http://www.nsa.gov/ia/industry/crypto_elliptic_curve.cfm>

[33] RFC 1487: X.500 Lightweight Directory Access Protocol. [online], [cit. 2008-4-26].
URL <http://rfc.net/rfc1487.html>

[34] RFC 2821: Simple Mail Transfer Protocol. [online], [cit. 2008-03-03].
URL <http://www.rfc.net/rfc2821.html>

[35] RFC 4954: SMTP Service Extension for Authentication. [online], [cit. 2008-03-03].
URL <http://www.rfc.net/rfc4954.html>

[36] Register of Known Spam Operations. [online], [cit. 2008-04-28].
URL <http://www.spamhaus.org/rokso/index.lasso>

[37] The Threat – Rootkits. [online], [cit. 2008-03-16].
URL <http://www.virus.fi/blacklight/rootkit.shtml>

[38] Virus Primer. [online], [cit. 2008-04-29].
URL <http://us.trendmicro.com/us/support/virus-primer/index.html>

[39] Web Game Provides Breakthrough in Predicting Spread of Epidemics. [online], 2006, [cit. 2008-05-01].
URL <http://www.scienceblog.com/cms/node/9874/>

[40] Where's George? [online], [cit. 2008-05-01].
URL <http://www.wheresgeorge.com/>

Dodatek A

Obsah přiloženého CD

Součástí této práce je také datový nosič typu CD-ROM, který obsahuje veškeré zdrojové kódy vytvořené v implementační části práce, včetně souboru *Makefile* sloužícího k bezproblémovému překladu do binární podoby. CD-ROM obsahuje rovněž elektronickou verzi tohoto dokumentu ve formátu PDF.