CARNet
HRVATSKA AKADEMSKA I ISTRAŽIVAČKA MREŽA
CROATIAN ACADEMIC AND RESEARCH NETWORK

Kriptografija u službi napadača

CCERT-PUBDOC-2008-04-226

+CERT.hr u suradnji s LSS

Sigurnosni problemi u računalnim programima i operativnim sustavima područje je na kojem CARNet CERT kontinuirano radi.

Rezultat toga rada je i ovaj dokument, koji je nastao suradnjom CARNet CERT–a i LS&S–a, a za koji se nadamo se da će Vam koristiti u poboljšanju sigurnosti Vašeg sustava.

CARNet CERT, www.cert.hr

Nacionalno središte za **sigurnost računalnih mreža** i sustava.

LS&S, www.LSS.hr

Laboratorij za sustave i signale pri Zavodu za elektroničke sustave i obradbu informacija Fakulteta elektrotehnike i računarstva Sveučilišta u Zagrebu.

Sadržaj

1 Uvod

Virusi i svi drugi oblici zlonamjernih programa postali su neizostavni dio Interneta. Štoviše, oni žive i u sustavima koji nemaju vezu s Intranetom. Šire se i drugim, sporijim medijima: CD i DVD, flash memorijama, mobilnim telefonima i sl.

Količina virusa, njihova dugovječnost (neke statistike pokazuju da većinu štete rade virusi stari i po nekoliko godina) i brzina nastajanja novih doveli su do toga da svi korisnici danas koriste neku vrstu antivirusne zaštite.

Drugi efekt koji je izazvalo to stanje jest da autori programa, projektanti komunikacijskih protokola te administratori sustava rutinski posežu za kriptografskim tehnikama ne bi li svoje proizvode i sustave zaštitili od napada.

Stoga se kriptografija smatra dobrom stvari i obrambenom metodom.

Međutim, mogu li se i napadači koristiti kriptografijom da stvore još opasnije oružje ili čak pretvoriti kriptografiju u napadačku tehnologiju?

Ovaj dokument daje osnovni uvid u tehnike koje autori virusa primjenjuju u nastojanju kako bi svoje programe učinili nepobjedivima i u načine na koje primjenjuju kriptografiju.

1.1 Kriptografija

Kriptologija je znanost koju dijelimo na kriptografiju i kriptoanalizu.

Kriptografija razvija algoritme koji trebaju osigurati povjerljivost (tajnost), autentikaciju i cjelovitost podataka. U načelu se zasnivaju na nekoj tajni, koju najčešće zovemo ključem i/ili posebnoj matematičkoj funkciji (algoritmu, "formuli") vrlo često "jednosmjernoj" koju zovemo "šifra".

Kriptoanaliza je obrnuti napor, usmjeren na "dešifriranje" tj. vraćanje kriptiranog sadržaja u otvoreni, čitljivi oblik. Kriptoanaliza nastoji dati rješenja u slučaju kad nemamo ključ ili šifru ili oboje.

Ovako opisane, kriptografiju uobičajeno smatramo "obrambenom" tehnikom dok kriptoanalizu smatramo "napadačkom" tehnikom. U ovom dokumentu će biti pokazano da su tehnike samo tehnike, a služe li napadu ili obrani ovisi o čovjeku koji ih primjenjuje i njegovoj namjeri.

Kriptografija kao metoda zaštite se koristi posvuda u računalnim sustavima: za zaštitu pohranjenih podataka, rada aplikacija, pristupa računalnim sustavima, te u mrežnim komunikacijama. U mrežnim komunikacijama koristi se već na drugom mrežnom sloju (WEP, WPA/WPA2) te na trećem i višim slojevima (IPSec, SSH, SSL, Kerberos, Radius, PGP, GPG, ...)

Za zaštitu (kriptiranje) podataka, bilo pohranjenih na digitalnom mediju, bilo za vrijeme prijenosa komunikacijskim kanalom koriste se brojni algoritmi: RC4, IDEA, DES, 3DES, AES.

Za zaštitu lozinki, programskih aplikacija, a ponekad i podataka koriste se tzv. hashing algoritmi koji daju svojevrstan jedinstveni, kratki "potpis" skupine podataka: MD5, Whirlpool, SHA-1, SHA-2.

1.2 Virusi i napadi

Virologija u računalnom području je znanost koja se bavi virusima i drugim zlonamjernim programima.

Zlonamjernim programima nazivamo sve računalne aplikacije koje se šire i izvode u računalnim sustavima bez volje i dozvole vlasnika tog sustava, bez obzira hoće li taj program učiniti neku izravnu štetu sustavu ili vlasniku.

I virologija ima dvije strane: stvaranje virusa i borbu protiv njih.

Virusi su samoumnažajući programi koji se šire sa zaraženog informacijskog sustava na druge, kopirajući svoj programski kod u druge programe ili dokumente.

Jedna od definicija virusa glasi:

> *Virus je niz naredbi, koje kad se interpretiraju u odgovarajućoj okolini, mijenjaju slijed drugih naredbi tako da stvaraju novu kopiju sebe (po mogućnosti drugačijeg izgleda) u toj drugoj okolini.*
>
> Fred Cohen

Antivirusni program nastoji prepoznati i onemogućiti viruse (i druge zlonamjerne programe) prije nego se aktiviraju. Njihova metoda rada se uglavnom zasniva na prepoznavanju obrazaca: dijelova virusnog programa, takozvanih "potpisa", kao i na otkrivanju sumnjivog ponašanja tzv. "heuristici".

1.3 Kriptografija u službi napadača

Već je spomenuto kako legitimni korisnici upotrebljavaju kriptografiju za zaštitu svojih podataka i sustava. I proizvođači antivirusnih programa koriste kriptografiju za zaštitu samih programa, njihovih baza podataka, svojih središnjica iz kojih se osvježavaju instalacije diljem svijeta te komunikacijske putove kojima su oni povezani.

No i napadač ima mogućnost koristiti kriptografiju za svoje potrebe.

2 Kriptovirologija

Kriptovirologija je aktivnost koja se bavi proučavanjem primjenom kriptografskih i kriptoanalitičkih metoda i tehnika na stvaranje zlonamjernih programa i aktivnosti te borbu protiv njih.

Ovdje su objašnjeni različiti načini na koje napadač može koristiti kriptografiju, no težište je stavljeno na stvaranje tzv. „oklopljenih virusa“ (eng. armored viruses): koje se ne može prepoznati i/ili zaustaviti uopće, ili barem na vrijeme.

2.1 Napadač štiti sebe kriptografijom

Napadači su ljudi i isto su korisnici računalnih sustava. Tipični napadač razvija alate za napade na svom, skrivenom računalu koje štiti lozinkama, kriptiranjem izvornog koda napadačkih programa i slično. No, svoje napade ne pokreče sa svog računala, već s nekog tuđeg računala u koje je prodro, iskoristivši njegovu ranjivost. Kako bi zameo tragove, u pravilu, do računala s kojeg će pokrenuti napad dolazi kroz nekoliko drugih računala čiju je sigurnosnu zaštitu također probio. Svoju prisutnost na tim računalima, a posebno na računalu s kojeg će pokrenuti napad također štiti lozinkama te kriptiranjem napadačkih programa i podataka koje on sakupi. Kriptirat će i komunikaciju umnoženih i raširenih napadačkih programa s izvorišnim računalom (s kojeg je pokrenut napad).

Ovakvi bi se oblici korištenja kriptografije uvjetno mogli nazvati "normalnim" (što ne znači i dozvoljenim) korištenjem u rukama napadača.

2.2 Korištenje kriptografije protiv žrtve

Osim što, napadač može koristit kriptografiju na "normalan" način: da zaštiti svoje alate i svoje sjedište (svoje vlastito računalo i mrežu), te da zaštiti sustave, svoje alate i podatke na sustavima u koje je prodro i koristi ih kao bazu za daljnje djelovanje ili zametanje tragova, može koristiti kriptografiju i agresivno, protiv žrtve.

Može primijeniti kriptografiju na žrtvinim alatima, sustavu ili podacima. Napadački program, nakon što prodre u sustav žrtve, može potražiti točno određeni program i/ili podatke ili ih odabrati nasumce, te na njih primijeniti kriptografski postupak. Na računalu će ostati samo kriptirani podaci koje njihov vlasnik ili legitimni korisnik više ne može upotrijebiti.

Napadač će se nekim komunikacijskim putem (vidljivom porukom u kriptiranim podacima, e-mailom, telefonom, faxom, ...) obratiti žrtvi te tražiti otkupninu u zamjenu za dekriptiranje žrtvinih podataka.

Tu se za napadača pojavljuje tehnički izazov: kako dekriptirati podatke žrtve na način da to žrtva može prihvatiti, a da napadač ne otkrije svoj identitet. U praksi postoje dva modela: osnovni i hibridni.

2.2.1 Osnovni model

U osnovnom modelu, napadač stvori RSA ključ. Javni dio ključa pohrani u virus, a privatni čuva kod sebe. Kad virus prodre u sustav žrtve, kriptirat će njen sadržaj javnim ključem.

Napadač od žrtve traži otkupninu i kad ju dobije šalje žrtvi privatni dio ključa kojim žrtva onda može vratiti svoje podatke.

Nekoliko napadača je već koristilo ovu tehniku u svojim virusima: GpCode and Krotten.

No, ovaj model ima neke slabosti, za napadača.

Temeljna slabost koja je izvan domene ovog dokumenta jest pitanje: kako dobiti novac i ostati anoniman.

Druga slabost je u tome što bi žrtva mogla objaviti privatni dio ključa i time omogućiti svim ostalim žrtvama da besplatno povrate svoje podatke.

Najjednostavnije rješenje za napadača bi bilo da traži od žrtve da mu pošalje kriptirane podatke. Nakon što žrtva pošalje kriptirane podatke i plati otkupninu, napadač će dekriptirati podatke i vratiti ih žrtvi.

No, neke žrtve bi radije izgubile podatke nego ih dale u ruke napadaču, što napadač, pak, može riješiti time da već kod prodora u sustav sam pošalje sebi kriptirane podatke.

Ovi napadi imaju smisla samo kad se radi o većoj količini podataka, a prijenos veće količine podataka i to čak dva puta, lakše je pratiti kao trag do napadača, nego samo slanje ključa.

Stoga napadači radije koriste hibridni model.

2.2.2 Hibridni model

Hibridno je rješenje prilično elegantno i sigurno za napadača. I ovdje se u virusu nalazi javni ključ napadača, ali se žrtvini podaci ne kriptiranju njime već jednim novim, slučajno izabranim ključem. Taj se ključ kriptira javnim ključem napadača i pohrani na sustavu ili pošalje napadaču.

Kad otkupnina bude plaćena, napadač samo treba dekriptirati ključ kojim su kriptirani žrtvini podaci i poslati ga žrtvi. Radi se o relativno vrlo malom broju podataka, koji se gotovo potpuno sigurno mogu poslati bez traga koji bi istražitelje doveo do napadača.

3 Oklopljeni virusi

Treća glavna primjena kriptografije za potrebe napadača leži u jačanju napadačkih alata i njihove obrane od antivirusnih tehnika. Krajnji je cilj stvoriti nepobjedivi virus, kojeg se ne može ni otkriti ni spriječiti.

Tri su komponente koje napadač koristi: pronalaženje žrtava, usporavanje analize i nevidljivost.

Pronalaženje pogodnih žrtava je i inače prvi posao pri pisanju virusa. Virus ne može napadati bilo koji računalni sustav, već je specijaliziran za iskorištavanje točno određenih slabosti. Stoga napadač, ali i sam virus, traže sustave koji imaju te slabosti.

Osim slabosti, traže se i druga svojstva žrtve: mogućnost skrivanja virusa, povezanost žrtve s drugim pogodnim žrtvama, postojanje sustava povjerenja između žrtve i drugih sustava i sl.

U kontekstu kriptovirologije u poglavlju o nevidljivosti bit će pokazano kako virusi mogu tražiti i iskoristiti sustave koji koriste legitimne aplikacije s tajnim sustavima zaštite (npr. Skype) kako bi njihovu zaštitu, kad prodru u taj sustav, koristili da budu nevidljivi anti virusnim zaštitama.

Za dobro odabrani skup žrtava moguće je konstruirati gotovo nepobjedive viruse.

3.1 Usporavanje analize (eng. analysis delay)

Kad se u računalnim sustavima diljem svijeta pojavi novi virus, svi proizvođači antivirusne zaštite i nezavisni laboratoriji i istraživači moraju što brže analizirati virus, odrediti mu potpis i te informacije dostaviti korisnicima anti virusnih programa.

Već godinama je poznato da glavni učinak virusa je onaj kojeg postigne u prvih nekoliko sati "na slobodi", ponekad tek samo u prvih 30 minuta.

Ako autor virusa uspije napisati takav programski kod da analitičarima treba više sati ili dana za analizu, višestruko je povećao učinak virusa. Štoviše, možda je tako dobiveno vrijeme ključno da se virus ugradi u žrtvin sustav na takav način da ga je poslije nemoguće izbrisati čak i kada ga se pronađe.

Tehnike koje se koriste za usporavanje analize su: zamagljivanje koda, promjena oblika koda (eng. shape shifting) i kriptiranje koda.

Zamagljivanje koda ima za svrhu programski kod virusa učiniti nepogodnim za automatsku analizu specijaliziranim programima te prisiliti analitičare na ručni rad, analizom jedne po jedne naredbe. Na taj se način proces analize učini sporim i dugotrajnim. Brojne su metode koje "zamagljuju" programski kod virusa.

Leksičke transformacije su postupci kad se tijekom izvođenja programskog koda mijenjaju ili zamjenjuju imena varijabli: u jednom dijelu programa, varijabla "A" se koristi u jednu svrhu, a u drugom dijelu programa ista se varijabla koristi u drugu svrhu.

Transformacije tijeka programa (nepotrebno, ali namjerno) kompliciraju redoslijed izvođenja programa, koriste nepotrebne naredbe ili potprograme, višestruke nepotrebne transformacije podataka i sl.

Promjene tijeka podataka mijenjaju položaj podataka u memoriji tijekom izvođenja, načine kodiranja podataka, sastavljanja i rastavljanja podataka u veće nakupine te redoslijed podataka.

3.2 Nevidljivost (eng. stealth)

Najbolji (i jedini) način za zlonamjerni program da ne bude uništen u napadnutom sustavu jest da ne bude prepoznat, da bude "nevidljiv" (eng. stealth). U stvari, sustav vidi program u izvođenju, ali ga ne prepoznaje kao zlonamjerni.

Tu se autori virusa bore na dvije fronte: onemogućiti analizu programa te onemogućiti stvaranje "potpisa" virusa po kome će ga se moći prepoznati.

Naime, antivirusni programi na štićenom sustavu rade tako da u svakoj datoteci koja je pohranjena ili se prenosi u sustav traže potpise poznatih virusa pohranjenih u lokalnoj bazi podataka. Ta se baza stalno osvježava iz središnjice proizvođača antivirusnog programa.

Osim toga, antivirusni program promatra procese (programe u izvođenju) i njihovo ponašanje, te ih također uspoređuju s podacima u bazi.

To znači, da ako virus treba biti nevidljiv, onda svaki put kad se kopira u novi sustav treba promijeniti svoj izgled, a i način rada. Takav način ponašanja virusa se zove „promjena oblika“ (eng. shape shifting).

Prvi zadatak, onemogućiti analizu programa čak i kad analitičari imaju na raspolaganju datoteku koja sadrži programski kod virusa, autori virusa pokušavaju ispuniti kriptiranjem programskog koda.

3.2.1 Promjene oblika (shape shifting)

Ranije spominjane definicije virusa govore o nizu naredbi i aludiraju da se radi o jednom programu. Danas kad se koristi pojam "virus" zapravo se misli na "virusni skup" (eng. viral set): niz različitih programskih kodova koji imaju isto značenje (eng. semantics) tj. iz istih podataka daju iste rezultate. U toj terminologiji još se govori i o "evoluciji" kao procesu preobrazbe iz jednog pojavnog oblika (člana virusnog skupa) u drugi. Tipično, virus u jednom zaraženom sustavu ima jedan oblik, a kad se kopira u drugi sustav, evoluira i dobije novi pojavni oblik.

Tu sposobnost virusa da sam promijeni svoj oblik (eng. shape shifting) zovemo polimorfizam i metamorfizam.

3.2.1.1 Polimorfizam

Polimorfizam je tehnika kojom virus kriptira samog sebe i stvara novi, na izgled drugačiji, algoritam za dekripciju i ključ svaki put kad se kopira.

Najjednostavniji bi oblik bio da se uvijek koristi XOR funkcija, ali da se ključ mijenja svaki put (kod svake evolucije, kopije). Međutim, u stvarnosti se koriste složenije (i učinkovitije) metode.

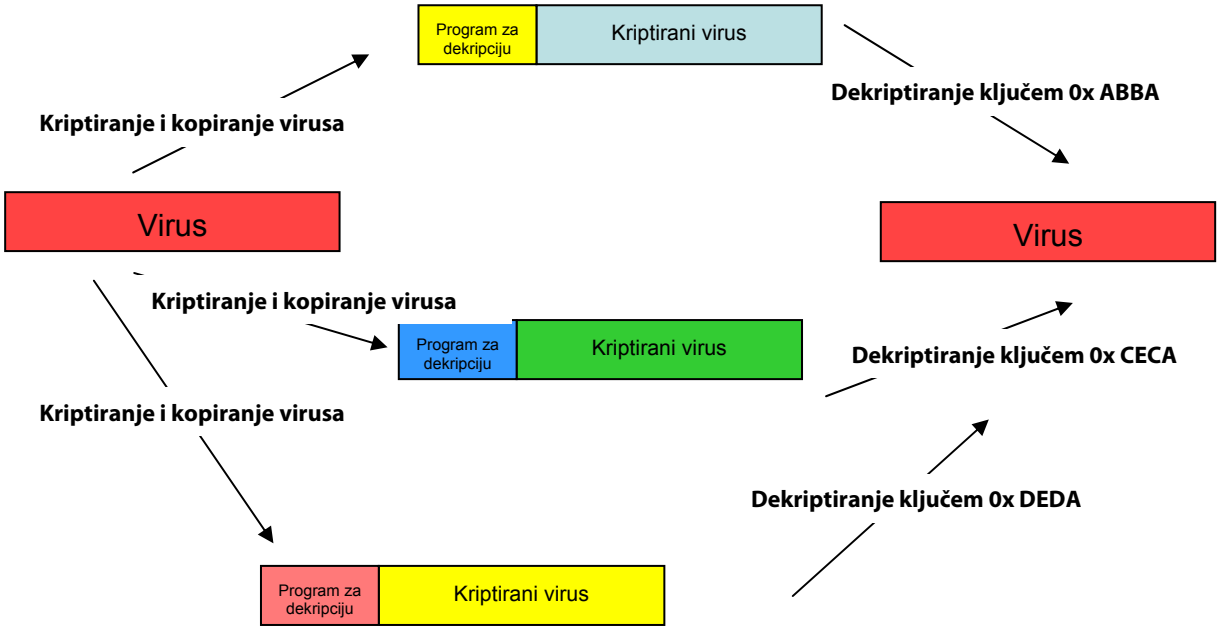
Promjena redoslijeda (eng. out-of- order) dekodiranja je postupka u kome se i program za dekodiranje (dekripciju) virusa mijenja i to tak oda se pojedine skupine operacija u novoj kopiji izvode u drugom redoslijedu nego u kopiji iz koje nastaju.

Nasumični redoslijed dekriptiranja (eng. Pseudo-random index decryption) se također može primijeniti. Naime, podaci koje treba kriptirati i dekriptirati (programski kod virusa) ne moraju se uzimati slijedno, od početka do kraja virusnog programa, već napreskokce. Taj slijed svaki put može biti drukčiji.

Različite programske naredbe (eng. Multiple code paths) mogu se koristiti da se postigne isti efekt, umjesto da se uvijek istim naredbama dolazi do željenog cilja (npr. x*2 je isto što i x+x, 4x / 2 i sl.).

Nepotrebne naredbe (eng. Junk code) mogu se ubacivati u programski kod na nasumičnim mjestima.

Nasumično korištenje procesorskih registara (eng. Register randomization) je također moguće. Uobičajeno, prevoditelj s viših jezika, nekim naredbama strojnog jezika uvijek pridjeljuju iste procesorske registre da u njima obavljaju zadane operacije. To ne mora nužno biti tako i virus može svaki puta promijeniti dodjelu registara u novoj kopiji.



Slika 1 Polimorfne inačice istog virusa

No, polimorfizam ima i slabosti. Glavna slabost leži u činjenici da je programski kod na kraju ipak uvijek isti, nakon što se dekriptira. To znači da bi antivirusni programi mogli ipak prepoznati potpis virusa, najkasnije onda kad se on nalazi u radnoj memoriji i izvodi.

3.2.1.2 Metamorfizam

Metamorfizam ide jedan korak dalje od polimorfizma. On u bitnome mijenja izgled i programskog koda glavnog dijela virusa (prije kriptiranja) kod svakog kopiranja. Tako bismo polimorfizam mogli smatrati posebnom vrstom metamorfizma koja se primjenjuje samo na dio virusa u kojem je dekripcijski algoritam.

Najjednostavniji oblici metamorfizma bi bili: ubacivanje nepotrebnog koda, zamjena registara i sl. Napredne i učinkovitije tehnike uključuju i:

- **permutiranje programskog koda**: potprograma, blokova u potprogramima, pojedinačnih naredbi čiji međusobni redoslijed nije bitan, …

- **izmjene tijeka izvođenja programa**: ubacivanje skokova i poziva potprograma, …
- **integracija koda**: dio naredbi se ubaci u drugi dio naredbi

Metamorfičke operacije su zahtjevne, pa nije neuobičajeno da čine najveći dio programskog koda virusa. "Metamorphism in practice" [6]opisuje takav kod:

- programski kod samih „virusnih“operacija se pretvori u međuoblik
- suvišne i nepotrebne naredbe se maknu
- primijene se transformacije (permutacije., promjene registara, …) na tako očišćenom kodu (zapravo, samom virusu)
- suvišne i nepotrebne naredbe se ubace
- takav kod se pretvori u izvršni
- dodaje se zaraženim datotekama.

3.2.2 Kriptiranje virusa

Vidljivo je da je velik dio programskog koda virusa posvećen njegovom prikrivanju: promjeni oblika. Drugi dio koda služni širenju virusa. No, sve to služi trećem dijelu: stvarnoj zlonamjernoj aktivnosti. Taj dio virusa se naziva „shellcode“. Naziv dolazi od toga što u velikom borju slučajeva virus ima svrhu omogućiti napadaču pristup upravljačkom sučelju operacijskog sustava (eng. shell) s namjerom da ju napadač koristi za pokretanje svojih naredbi. Iako shellcode može imati i druge zadatke, naziv se održao.

Za raspravu o polimorfizmu je važno skrenuti pažnju na to da se i na shellcode primjenjuje polimorfizam, pa naredbe koje će izvršavati radi izvršenja zlonamjerne radnje ne moraju uvijek biti iste, u istom redoslijedu i s istim podacima. Sve to dodatno otežava zadatak antivirusnih programa.

Na sreću, napraviti dobar program za polimorfizam nije jednostavno. On mora osigurati što veću slučajnost (tj. izostanak prepoznatljivih ili predvidljivih pravilnosti) između različitih kopija istog virusa. Ako to ne uspije, postojat će „čvrste točke“ koje se mogu koristiti kao potpis virusa.

Većina autora virusa nema dovoljno znanja da stvori takve programe. Osim toga, svako dodatno kompliciranje postupka unosi nove probleme za autore.

Međutim, najveća i glavna slabost je prisutnost ključa u kodu virusa. Naime, čak i ako dobijemo potpuno drugačiji program, ključa za njegovo dekodiranje mora biti pohranjen u njemu. To mogu iskoristiti ne samo analitičari, već i njihovi automatizirani programi (eng. emulator).

Stoga je slijedeći izazov pokušati naći način da programski kod virusa ne sadrži bilo ključ, bilo šifru (algoritam za dekriptiranje) ili oboje. Kad bi to bilo moguće, virus se ne bi moglo prepoznati, ali niti analizirati.

3.2.2.1 Skrivanje ključa

Autor virusa može za kriptiranje programskog koda virusa primijeniti slabu ili jaku kriptografsku metodu. Slaba bi metoda bila, npr. XOR algoritam s relativno kratkim ključem.

Takva zaštita se relativno jednostavno napada metodom iscrpljivanja (eng. exhaustive attack), tj. isprobavanje svih mogućih kombinacija ključeva dok se ne dekodira zaštićeni sadržaj.

Dakle, autor bi mogao namjerno izbrisati (ne spremiti) ključ kojim je kriptirana ta kopija virusa. Stoga, antivirusni sustavi i analitičari ne mogu koristiti ključ za prepoznavanje virusa.

No, kako će se onda virus otključati, kad nema ključa? Metodom iscrpljivanja: isprobavat će sve kombinacije dok ne uspije naći pravu.

Ova se metoda zove "slučajna dekripcija" (eng. Random Decryption Algorithm = RDA) i postoje dvije vrste: deterministička u kojoj je vrijeme za pronalaženje ključa uvijek isto (primjer je virus W32/Crypto) i nedeterministička u kojoj se potrebno vrijeme ne može predvidjeti (virusi RDA Fighter i W32/IHSix).

Naravno i borci protiv virusa mogu primijeniti istu tehniku, ali ju moraju primjenjivati na svakoj novoj kopiji virusa. Ova tehnika "kupuje" dovoljno vremena virusu da se aktivira i proširi.

Osim korištenja tehnike slabe kriptografije, autor virusa može upotrijebiti neki provjereni algoritam za kriptiranje i dovoljno dugi ključ da otkrivanje ključa postane nemoguće. To se naziva jakom kriptografskom metodom.

Druga metoda [1] skrivanja ključa je ta da se on proizvodi iz varijabli koje se mogu naći u okolišu napadnutog sustava: vrijeme, hash-evi različitih podataka na sustavu i o sustavu (naziv, e-mail adresa, IP adresa, …). Ova se metoda koristi u Bradley virusu.

3.2.2.2 Skrivanje šifre

Ako antivirusni program (ili analitičar) ima ključ virusa još mu treba šifra, tj. algoritam i programski kod za dekriptiranje.

Može li autor virusa i njega sakriti? Može još i bolje: uopće ga nema, već koristi algoritam koji se već nalazi u napadnutom sustavu. Na primjer: OpenSSL ili CryptoAPI. Programski kod za korištenje kriptografskih funkcija u CryptoAPI modulu operacijskog sustava MS-Windows bi mogao izgledati ovako:

```
int main(int argc, char *argv[])
{
   HCRYPTPROV hCryptProv;
   HCRYPTHASH hCryptHash;
   HCRYPTKEY hCryptKey;
   BYTE szPassword[] = "...";
   DWORD i, dwLength = strlen(szPassword);
   BYTE pbData[] = "...";

   CryptAcquireContext(&hCryptProv, NULL, NULL, PROV_RSA_FULL, 0);
   CryptCreateHash(hCryptProv, CALG_MD5, 0, 0, &hCryptHash);
   CryptHashData(hCryptHash, szPassword, dwLength, 0);
   CryptDeriveKey(hCryptProv, CALG_RC4, hCryptHash, CRYPT_EXPORTABLE, &hCryptKey);
   CryptEncrypt(hCryptKey, 0, TRUE, 0, pbData, &dwLength, dwLength);
}
```

Za dekriptiranje je potrebno samo zamijeniti funkciju CryptEncrypt() funkcijom CryptDecrypt(). Korištenje "vanjskih" velikih i složenih funkcija dodatno otežava posao sustavima koji trebaju analizirati ili otkriti viruse. Međutim, iako je programski kod u virusu vrlo kratak, ipak predstavlja prepoznatljiv oblik koji se može koristiti kao potpis virusa.

3.2.3 Skrivanje u legalnim kriptiranim sustavima

Tvorci legalni sustava trude se učinit ih sigurnima za korisnike. To je posebno važno za komunikacijske sustave koji povezuju mnoštvo računalnih sustava i samim time predstavljaju sigurnosni rizik. Neki od autora pribjegavaju sigurnosnim metodama poznatim kao „sigurnost u tajnovitosti“ (eng. security by obscurity). Za razliku od standardnih kriptografskih algoritama čiji su principi poznati i postoje mnogi objavljeni programski kodovi za njihovu primjenu, neki autori programskih rješenja drže algoritme za kriptiranje (šifre) tajnima, nadajući se tako povećati njihovu sigurnost. Nedostatak tog pristupa je što je osjetljiv na „proboj iznutra“, tj. na prijevaru nekog od upućenih ili slamanje nekog od brojnih (što brojniji, to više ranjivosti) sustava zaštite koji čuvaju tajnu.

Skype je dobar primjer. Njegov izvršni programski kod je zaštićen s nekoliko slojeva kriptiranja, a provjera cjelovitosti obavlja se na nekoliko mjesta unutar samog koda. Na mrežnoj razini, Skype može prodrijeti kroz svaki vatrozid tunelirajući kroz dopuštene protokole (portove). Pored toga, njegov komunikacijski protokol je potpuno tajan, nema nikakve dokumentacije o njemu. I na kraju, toliko je popularan, koristan, besplatan i pouzdan da mu korisnici potpuno vjeruju. Instaliraju ga na svoje sustave bez zadrške.

Iz svega navedenog, vidljivo je da Skype sustav predstavlja idealni „autoput“ za nevidljivo širenje virusa. Ako napadač uspije probiti Skype-ovu zaštitu, više ga nitko ne može otkriti dok se munjevito širi Skype-ovom mrežom. Doduše, Skype sustav ne nudi udaljeno izvođenje naredbi, pa napadač mora otkriti ranjivost u samoj korisničkoj Skype aplikaciji. Iako, ima i vjerojatniji način, a to je korištenje slabosti u vanjskim bibliotekama koje Skype mora koristiti, a nalaze se na svakom računalu, javne su i vjerojatnost je veća da će se pojaviti ranjivost koja se može zloupotrijebiti.

Štoviše, napadač bi mogao podmetnuti svoje poslužitelje umjesto Skypeovih središnjih autorizacijskih poslužitelja i time bi efektivno preuzeo cijelu mrežu korisnika i slobodno se njom širio [8].

3.3 Dodatne zaštite virusa

Pored opisanih zaštita, virusu je jako važno da ga se ne može analizirati u radu. Uobičajene metode analitičara uključuju korištenje debuggera i sličnih alata.

Dobro zaštićeni virusi pokušavaju otkriti rad debuggera i prisustvo antivirusnih programa te im onemogućiti rad. Ako to ne mogu, onda ili blokiraju rad cijelog sustava ili obrišu sami sebe kako bi onemogućili analizu svog rada.

U ovisnosti o okolini (operacijskom sustavu) u kojem se izvode, virusi se pokušavaju sakriti i tako što dio radne memorije u kojem se izvode učine nedostupnim drugim programima.

4 Primjeri

Odabrani su primjeri virusa koji su donijeli tehnološke novine vezane uz kriptovirologiju.

4.1 Whale virus

Whale virus se pojavio u rujnu 1990. godine i, iako sam po sebi nije bio previše opasan, označio je novu eru virusa i borbe protiv njih, donijevši nekoliko važnih novina:

- **polimorfizam**: i programski kod i proces su bili kriptirani i pojavljivali se u jednom od 30 oblika
- **nevidljivost**: presretao je prekidne rutine i skrivao se u memoriji (high .memory DOS-a)
- **oklopljenost**: programski kod se mijenjao u ovisnosti o procesoru (8088 ili 8086), intenzivno zamagljivanje koda, protu-debugger djelovanje (ako otkrije debugger, blokira tipkovnicu i obriše samog sebe)

4.2 SSH worm

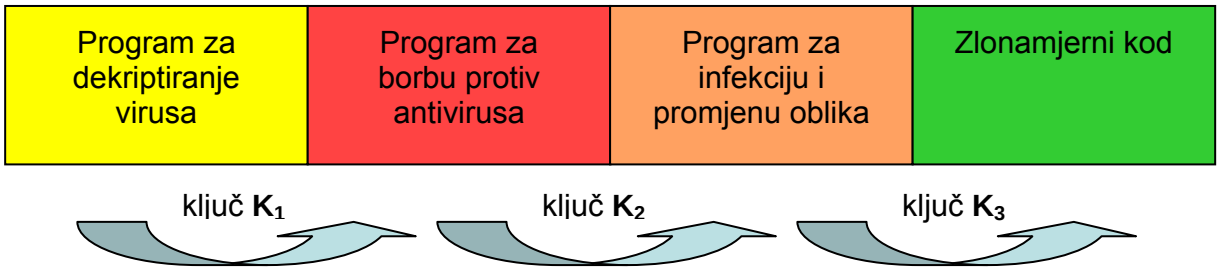
Grupa autora s MIT-a [5] istražila je 2005. ranjivosti implementacija SSH protokola i razradila je moguće protumjere za crva koji bi ih pokušao iskoristiti. SSH protokol se naširoko koristi kao sigurno sredstvo za udaljeni pristup računalima. Kako bi se lakše „prolazilo“ kroz računala za svakog korisnika postoji datoteka s popisom poznatih računala. Taj popis predstavlja potencijalnu opasnost, jer kad napadač uspije kompromitirati korisnički račun, iz popisa vidi računala na kojima će se vjerojatno uspjeti prijaviti s istim korisničkim imenom i lozinkom.

Autori su surađivali sa zajednicom koja održava SSH aplikacije i omogućili promjene u njima. Najjednostavniji je način čuvanja hash-eva imena računala umjesto samih imena, slično kao što se to radi za korisničke lozinke na operacijskom sustavu UNIX.

4.3 Bradley virus

Bradley [7] je virus kojeg se ne može analizirati. To je zapravo studija jednog takvog generičkog virusa. Opća struktura ima četiri dijela koje prikazuje Slika 2:

1. Modul za dekriptiranje koji prikuplja podatke iz okoliša iz kojih će izračunati ključ kojim će, pak, dekriptirati slijedeći modul.
2. Drugi modul sadrži programski kod koji se bori protiv antivirusnih programa.
3. Treći modul se bavi inficiranjem žrtvinih programa te promjenom oblika virusa.
4. U posljednjem modulu je zlonamjerni programski kod (eng. payload)



Slika 2 Struktura Bradley virusa

Riordan i Schneier su 1998. predložili [1] metodu „ključevi iz okoliša" (eng. *environmental keys). Pokušavajući zadovoljiti potrebu mobilnih agenata (virusa) da ključ ne bude sadržan u programu oni su predložili da se ključ generira iz podataka u okolišu ciljanog programa. To može biti lokalni okoliš, al ii globalni: podataka s neke web stranice bilo gdje u svijetu.*

Kombinacijom tehnike ključeva iz okoliša i promjene oblika, virus može ostati potpuno neprepoznat, u svim sustavima zaštite. Laboratorijska su ispitivanja pokazala visok stupanj „oklopljenosti" Bradley virusa.

5 Zaključak

Iz opisanih metoda može se zaključiti da kriptografske metode u teoriji omogućavaju stvaranje vrlo otpornih virusa koji su potencijalno nepobjedivi. Posebno zabrinjava činjenica da takvi virusi zahtijevaju ulaganje velike količine vremena za analizu što im ostavlja mogućnosti velikog širenja, promjene oblika i daljnjeg kompliciranja borbe protiv njih.

Može se zamisliti situacija u kojoj bi takav virus iskoristio vrijeme u kojem još nije savladan, da se instalira na brojna računala na takav način da zamijeni vitalan dio sistemskog softvera, na primjer interakciju s diskom i pri tome kriptira korisnikove podatke. Na taj način korisnik, čak i kada otkrije virus, ne želi ga uništiti jer jedino preko njega može doći do svojih bitnih resursa.

Glavni problem je što se, jednom pušteni virus, velikom brzinom replicira i širi, a za analizu su potrebni ljudi, kojih je malo i kojima treba relativno puno vremena.

6 Reference

[1] J. Riordan and B. Schneier: "Environmental Key Generation towards Clueless Agents", Mobile Agents and Security, G. Vigna, ed., Springer-Verlag, 1998, pp. 15-24., http://www.schneier.com/paper-clueless-agents.html

[2] Christian Collberg, Clark Thomborson, Douglas Lowbtopic: "A Taxonomy of Obfuscating Transformations"; http://www.cs.arizona.edu/~collberg/Research/Publications/CollbergThomborsonLow97a/A4.pdf

[3] Prashant Shah: "Code Obfuscation For Prevention of Malicious Reverse Engineering Attacks"; http://islab.oregonstate.edu/koc/ece478/02Report/S2.pdf

[4] Whale virus; McAfee Virus Description; http://vil.nai.com/vil/content/v_1383.htm

[5] Schechter, Jung, Stockwell, McLain: „Inoculating SSH Against Address-Harvesting Worms“; http://nms.csail.mit.edu/projects/ssh/sshworm.pdf

[6] „Metamorphism in practice“; The Mental Driller, 29A, vol. 6

[7] Éric Filiol : „Strong Cryptography Armoured Computer Viruses Forbidding Code Analysis: the Bradley virus“, Proceedings of the 14th EICAR Conference, 2005

[8] Silver Needle in the Skype
P. Biondi, F. Desclaux, Black Hat Amsterdam, 2006

[9] Frederic Raynal: „Malicious cryptography“, 2006, ; http://www.securityfocus.com/infocus/1865/1

[10] Adam Young and Moti Yung: „Malicious Cryptography: Exposing Cryptovirology“; http://www.amazon.com/Malicious-Cryptography-Cryptovirology-Adam-Young/dp/0764549758/ref=pd_bbs_sr_1?ie=UTF8&s=books&qid=1210539230&sr=8-1