pd.tvz.hr
POLYTECHNIC & DESIGN Vol. 6, No. 4, 2018.
DOI: 10.19279/TVZ.PD.2018-6-4-02

NAPREDNA STATIČKA ANALIZA ZLONAMJERNOG KODA

ADVANCED STATIC ANALYSIS OF MALICIOUS CODE

Domagoj Javorović[1], Marinko Žagar[2]

[1]Ministarstvo obrane Republike Hrvatske, Trg kralja Petra Krešimira IV br. 1, Zagreb
[2]Tehničko veleučilište u Zagrebu, Vrbik 8, Zagreb, Hrvatska

Sažetak

Sve većim razvojem i primjenom računala u industriji, gospodarstvu i svakodnevnom životu gotovo je nemoguće naći područje ljudskog djelovanja u kojem se ne koriste računala, a jednako tako primjenu nalaze i u različitim nelegalnim aktivnostima. Prema nekim izvorima [1], zlonamjerni kodovi najveći su računalni problem današnjice. Reverznim inženjeringom programskog koda potvrđuje se da li je kod zlonamjeran, kojoj vrsti pripada, kako ga otkriti i ukloniti te spriječiti širenje kako u budućnosti ne bi nanosio štetu drugim korisnicima. Ovaj rad obrađuje naprednu statičku analizu zlonamjernog koda WannaCry 2.0.

Ključne riječi: *napredna statička analiza, zlonamjerni kod, IDA Pro, asembler, WannaCry.*

Abstract

With the growing development and application of computers in the industry, economy and everyday life, it is almost impossible to find a human activity area where computers are not used and equally applied in various illegal activities. According to some sources [1], malicious codes are the biggest computer problem today. Reverse engineering of software code verifies whether it is malicious, which type of malicious code belongs, how to detect and remove it and how to prevent spreading so that it would not harm others in the future. This paper describes advanced static analysis of malicious code WannaCry 2.0.

Keywords: *advanced static analysis, malicious code, malware, IDA Pro, Assembly, WannaCry.*

1. Uvod

1. *Introduction*

2017. godine, napadi zlonamjernim kodom uzrokovali su najveći broj sigurnosnih incidenata. Usporedbom 2016. godine i 2017. godine vidljiv je porast od 54% zlonamjernih kodova za mobilne uređaje, 46% zlonamjernih kodova za ucjenu i 600% povećanje napadima koristeći se IoT uređajima. [1] Programski kod koji izvodi radnje na računalu bez korisnikovog znanja ili odobrenja, može se identificirati kao zlonamjerni kod. [2] Rad obrađuje statičku analizu kao jednu od metoda reverznog inženjeringa zlonamjernog koda. Statička analiza opisuje proces analize koda ili strukture programa kako bi odredila njegovu namjenu. Provođenjem statičke analize zlonamjerni kod se ne pokreće već se rastavlja iz strojnog jezika u jezik niske razine, što je u većini slučajeva asemblerski jezik.

U većini slučajeva, nakon sigurnosnog incidenta, dostupna je izvršna datoteka, njezini artefakti i podatci spremljeni u radnoj memoriji. Izvršna datoteka nije u čitljivom formatu, a kako bi se steklo znanje o funkcionalnosti i cilju zlonamjernog koda, analitičari koriste veliki opseg alata i tehnika, a svaki/a od njih otkriva jedan dio. Postoje dva osnovna tipa analize zlonamjernog koda:

- statička analiza
- dinamička analiza

Prije same analize zlonamjernog koda potrebno je znati koje sve mogućnosti isti može sadržavati. Dijeljenjem zlonamjernog koda u kategorije analitičar može brže i lakše otkriti o kojoj vrsti koda se radi, te koja je njegova namjena.

Svaka veća sigurnosna organizacija ili tvrtka posjeduje vlastitu kategorizaciju zlonamjernog koda sukladno njihovim potrebama, a za potrebe rada opisujemo opću podjelu zlonamjernih kodova:

- Zakulisni – engl. *Backdoor* – zlonamjerni kod koji napadaču osigurava neovlašten pristup računalu ili računalnoj mreži.
- Botnet – mreža zaraženih računala kojima upravlja napadač.
- Kod za preuzimanje – engl. *Downloader* – koristi se kako bi preuzeo neki drugi zlonamjerni kod prilikom prvog pristupa računalu ili mreži.
- Trojanski konj – zlonamjerni kod čija je namjena prikupljanje informacija o korisniku, a pritom se prikazuje korisniku kao legitimni program.
- Pokretački kod – engl. *Launcher* – program čija je svrha pokretanje drugih zlonamjernih programa, a korištenjem ne tradicionalnih tehnika pokretanja kodu koji se pokreće mogu osigurati prikrivenost i veća prava na računalu ili mreži.
- Rootkit – kod dizajniran na način da se prikrije na nižim programskim razinama. Često se veoma teško otkriva, a još teže se uklanja.
- Kod za ucjenu – engl. *Ransomware* – zlonamjerni kod za ucjene
- Scareware – nije zlonamjerni kod, već se oslanja na strah žrtve kako bi ostvario svrhu. Primjer je popularni „FBI“ Scareware koji se javlja na stranicama neprimjerenog sadržaja i naplaćuje „kaznu“ korisnicima.
- Crv – kod koji ima mogućnost kopiranja samog sebe velikim brzinama.
- Virus – kod koji kopira sam sebe i veziva se za neke datoteke, za pokretanje koda potrebna je korisnikova interakcija.

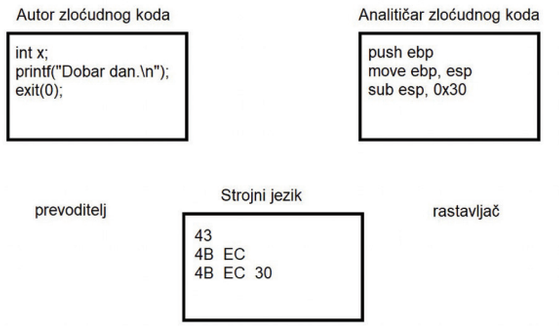
Većina zlonamjernih kodova može se svrstati u više od jedne kategorije, a primjer je zlonamjerni kod za prikupljanje aktivnosti korisnika na tipkovnici (engl. *keylogger*) koji pripada kategoriji trojanski konj, a pokreće se otvaranjem .doc dokumenta kojeg je korisnik zaprimio e-poštom (macro virus). Jedna od metoda, koju koriste iskusni analitičari, je metoda iskustvenog nagađanja, odnosno postavljanje hipoteza, te kroz analizu traženje potvrda o istima. Ova metoda višestruko ubrzava analizu koda, samim time što se analitičar fokusira samo na dijelove za koje smatra da su zlonamjerni.

Ovisno o cilju autora zlonamjernog koda, kod može biti pisan za jedno ili manji broj računala te za jako veliki opseg mogućih meta. Masovni napadi su napadi koji se relativno brzo detektiraju i uglavnom nisu opasni obzirom na to da u većini slučajeva koriste ranije objavljene eksploatacije za koje postoje sigurnosne zakrpe. Ciljani napadi su opasniji, uglavnom koriste zero-day eksploatacije i teško se otkrivaju obzirom da su izrađeni za specifičan tip računala ili računalne mreže. Sama detekcija i zaštita specifičnih sustava bez detaljne analize zlonamjernog koda je gotovo nemoguća.

2. Introduction to static analysis
2. *Uvod u statičku analizu*

Namjena osnovne statičke analize je potvrda zlonamjernosti programa te stjecanje znanja o osnovnim funkcionalnostima koda. Prilikom osnovne statičke analize ne pregledavaju se instrukcije programa već se koriste metode provjere zlonamjernog koda antivirusnim programima, korištenje sažetih vrijednosti kako bi se identificirao tip zlonamjernog koda te prikupljanje informacija kroz nizove znakova, zaglavlja i funkcija koje kod koristi. Kada je zlonamjerni kod nepoznat koristi se veći broj metoda kako bi se steklo više informacija o istome. [3]

Kako bi se u potpunosti shvatilo kako zlonamjerni kod funkcionira te kako ga identificirati potrebno je koristiti naprednu statičku analizu. Zlonamjerni kod se uglavnom razvija koristeći jezike visoke razine te se prevodi u strojni jezik koji zatim pokreće procesor računala. Napredna statička analiza koristi se suprotnom taktikom te se strojni jezik rastavlja u asemblerski kod. Slika 1. prikazuje pojednostavljeni model.



Slika 1 *Pojednostavljeni model razina koda [4]*
Figure 1 *Simplified model of code levels*

Uobičajeno se računalni sustavi opisuju kroz šest razina apstrakcije. Pisanjem koda u nižim razinama smanjuje se opseg meta napada, ali se značajno otežava pronalazak i uklanjanje zlonamjernog koda, samim time razvoj koda u jezicima nižih razina je kompleksan i zahtjeva veliku količinu resursa.

- Računalna sklopovska podrška – engl. *hardware* - jedina fizička razina koja implementira logičke operacije. Fizička obilježja otežavaju manipulaciju sklopovskom podrškom računala.
- Mikroprogram – engl. *firmware*, koriste se na strujim krugovima za koje su razvijeni.
- Strojni jezik – operacijski kod je sastavni dio, a nastaje prilikom prevođenja jezika visoke razine.
- Jezici niske razine – najniži ljudski čitljiv jezik. Najčešće korišteni jezik niske razine je asembler, ujedno kao i jezik najviše razine koji se može konzistentno i pouzdano rastaviti iz strojnog jezika. [3]
- Jezici visoke razine – korištenjem programske logike olakšavaju razvoj programa (C, C++ i ostali).
- Interpretirani jezici – jezici najviše razine, a izvode se odmah nakon čitanja koda (C#, Perl, Java i ostali).

Reverznom inženjeru uglavnom je dostupna samo izvršna datoteka, koja je napisana u strojnom jeziku. Kao što prikazuje Slika 1. strojni jezik pomoću rastavljača generiramo u asemblerski kod.

Asemblerski dijalekti se odnose na određene tipove mikroprocesora: x86, x64, PowerPC, MIPS, SPARC i ARM.

Zlonamjerni kod je najčešće pisan x86 arhitekturom. Razlog tome je što x64 arhitektura podržava programe napisane u x86, što ne vrijedi suprotno. [5]

3. Working environment and IDA disassembler

3. *Radna okolina i rastavljač koda IDA*

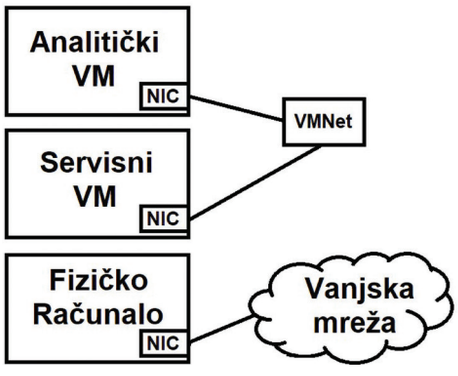
Sigurnost je najbitnija prilikom rada sa zlonamjernim kodovima. Prije početka rada potrebno je osigurati da kod koji se analizira, ne može zaraziti računala u mreži, odnosno osigurati sigurnu radnu okolinu. Prilikom analize zlonamjernih kodova uglavnom se koriste virtualni strojevi jer pružaju mogućnost uzimanja snimke stanja te ukoliko se analitičar želi vratiti u prošlo stanje, to jednostavno može učiniti jednim klikom.

Većina zlonamjernih kodova koristi računalnu mrežu kao medij propagacije, samim time je ključno da se kodu ne pruže sredstva širenja. Računalni alati za virtualizaciju pružaju mogućnost konfiguracije mrežnih adaptera koja se sastoji od:

- NAT – virtualni stoj koristi IP adresu domaćina kako bi se spojio sa drugim računalima.
- Premošćen – engl. *Bridged* – virtualnom stroju dodijeljena je vlastita IP adresa.
- Domaćin – engl *Host-only* – virtualni stroj koristi privatnu mrežu u koju su spojeni ostali virtualni strojevi i računalo domaćin.
- Privatna mreža – engl. *Internal* – virtualni stroj koristi privatnu mrežu u koju su spojeni samo ostali virtualni strojevi, bez domaćina.
- Bez mreže – engl. *None* – mrežni adapter na virtualnom stroju nije omogućen.

U većini se slučajeva na virtualni stroj podiže Windows operativni sustav, iz razloga što MS Windows drži oko 75% tržišta operativnim sustavima. [6]

Za potrebe ovog rada korišten je WMware alat za virtualizaciju te operativni sustav Windows 8.1. Laboratorijski uvjeti dizajnirani za potrebe rada uključuju korištenje mrežnog adaptera u stanju bez mreže, s obzirom da se nad uzorkom provodi samo statička analiza. U slučaju provođenja dinamičke analize zlonamjernog koda, mrežni adapter bi se konfigurirao u stanju privatne mreže kako bi omogućio potencijalnu komunikaciju zlonamjernog koda s lažno pokrenutim servisima. Slika 2. prikazuje konfiguraciju mrežnog adaptera u stanju privatne mreže.



Slika 2 *Konfiguracija mrežnog adaptera u stanju privatne mreže [4]*

Figure 2 *Internal state of network adapter*

Sukladno karakteristikama proizvođača alata Hex-Rays, IDA (engl. *Interactive Disassembler*) nije samo rastavljač već i alat za otklanjanje grešaka u programskom kodu za više tipova procesora. U ovom radu opisan je IDA rastavljač s obzirom na to da je on izbor većine sigurnosnih stručnjaka i reverznih inženjera.

Ulazni tipovi koji su podržani unutar IDA alata su PE (engl. *Portable Executable*), COFF (engl. *Common Object File Format*) i ELF (engl. *Executable and Linkable Format*). Uz rastavljanje programa IDA pruža mogućnost otkrivanja funkcija, identifikacije lokalnih varijabli i analize stoga. [4]

4. Example of disassembled malware
4. *Primjer rastavljenog zlonamjernog koda*

Analizu zlonamjernog koda vrše profesionalci i entuzijasti. Novi zlonamjerni kodovi se uglavnom prikupljaju koristeći *honeypot* sustave koji su namjerno oslabljeni kako bi privukli što veći broj zlonamjernih kodova.

Ovim tipom sustava proizvođači antivirusnih programa prikupljaju informacije o novonastalim zlonamjernim kodovima te ih rastavljaju, raspoznaju i pohranjuju sažete vrijednosti određenih funkcija i/ili cijelog koda u bazu antivirusnog programa koju klijent zanavlja.

Uz *honeypot* sustave na Internetu postoji veći broj arhiva zlonamjernih kodova, gdje se uz dobro objašnjenje, entuzijastični analitičar može registrirati i preuzeti kodove za analizu. Zlonamjerni kod obrađen u ovom radu preuzet je sa web stranice: https://virusshare.com/. Preuzeti zlonamjerni kod je „WannaCry 2.0“ sa slijedećim digitalnim potpisom:

- MD5: 84c82835a5d21bbcf75a61706d8ab549
- SHA1: 5ff465afaabcbf0150d1a3ab2c2e74f3a4426467

Kao što ovaj rad navodi, da različiti antivirusni programi koriste različite baze sažetih vrijednosti zlonamjernog koda, može se potvrditi na stranici: https://www.virustotal.com, koja omogućava analizu zlonamjernog koda na velikom broju antivirusnih alata. Analizom zlonamjernog koda utvrđeno je da 60 od 68 antivirusnih programa prepoznaje zlonamjerni kod, dok ih osam ne prepoznaje kod kao zlonamjeran, a to su: Comodo, Kingsoft, TotalDefense, Alibaba, Avast Mobile Security, eGambit, SUPERAntiSpyware i Zillya.

Rastavljanjem zlonamjernog koda utvrđeno je da zlonamjerni kod vrši provjeru *mutexa* pod nazivom MSWinZonesCacheCounterMutexA. U slučaju da *mutex* postoji program obustavlja daljnje izvršavanje. U zlonamjernom kodu ne postoji funkcija koja kreira ovu vrstu *mutexa*. Kod ispod prikazuje opisanu provjeru [4]:

```
.text:00401F08          push    offset aGlobalMswinzon
; "Global\\MsWinZonesCacheCounterMutexA"
.text:00401F0D          lea     eax, [ebp+Dest]
.text:00401F10          push    offset aSD      ; "%s%d"
.text:00401F15          push    eax             ; Dest
.text:00401F16          call    ds:sprintf
.text:00401F1C          xor     esi, esi
.text:00401F1E          add     esp, 10h
.text:00401F21          cmp     [ebp+arg_0], esi
.text:00401F24          jle     short loc_401F4C
.text:00401F26 loc_401F26:      ; CODE XREF:
```

```
sub_401EFF+4Bj
.text:00401F26          lea     eax, [ebp+Dest]
.text:00401F29          push    eax             ; lpName
.text:00401F2A          push    1               ;
bInheritHandle
.text:00401F2C          push    100000h         ;
dwDesiredAccess
.text:00401F31          call    ds:OpenMutexA
.text:00401F37          test    eax, eax
.text:00401F39          jnz     short loc_401F51
.text:00401F3B          push    3E8h            ;
dwMilliseconds
.text:00401F40          call    ds:Sleep
.text:00401F46          inc     esi
.text:00401F47          cmp     esi, [ebp+arg_0]
.text:00401F4A          jl      short loc_401F26
```

ispušta datoteke koje kasnije koristi te im mijenja atribut u „sakriveno“ i dozvoljava potpuni pristup svim datotekama koje se nalaze u trenutnom direktoriju kao i onima hijerarhijski ispod njega. Kod ispod prikazuje dio zlonamjernog koda koji opisuje navedene karakteristike. [4]

```
.text:004020DA          push    ebx             ; lpExitCode
.text:004020DB          push    ebx             ;
dwMilliseconds
.text:004020DC          push    offset CommandLine ;
"attrib +h ."
.text:004020E1          call    sub_401064
.text:004020E6          push    ebx             ; lpExitCode
.text:004020E7          push    ebx             ;
dwMilliseconds
.text:004020E8          push    offset aIcacls_GrantEv ;
"icacls . /grant Everyone:F /T /C /Q"
.text:004020ED          call    sub_401064
.text:004020F2          add     esp, 20h
```

Zlonamjerni kod koristi napredni Microsoft RSA i AES algoritam za enkripciju podataka na računalu žrtve. Važno je napomenuti da zlonamjerni kod kriptira samo određene tipove podataka i to samo one koji sadrže slijedeće ekstenzije: .docx, .text, .rsrc, .res, .der, .pfx, .key, .crt, .csr, .p12, .pem, .odt, .ott, .sxw, .stw, .uot, .3ds, .max, .3dm, .ods, .ots, .sxc, .stc, .dif, .slk, .wb2, .odp, .otp, .sxd, .std, .uop, .odg, .otg, .sxm, .mml, .lay, .lay6, .asc, .sqlite3, .sqlitedb, .sql, .accdb, .mdb, .dbf, .odb, .frm, .myd, .myi, .ibd, .mdf, .ldf, .sln, .suo, .cpp, .pas, .asm, .cmd, .bat, .ps1, .vbs, .dip, .dch, .sch, .brd, .jsp, .php, .asp, .java, .jar, .class, .mp3, .wav, .swf, .fla, .wmv, .mpg, .vob, .mpeg, .asf, .avi, .mov, .mp4, .3gp, .mkv, .3g2, .flv, .wma, .mid, .m3u, .m4u, .djvu, .svg, .psd, .nef, .tiff, .tif, .cgm, .raw, .gif, .png, .bmp, .jpg, .jpeg, .vcd, .iso, .backup, .zip, .rar, .tgz, .tar, .bak, .tbk, .bz2, .PAQ, .ARC, .aes, .gpg, .vmx, .vmdk, .vdi, .sldm, .sldx, .sti, .sxi, .602, .hwp, .snt, .onetoc2, .dwg, .pdf, .wk1, .wks, .123, .rtf, .csv, .txt, .vsdx, .vsd, .edb, .eml, .msg, .ost, .pst, .potm, .potx, .ppam, .ppsx, .ppsm, .pps, .pot, .pptm, .pptx, .ppt, .xltm, .xltx, .xlc, .xlm, .xlt, .xlw, .xlsb, .xlsm, .xlsx, .xls, .dotx, .dotm, .dot, .docm, .docb, .doc. [4]

Proučavanjem zlonamjernog koda utvrđeno je da se unutar njega nalaze tri Bitcoin računa na koja je moguće izvršiti uplatu u Bitcoin kripto valuti. Brojevi Bitcoin račna: [4]

- 115p7UMMngoj1pMvkpHijcRdfJNXj6LrLn,
- 12t9YDPgwueZ9NyMgw519p7AA8isjr6SMw,
- 13AM4VW2dhxYgXeQepoHkHSQuy6NgaEb94.

Kod naveden ispod prikazuje te tri linije:

```
.text:00401EB0          mov     [ebp+Source], offset
a13am4vw2dhxygx ;  "13AM4VW2dhxYgXeQepoHk
HSQuy6NgaEb94"
.text:00401EB7          mov     [ebp+var_8],
offset a12t9ydpgwuez9n ;
"12t9YDPgwueZ9NyMgw519p7AA8isjr6SMw"
.text:00401EBE          mov     [ebp+var_4], offset
a115p7ummngoj1p ; "115p7UMMngoj1pMvkpHijcRd
fJNXj6LrLn""13AM4VW2dhxYgXeQepoHkHSQuy6
NgaEb94"
.text:00401EB7          mov     [ebp+var_8],
offset a12t9ydpgwuez9n ;
"12t9YDPgwueZ9NyMgw519p7AA8isjr6SMw"
.text:00401EBE          mov     [ebp+var_4],
offset a115p7ummngoj1p ;
"115p7UMMngoj1pMvkpHijcRdfJNXj6LrLn"
```

Unutar koda se nalazi i funkcija 4018F9 koja služi za upravljanje CSPom[1] te se svakim pozivom te funkcije umanjuje brojač za jedan. Kada on dostigne nulu tada kontekst koji je zadržavan se otpušta, samim time se uništava ključ i/ili parovi ključeva kojim su kriptirani podatci na računalu žrtve.

1 engl. *CryptReleaseContext* – sadrži veliki dio implementacija kriptografskih standarda.

Iduća zanimljiva funkcija je 401A45 koja dohvaća adrese vanjskih funkcija iz DLL[2] datoteka. U slučaju da zloćudni kod može pristupiti svim vanjskim funkcijama tada funkcija unutar koda vraća vrijednost jedan (1). Ukoliko neka od DLL datoteka nije dostupna tada funkcija ne može pristupiti traženom sadržaju i postavlja varijablu *result* u nulu te program završava s radom bez kompromitacije žrtvinog računala. [4]

```
result = 1;
if ( !dword_40F894 )
{
  v0 = LoadLibraryA(aAdvapi32_dll_0);
  v1 = v0;
  if ( !v0 || (dword_40F894 = (int (__stdcall *)(_
DWORD, _DWORD,
  _DWORD, _DWORD, _DWORD))
GetProcAddress(v0, aCryptacquireco),
  dword_40F898 = (int (__stdcall *)(_DWORD,
_DWORD, _DWORD,
  _DWORD, _DWORD, _DWORD))
GetProcAddress(v1, aCryptimportkey),
  dword_40F89C = (int (__stdcall *)(_DWORD))
GetProcAddress(v1,
  aCryptdestroyke),dword_40F8A0 = (int)
GetProcAddress(v1,
  aCryptencrypt),dword_40F8A4 = (int (__stdcall
  *)(_DWORD, _DWORD, _DWORD, _DWORD,
_DWORD,
  _DWORD))GetProcAddress(v1, aCryptdecrypt), v2
=

GetProcAddress(v1,aCryptgenkey),
      dword_40F8A8 = (int)v2,
      !dword_40F894)
      || !dword_40F898
      || !dword_40F89C
      || !dword_40F8A0
      || !dword_40F8A4
      || !v2 )
      {
      result = 0;
      }
  }
  return result;
}
```

Datoteke koje je zlonamjerni kod ispustio su u isto vrijeme zaštićene lozinkom kod komprimiranja. Kako bi se omogućio pristup daljnjoj analizi nakon pokretanja zlonamjernog koda potrebno je doznati lozinku korištenu za komprimiranje datoteka. Naprednom statičkom analizom utvrđeno je da je korištena lozinka: WNcry@2ol7, što je vidljivo u slijedećoj naredbi: [4]

```
.text:004020C8    mov    [esp+6F4h+Str], offset Str ;
"WNcry@2ol7"
.text:004020CF    push   ebx            ; hModule
.text:004020D0    call   sub_401DAB
```

Nakon isteka zadanog vremena za plaćanje otkupnine, ukoliko žrtva nije izvršila uplatu, zlonamjerni kod pokreće izvršnu datoteku taskdl.exe koja kriptirane dokumente prepoznaje po ekstenziji .WNCRYT te ih nepovratno briše. Dio koda naveden ispod paragrafa prikazuje pretraživanje datoteka sa navedenom ekstenzijom i brisanje istih. [4]

```
.text:0040105E          push   offset aRecycle ;
"$RECYCLE"
.text:00401063          push   eax            ; Format
.text:00401064          push   offset aCS     ;
"%C:\\%s"
.text:00401069          push   esi            ; String
.text:0040106A          call   ds:swprintf
.text:00401070          add    esp, 10h
.text:004010E9          push   offset a_wncryt ;
".WNCRYT"
.text:004010EE          push   eax            ; Format
.text:004010EF          push   offset aSS     ;
"%s\\*%s"
.text:004010F4          push   ecx            ; String
```

Taskse.exe izvršna datoteka koju je zlonamjerni kod ispustio i koju pokreće služi za kreiranje zakulisnog pristupa (engl. *backdoor*) računalu žrtve, kako bi se napadaču omogućio neovlašten i nesmetan ulaz na računalo. [4]

Zbog opsežnosti zlonamjernog koda i ograničenjima veličine rada, opisane su važnije funkcije koda, uz navedeno kod još provjerava imaju li pozivani procesi pravo čitanja u određenim opsezima radne memorije, koristi pokazivače, stog, iznimke, briše sadržaj iz RAMa, kopira sadržaj RAMa, te bilježi sistemsko vrijeme u datoteke koje sam kreira. [4]

2 engl. *Dynamic-link library* – biblioteka koja sadrži funkcije koje mogu pozivati vanjski programi.

5. Conclusion

5. *Zaključak*

Velikim rastom međunarodne i industrijske špijunaže, ratovanja i sabotaže u kiber prostoru zahvaćeni su i uobičajeni korisnici računala. Cilj rada je bio pojasniti kako funkcionira zlonamjerni kod te da kroz shvaćanje rada zlonamjernog koda korisnici sami zaključe na koji način se zaštititi.

Redovnim ažuriranjem programske podrške (engl. *software*), počevši od operativnog sustava pa do web pretraživača, korisnici se štite od najvećeg broja zlonamjernih kodova, a to su oni koji koriste već objavljene propuste. Druga opcija zaštite od zlonamjernih kodova za ucjenu je provođenje redovnih sigurnosnih kopija podataka. Važno je napomenuti da su ažuriranja programske podrške i izrada sigurnosnih kopija procesi koji se moraju stalno provoditi te jedan ne isključuje drugoga. [7]

Zaštita privatnosti, integriteta i autentičnosti podataka zahtjeva brigu o svim mogućim propustima, koji osim računalnih uključuju i ljudske. Dovoljno je pronaći samo jedan sigurnosni propust i računalo žrtve postaje kompromitirano.

6. REFERENCE

6. *REFERENCES*

[1.] G. Cleary, Internet Security Threat Report Volumen 23, Symantec, 2018.

[2.] S. P. Oriyano, Certified Ethical Hacker Version 9, Sybex, 2016.

[3.] M. Sikorski i A. Honig, Practical Malware Analysis, No Starch Press, 2012.

[4.] D. Javorović, Reverzni inženjering zlonamjernog koda za ucjene, Veleučilište Velika Gorica, 2018.

[5.] M. H. Ligh, S. Adair, B. Hartstein i M. Richard, Malware Analyst's Cookbook and DVD, Wiley Publishing, 2011.

[6.] »W3schools,« 2018. [Mrežno]. Available: https://www.w3schools.com/browsers/browsers_os.asp. [Pokušaj pristupa 20 09 2018].

[7.] M. Žagar i M. Rak, Istraživanje ransomware napada i prijedlozi za bolju zaštitu, Tehničko Veleučilište u Zagrebu, 2016.

AUTORI · *AUTHORS*

Domagoj Javorović
Rođen je u Zagrebu 1993. godine. Preddiplomski studij računarstva na Tehničkom Veleučilištu u Zagrebu završio je 2015. godine, a diplomski studij infromacijskih sustava na Veleučilištu Velika Gorica završava 2018. godine. Zaposlen je u Ministarstvu Obrane Republike Hrvatske. 2015. godine sudjelovao je na konferenciji "Central European Forum on Military Education" u Varšavi, sa temom "Data Security". Područja interesa su mu računalna sigurnost i reverzni inženjering zlonamjernog koda.

Korespodencija
javorovic.domagoj@gmail.com

Marinko Žagar - nepromjenjena biografija nalazi se u časopisu Polytechnic & Design Vol. 4, No. 1, 2016.

Korespodencija
marinko.zagar@tvz.hr