Seznamte se – Morfismy (oligomorfismus, polymorfismus, metamorfismus)

Vložil/a RubberDuck [1], 7 Únor, 2013 - 17:23

- Virus & Worms [2]

„Nový malware XYZ využívá velmi pokročilý polymorfní engine. Antivirová společnost ABC je schopna ho detekovat s přesností 20%.“ Pojmy jako polymorfismus a metamorfismus se staly denním chlebem všech IT specialistů, bezpečnostních konzultantů i řadových programátorů. Máme polymorfní viry, polymorfní exploity, polymorfní packery (Občas mám pocit, že i komentáře ve fórech a diskuzích jsou polymorfní :)) Ale jen málokdo dokáže skutečně vysvětlit rozdíl mezi jednotlivými morfismy.

V době, kdy antiviry postupně stále efektivněji detekovaly jednotlivé viry, přešli tvůrci virů do protiútoku. Základní myšlenka byla položena následovně: Pokud má vir konstantní tělo (tělo je viru je za všech okolností neměnné) a antiviry jej dokáží detekovat, existuje způsob jak tuto konstantnost rozbít? Odpověď je velice jednoduchá: Pokud něco zašifruju/zakóduju, změní se celý vzhled kódu. Pokud budu v každé iteraci infekce vždy šifrovat/kódovat vir novým způsobem, pravděpodobnost úspěšné detekce se sníží.

Oligomorfismus

První na scéně se objevuje tzv. oligomorfismus. Autoři virů jednoduše do svých výtvorů přidali seznam dekryptorů a rutinu/funkci, která si vždy jeden dekryptor vybrala a v dané iteraci podle něj tělo viru zašifrovala/zakódovala a na začátek připojila dekryptor. Standardně se setkáváme nejčastěji s instrukcemi součtu, rozdílu, dělení (kvůli zbytku po dělení) a notoricky známý XOR. Na autory tato technika nekladla nijak závratné nároky. Stačilo pouze vědět, jak označit začátek a konec těla viru pro potřeby dekryptování a jak provést proces samotného dekryptování.

Proces vytvoření nového kódu je přibližně následující: Vir dekóduje svoje tělo a uloží si ho v dekódované podobě do paměti. Následně se pokusí najít nový hostitelský soubor. Pokud ho nalezne, vybere si ze seznamu dekryptorů jeden a podle něj zakóduje dekódované tělo viru. Na začátek přidá dekryptor a infikuje s ním hostitele. Celý proces opakuje, dokud jsou splněny podmínky infekční rutiny/funkce.

Nevýhodou této techniky je skutečnost, že virus s sebou nese předem nadefinovaný počet dekryptorů, tedy existuje jen takový počet variant daného viru, jaký je počet dekryptorů. Nehledě na fakt, že právě podle dekryptoru je možné oligomorfní vir detekovat.

Příkladem oligomorfního viru může být Win95/Memorial[1]. Je schopný vygenerovat až 96 různých dekryptorů.

Polymorfismus

Autoři virů velmi rychle pochopili, že tudy cesta vede, ale jen za předpokladu, budou-li schopni vytvořit kód, který bude dekryptor generovat v plném rozsahu zcela sám a nezávisle na okolnostech. To už byl podstatně náročnější úkol. Předpokládal schopnost vytvářet a skládat instrukce assembleru tak, aby byl výsledkem funkční kód a zároveň nenarušil konzistenci vykonávání kódu nového viru. Prakticky se dá říct, že rutina funguje jako velmi zjednodušený kompilátor assebleru. Právě potenciál generovat obrovské množství různých dekryptorů, které v každé generaci zcela mění tělo viru, dal této technice název polymorfismus (poly – mnoho, více; morfismus – tvar). Kvalita polymorfního

generátoru se posuzuje podle počtu dekryptorů, které je schopný vygenerovat. Čím vyšší číslo, tím potenciálně lepší generátor (roli hrají i další aspekty, ale ty pro tentokrát vypustím).

Příkladem polymorfního enginu budiž stařičký EXPO[2] engine od Z0MBiEho.

Metamorfismus

Některým autorům virů však na mysli spočinula mnohem zajímavější technika. Technika, která by odbourala potřebu tvořit dekryptor a měnila by přímo tělo viru. To byl nesrovnatelně náročnější úkol, se kterým se nejlépe popasoval notoricky známý ruský autor virů Z0MBiE nebo bulharský virus writer Dark Avenger. Ti byli rovněž první, kdo položili základ pravidlům potřebných při psaní metamorfních enginů. Autor sice může pouze měnit použité registry. Ale tím lze docílit jen velmi malého množství změn. Daleko efektivnější je prohazovat nebo zcela měnit celé instrukce. Ale ty mohou byt na sebe vzajemne vazane. Z předchozího by mělo být jasné, že metamorfní enginy vlastně vytváří celé tělo viru zcela znovu, takže se prakticky jedná o disassembler, code rebuilder a assembler v jednom.

Jednoduše řečeno (a pravděpodobně kradu něčí definici): Metamorfismus je polymorfismus virového kódu.

Jako příklad uvedu Win9x/Regswap[3] od Vecny.

Shrnutí

Jak je vidno, jsou všechny tři techniky v základu velice podobné (všechny se snaží měnit tělo viru). Oligomorfismus můžeme chápat jako podskupinu polymorfismu. A metamorfismus jako extrémní případ polymorfismu (nepotřebujeme dekryptor, jednoduše změníme rovnou celé tělo).

Oligomorfní viry nejsou zase až tak populární. Víceméně se k nim uchylují začátečníci při pokusech o vytvoření polymorfního kódu, kdy si ověřují funkčnost dekryptování a kryptování.

Polymorfní kódy jsou v současnosti velkým problémem. Do budoucna lze očekávat více propracovaných enginů, které nebude tak snadné detekovat.

Metamorfní kódy jsou v současnosti stále spíše vzácností (uživatel se spíš setká s public variantou metamorfního enginu než se soukromým výtvorem), ale v budoucnu lze očekávat velmi rychlý a rozsáhlý nárůst metamorfního malware. Pokud bude s počtem enginů růst i jejich kvalita, můžeme očekávat skutečně černé dny v boji s malwarem.

Odkazy:

[1] - Win95/Memorial [3]
[2] - Zdrojové kódy EXPO [4]
[3] - Win9x/RegSwap [5]

URL článku:
https://security-portal.cz/clanky/seznamte-se-%E2%80%93-morfismy-oligomorfismus-polymorfismus-metamorfismus

Odkazy:
[1] https://security-portal.cz/users/rubberduck
[2] https://security-portal.cz/category/tagy/virus-worms
[3] http://www.informit.com/articles/article.aspx?p=366890&seqNum=4
[4] http://www.hackerzvoice.net/madchat/vxdevl/vxmags/mtx1/Tools/EXPO.ZIP
[5] http://www.securelist.com/en/descriptions/old20025