Нова активність Zebrocy

Дослідження ESET
Вересень 2019 р.

Вступ

Хоча літо, зазвичай, є порою відпусток, група Sednit не відпочиває, а розробляє нові компоненти для доповнення сімейства шкідливих програм Zebrocy.

Група Sednit (APT28, Fancy Bear, Sofacy, STRONTIUM) відома щонайменше з 2004 року і за останні роки часто потрапляє в заголовки новин.

20 серпня 2019 року кіберзлочинці Sednit запустили нову кампанію, спрямовану на посольства та міністерства закордонних справ у країнах Східної Європи та Центральної Азії.

Ця кампанія розпочалася з фішингового електронного листа зі шкідливим вкладенням, що запускає довгу послідовність завантажувачів, яка закінчується бекдором. Приклад такого електронного листа було завантажено в VirusTotal 22 серпня через два дні після отримання повідомлення. Огляд вектору атаки нещодавно був опублікований Telsy TRT.

Однак є ще деякі фрагменти цієї головоломки, які могли б допомогти створити більш повну картину кампанії.

Як і передбачалося іншими дослідниками, група Sednit використала мову програмування Nim в своєму наборі інструментів, а точніше у завантажувачі. Також кіберзлочинці Sednit вдосконалили Golang-завантажувач та переписали бекдор з Delphi в Golang.

Складне інфікування

На рисунку 1 зображено кроки, що призводять до інфікування жертви – від шкідливого електронного листа, який потрапляв у вхідні повідомлення, до бекдора, розгорнутого на пристроях цікавих для операторів цілей.

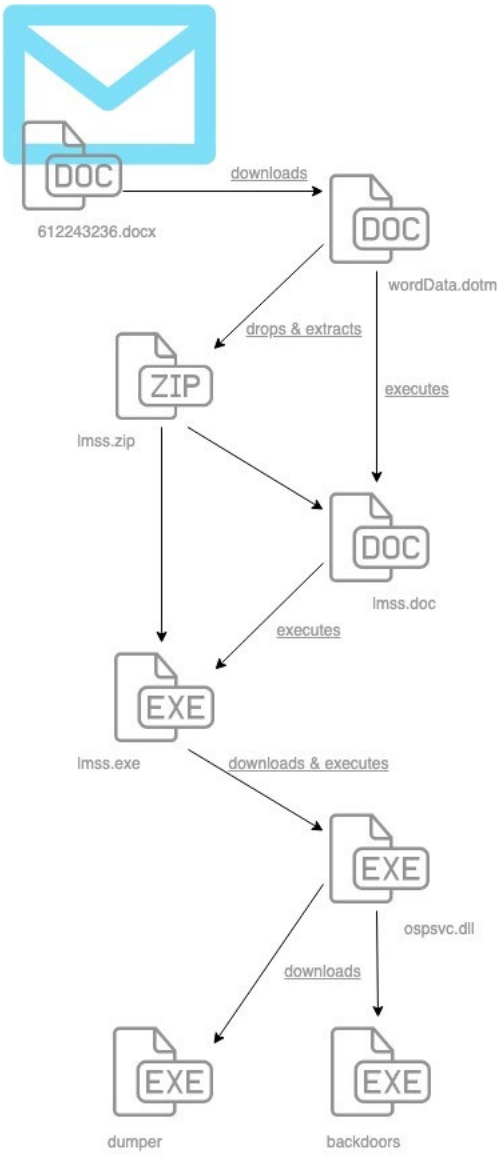

Рис. 1. Послідовність інфікування

Якщо жертва стає ціллю компонентів Zebrocy, процес інфікування, як правило, досить активний. Оскільки на комп’ютер жертви завантажується щонайменше шість шкідливих компонентів перед виконанням остаточного компонента. Така діяльність може бути зафіксована продуктом із безпеки.

Документ, прикріплений до фішингового електронного листа, пустий, але посилається на віддалений шаблон `wordData.dotm`, розміщений на Dropbox. Відкриття цього документа в Word спричиняє завантаження `wordData.dotm`, як показано на рисунку 2, та поєднання його з робочим середовищем пов'язаного документа, включно з будь-яким активним вмістом, який може містити шаблон.

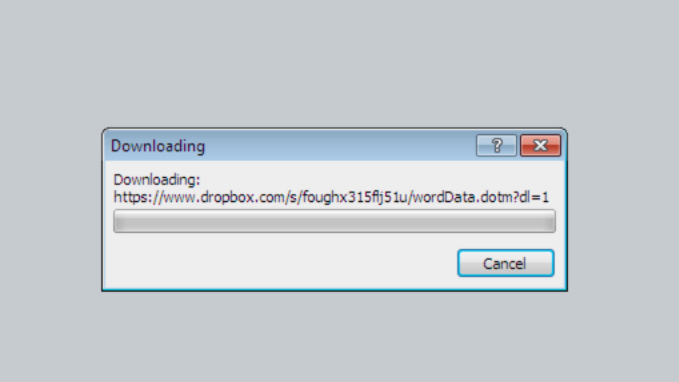

Рис. 2. Пустий Word-документ, що завантажує віддалений шаблон

Файл `wordData.dotm` містить шкідливі макроси, які потім виконуються[1]. Він також містить вбудований ZIP-архів, який завантажує та витягує макроси.

Як показано на рисунку 1, макроси в `wordData.dotm` відкривають ще один документ (`lmss.doc`, який був розпакований з архіву, витягнутого з `wordData.dotm`). Макроси в `lmss.doc` виконують `lmss.exe` (новий Nim-завантажувач Zebrocy, також витягнутий з архіву, вбудованого в `wordData.dotm`) замість `wordData.dotm`, що виконує завантажувач безпосередньо.

Однак важливо зауважити, `lmss.doc`, що містить VBA код для виконання нового завантажувача Nim, також вбудовує виконуваний файл, закодований в base64. Відповідно до документа DocumentProperties, `lmss.doc` був створений у січні 2019 року та змінений 20 серпня, за кілька годин до початку кампанії.

```
<dcterms:created xsi:type="dcterms:W3CDTF">2019-01-16T11:40:00Z</dcterms:created>
<dcterms:modified xsi:type="dcterms:W3CDTF">2019-08-20T06:20:00Z</dcterms:modified>
```

Рис. 3. Дати створення та останньої модифікації lmss.doc

Виконуваний файл, вбудований у `lmss.doc`, є завантажувачем AutoIt (SHA-1: `6b300486d17d07a02365d32b673cd6638bd384f3`), який використовувався в минулому в кампанії, проведеный приблизно в час створення `lmss.doc`. У цьому випадку завантажувач AutoIt не використовується і не має іншої мети, крім збільшення розміру документа. Оператор, ймовірно, забув видалити попередній вбудований завантажувач – це не перша помилка операторів Sednit.

[1] Залежно від версії Microsoft Word VBA макроси вимкнені за замовчуванням, і для їх активації потрібні дії користувача.

Завантажувачі

Оператори Sednit використовували декілька завантажувачів, написаних різними мовами. У цій кампанії зловмисники використовували нещодавнє розширення списку — завантажувач, написаний мовою Nim, яка є відносно новою. Це прямий «download-and-execute» бінарний файл з додаванням двох невеликих деталей. Перша, ймовірно, використовується як метод обходу пісочниці та перевіряє, чи не змінилася перша літера виконуваного файлу (буква `l` тут або `0x6C` у шістнадцятковій формі).

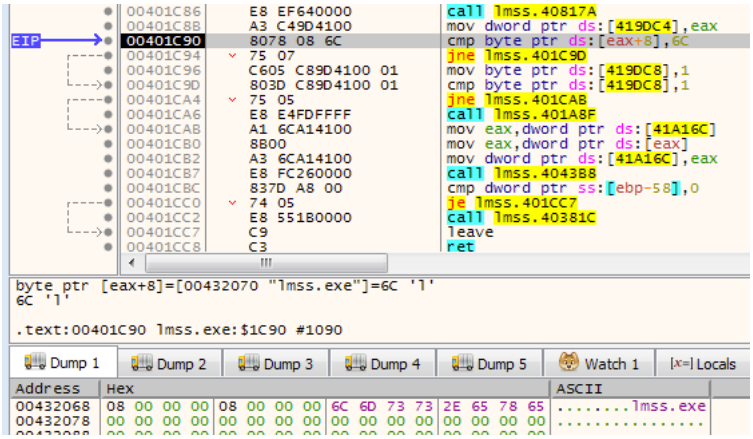

Рис. 4. Перевірка назви файлу

Друга деталь — це своєрідна обфускація, під час якої оператор замінює літери-заповнювачі в рядку правильними при визначених зміщеннях. Як показано на рисунку 5, завантажувач реконструює правильний рядок URL-адреси для завантаження за допомогою цього методу, щоб уникнути виявлення основними інструментами статичного аналізу, які в іншому випадку могли б знайти рядок URL-адреси.

```
cnc = copyString(&::cnc);
cnc[9] = 't';
cnc[0xA] = 't';
cnc[0x12] = '.';
cnc[0x16] = '.';
v4 = cnc;
cnc[0x1D] = '/';
cnc[0x26] = '-';
cnc[0x31] = '/';
cnc[0x36] = 'p';
cnc[0x37] = 'h';
path = copyString(&::path);
filename = copyString(&filname);
v6 = 0;
v7 = filename;
*(filename + 9) = 's';
*(filename + 0xC) = 'v';
*(filename + 0xF) = 'd';
if ( path )
  v6 = *path;
v8 = rawNewString(*filename + v6 + 1);
v9 = v8;
v10 = appendString(v8, path);
v11 = appendString(v10, &separator);
appendString(v11, v7);
noscreateDir(path);
if ( nosexistsFile(v9) )
{
  v13 = nosgetFileSize(v9);
  patha = HIDWORD(v13);
  v12 = v13;
}
else
{
  v12 = 0;
  patha = 0;
}
v22 = 0;
```

Рис. 5. Hex-Ray виведення рядків деобфускації

Наприклад, рядок `o-ps-c..ll` «виправлено» трьома зміщеннями відповідно `s`, `v` та `d`, щоб отримати `ospsvc.dll`. У випадку URL-адреси, оскільки в програмі завантажувача рядок починається на `h@@p://`, інструменти, які шукають `http://`, його не виявлять.

Nim-завантажувач вилучає компонент DLL під назвою `ospsvc.dll` в `C:\ProgramData\Java\Oracle\` та виконує його як сервіс через `regsvr32 /s`.

`ospsvc.dll` — це завантажувач, написаний мовою Golang, який відрізняється від попередніх завантажувачів Sednit.

Попередні Golang-завантажувачі Sednit були детально описані іншими дослідниками [1][2][3], ймовірно, що кіберзлочинці переписали свій попередній завантажувач Delphi мовою Golang. Попередні завантажувачі збирають багато інформації про комп'ютер жертви та надсилають її на свій командний (C&C) сервер. Однак новий завантажувач досить легкий з точки зору можливостей збору даних, як описано нижче.

Його функція `main_init()` містить бібліотеки, які ініціалізуються і не потребують додаткових пояснень через їх імена (більше інформації – в статті).

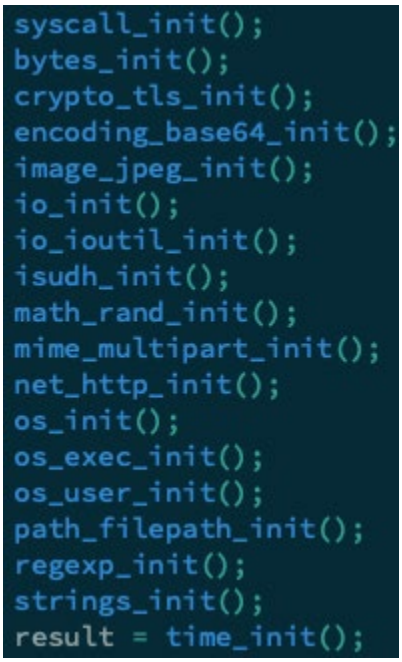

Рис. 6. Hex-Ray виведення ініціалізованих функцій у main_init (), використовуючи плагін IDAGolangHelper

Оскільки DLL запускається як сервіс через завантажувач Nim, потрібно звернути увагу на `main_DllRegisterServer()` замість `main_main()`. Рядки та ключ об'єднані, і їх можна розшифрувати за допомогою простого циклу XOR. Таке шифрування є досить ефективним у випадку з інструментами, які витягують рядки, об'єднані статично з двійкових файлів.

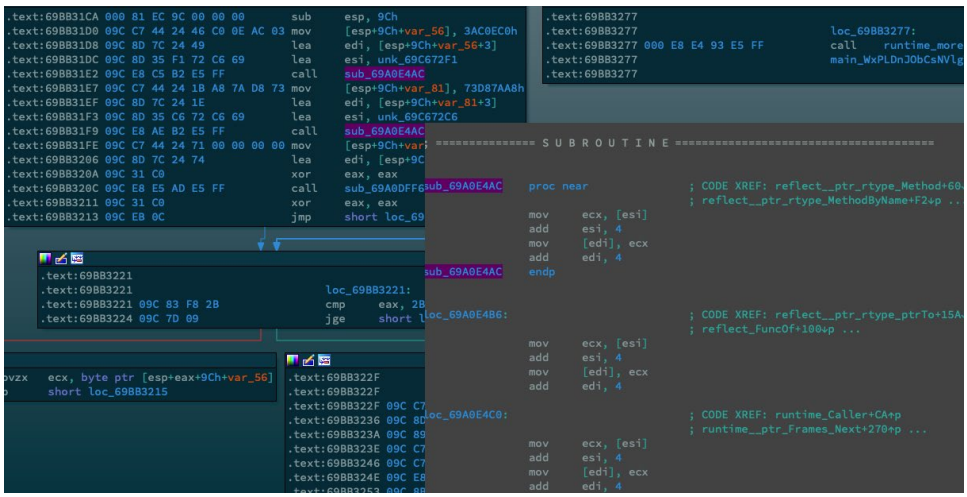

Рис. 7. Виведення IDA Pro із об'єднаних зашифрованих рядків

Крім завантаження шкідливого програмного забезпечення наступного етапу, основні функції цієї загрози — створення знімків екрана робочого столу жертви та виконання команд, отриманих від командного сервера (C&C).

Знімки екрана створюються кожні 35 секунд протягом перших хвилин виконання цього завантажувача, потім вони надсилаються на командний сервер у закодованій формі base64. Ім'я хоста та значення `%USERPROFILE%` також надсилаються на командний сервер (C&C), закодований у base64. Відповідь від командного сервера також пряма: це об'єднання рядків, закодованих у base64 та розділених знаком «|».

```
<spaces>|<cmd to execute>|<name of the binary to drop>|<binary to drop>
```

Відповідно до даних телеметрії ESET, цей завантажувач використовувався для виконання трьох різних частин шкідливого програмного забезпечення. Перша — це дампер, який спеціалісти ESET описали в попередньому дослідженні Zebrocy. Друга — це звичайний бекдор Delphi, який запускається як сервіс за допомогою тієї самої команди, що використовується завантажувачем Nim. Третя частина шкідливого програмного забезпечення — це новий бекдор, завантажений та виконаний на машині жертви.

Новий бекдор

Аналіз

Новий бекдор Zebrocy написаний мовою Golang, а не Delphi. Цей бекдор було зафіксовано вперше, але він має багато подібного з тим, який був написаний на Delphi.

Переглянувши ще раз код ініціалізації бібліотеки `main_init()` (рис. 9), спеціалісти ESET виявили нові записи. Алгоритм AES, шістнадцяткове кодування та можливості створення знімків екрану — це основні елементи, які були додані.

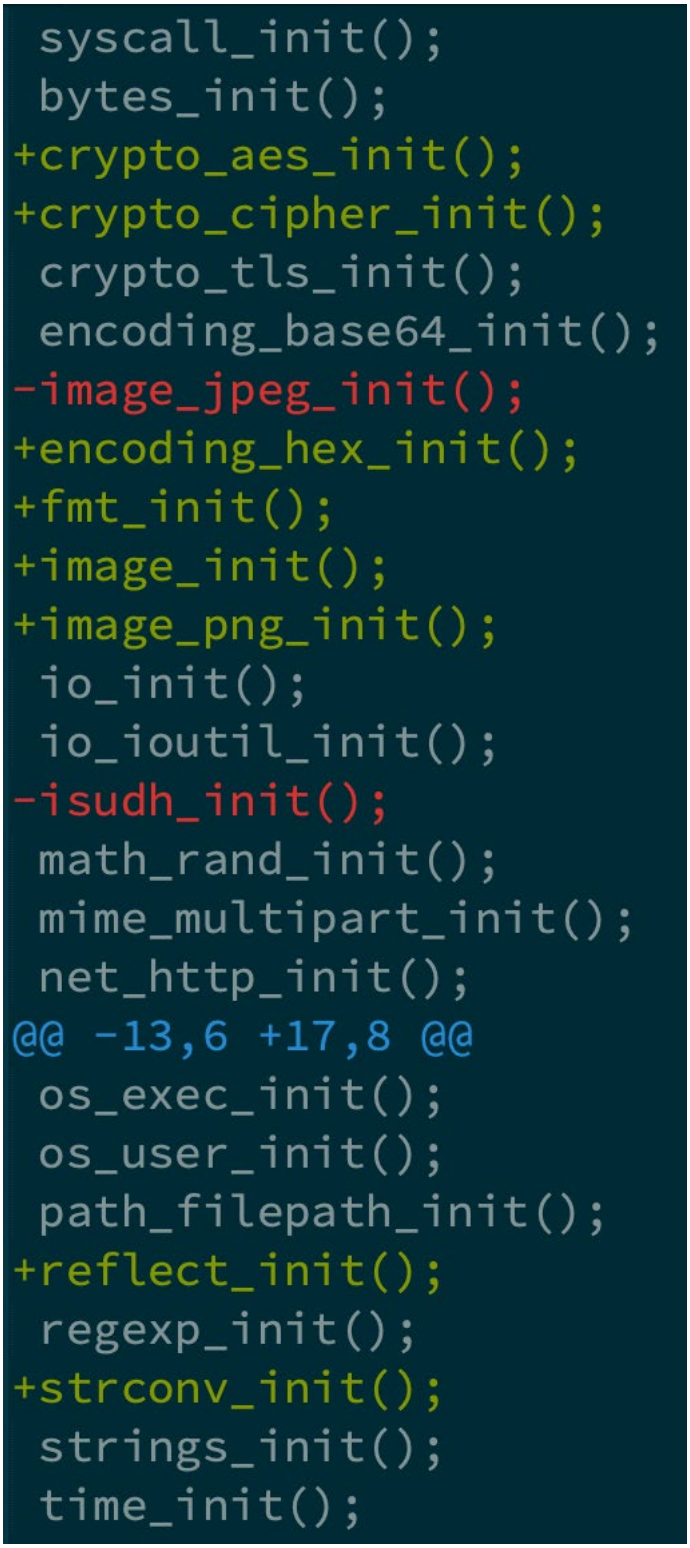

Рис. 8. Різниця між функціями бекдора та завантажувача, ініціалізованими в main_init ()

Варто зазначити, що `image_png_init` замінює `image_jpeg_init` для створення знімків екрану. Зображення у форматі JPG, зазвичай, мають менші розміри, ніж у форматі PNG.

Бекдор починається з аргументу, який є шістнадцятковим кодом. Усі, крім останнього шестибайтового фрагмента цього рядка, зашифровані XOR-ключем, що зберігається в останніх шести байтах рядка. Наступний фрагмент Python описує логіку дешифрування.

```
key = arg[-6:].decode('hex')
enc = arg[:-6].decode('hex')
''.join(chr(ord(i) ^ ord(j)) for i, j in zip(itertools.cycle(key), enc))
```

Це адреса командного сервера, яка пізніше шифрується та зберігається на диску. Це шифрування здійснюється за допомогою алгоритму AES-128 ECB з ключем, згенерованим з імені хоста. Тому відсутня можливість досягти цього командного сервера, просто подивившись на бінарний файл. Крім цього, відсутня стійкість, визначена завантажувачами, як це було раніше, а також бекдор не має будь-якого механізму стійкості.

Цей новий бекдор має різні можливості, які раніше спостерігалися в Delphi бекдорі Zebrocy:

- дії з файлами, такі як створення, модифікація та видалення
- можливість створення знімків екрану
- перелік зовнішніх накопичувачів
- виконання команд (через `cmd.exe`)
- планування завдання з назвою `Windows\Software\OSDebug` (яку оператори можуть використовувати для забезпечення стійкості вручну)

Як і в бекдор Delphi, новий бекдор має дуже обмежений набір команд, але можливість виконання довільних команд через cmd.exe розширює здатність залишатися непоміченим в системі, а також дозволяє збирати інформацію. Крім цього, виявлено ще одна подібність між бекдорами — тризначний номер версії (у форматі x.y.z); поточна основна версія — 4.y.z.

Мережа

Мережевий протокол має деяку схожість з версією Delphi бекдора. Перша взаємодія з командним сервером (C&C) шукає 32-бітний ключ AES для шифрування майбутніх з'єднань. Перехоплення даних першого запиту має такий вигляд:

```
POST [REDACTED URI] HTTP/1.1
Host: [REDACTED IP]
User-Agent: Go-http-client/1.1
Content-Length: 297
```

```
Content-Type:                                   multipart/form-data;
boundary=b116f1e0a94eff1bb406531e74bb0feba65687cf90ec8a64fc409f2
30fbd
Accept-Encoding: gzip

--b116f1e0a94eff1bb406531e74bb0feba65687cf90ec8a64fc409f230fbd
Content-Disposition:          form-data;          name="filename";
filename="[REDACTED]"
Content-Type: application/octet-stream

1
--b116f1e0a94eff1bb406531e74bb0feba65687cf90ec8a64fc409f230fbd--
```

Ті, кому відома група кіберзлочинців Sednit, можуть подумати, що ключові слова `Content-Disposition` та `boundary` виглядають знайомо. Раніше вони використовувалися бекдором Delphi у його мережевому протоколі; він також використовує алгоритм AES для шифрування фальшивого тіла (вміст після даних `Content-Type`). Варто зазначити, що навіть якщо `Content-Disposition` та другий екземпляр `Content-Type` є дійсними заголовками HTTP, тут вони використовуються всередині HTTP-повідомлення. Поле `boundary` випадкове для кожного обміну, і поле імені файлу всередині фальшивого заголовку `Content-Disposition` можна розшифрувати за допомогою наступного фрагмента Python:

```
len_filename = len(filename)
len_key = 14
xor_key = filename[-len_key:].decode('hex')
filename = filename[:len_filename-len_key].decode('hex')
val_filename = ''.join(chr(ord(i)^ord(j)) for i,j in
zip(itertools.cycle(xor_key),filename))
random_int = val_filename[-4:]
```

що приводить до наступного рядка:

```
757365722D5043181620201908211510552O7C.inc
```

Цей рядок можна зрозуміти таким чином:

```
Username: 757365722D5043
SID*: 181620
Date: 20190821151055
Random: 207C.inc
```

* 6 bytes comes from the current user's Security Identifiers (SID) `S-1-5-21-xxxxxxxxx-yyyyyyyyyy-zzzzzzzzzz-1000`

Подальші взаємодії з командним сервером відповідають цій схемі, за винятком фальшивого тіла у наведеному вище прикладі, яке замінюється результатом команди, за вимогою командного сервера (C&C). Повне тіло повідомлення також шифрується, використовуючи той самий алгоритм AES, який застосовувався з ключем під час першого обміну.

Висновки

Нові завантажувачі, новий бекдор свідчить про те, що група кіберзлочинців Sednit залишається активною, постійно оновлюючи власні компоненти. Зокрема Sednit переносить оригінальний код або повторно застосовує його іншими мовами з метою уникнення виявлення. Це, мабуть, найлегший спосіб, який не потребує змін TTP повністю. Початковий вектор компрометації залишається незмінним, проте використання служби Dropbox для завантаження віддаленого шаблону є незвичним для цієї групи кіберзлочинців.

У зв'язку з поширенням загрози спеціалісти ESET рекомендують користувачам бути уважними під час відкриття підозрілих вкладень електронної пошти.

Крім цього, дослідники ESET продовжують спостерігати за активністю групу Sednit та публікуватимуть матеріали у разі виявлення нової інформації про подальші атаки.

Індикатори компрометації (IoC)

Хеші (SHA-1)	Назви файлів	Назви виявлень ESET
c613fcccb380f7e3ce157c4f620efca503c1bad3	- (eml file)	DOC/TrojanDownloader.Agent.AMY
6f281b30d8d6a9bc1dbe2fe73995aac382c4a543	612243236.docx	DOC/TrojanDownloader.Agent.AMY
f3f945fb22916f82cb7407cde2a80a68cd83b074	wordData.dotm	VBA/TrojanDropper.Agent.AIP
a56af5b44624e8ada60057fd7f39af5b3de10724	lmss.zip	Win32/TrojanDownloader.Sednit.BK
b8ac400e1deb6e90fa4e2adb150c511c98bafc6e	lmss.doc	VBA/TrojanDropper.Agent.AIQ

f0793e02180f3ccf48e41bd67ec1161d93f07e01	lmss.exe	Win32/TrojanDownloader.Sednit.BK
04303024ff453f918925d7160abbd199f137a442	ospsvc.dll	Win32/Sednit.DI
c96db85ece2b57a9e82ba36b5f31ca9d2051a6f0	osppsvc.exe	Win32/Sednit.DJ

Мережа

https://www.dropbox[.]com/s/foughx315flj51u/wordData.dotm?dl=1

185.221.202[.]35

Методи MITRE ATT&CK

Тактика	ID	Назва	Опис
InitialAccess	T1193	Spearphishing Attachment	Zebrocy використовує фішингові листи з вкладеннями як метод компрометації.
Execution	T1059	Command-Line Interface	Бекдор Golang використовує cmd.exe для виконання команд.
	T1117	Regsvr32	Завантажувач Nim використовує regsvr32.exe для запуску завантажувача Golang.
	T1053	Scheduled Task	Бекдор Golang може створити заздалегідь заплановане завдання.
	T1064	Scripting	Віддалений шаблон містить VBA, який використовується для виконання шкідливого програмного забезпечення наступного етапу.
	T1204	User Execution	Zebrocy намагається змусити користувачів натискати на прикріплені документи Microsoft Office, що містять шкідливі макроскрипти.
Persistence	T1053	Scheduled Task	Бекдор Golang може створити заздалегідь заплановане завдання.
PrivilegeEscalation	T1053	Scheduled Task	Бекдор Golang може створити заздалегідь заплановане завдання.

DefenseEvasion	T1107	File Deletion	Бекдор Golang може видаляти файли.
	T1117	Regsvr32	Завантажувач Nim використовує regsvr32.exe для запуску завантажувача Golang.
	T1064	Scripting	Віддалений шаблон містить VBA, який використовується для виконання шкідливого програмного забезпечення наступного етапу.
Discovery	T1083	File and Directory Discovery	Бекдор Golang може створювати перелік дисків.
Collection	T1113	Screen Capture	HTTP використовується для з'єднання з командним сервером (C&C).
CommandAndControl	T1043	Commonly Used Port	Всі компоненти використовують порт 80 для з'єднання з командним сервером (C&C).
	T1024	Custom Cryptographic Protocol	Бекдор Golang використовує цикл XOR для своїх повідомлень.
	T1132	Data Encoding	Бекдор Golang кодує дані перед їх шифруванням base64.
	T1071	Standard Application Layer Protocol	HTTP використовується для з'єднання з командним сервером (C&C).
	T1032	Standard Cryptographic Protocol	Бекдор Golang шифрує з'єднання з C&C за допомогою AES ECB.
Exfiltration	T1022	Data Encrypted	Бекдор Golang шифрує дані за допомогою AES ECB, перш ніж надсилати їх через C&C.
	T1041	Exfiltration Over Command and Control Channel	Бекдор Golang розшифровує дані на своєму командному сервері (C&C).

Посилання:

[1] https://unit42.paloaltonetworks.com/sofacy-creates-new-go-variant-of-zebrocy-tool/

[2] https://securelist.com/a-zebrocy-go-downloader/89419/

[3] https://www.vkremez.com/2018/12/lets-learn-dissecting-apt28sofacy.html

---

Дослідження подано в перекладі. Оригінал доступний за посиланням.