# Статья Black Basta и неприметная доставка



**Введение**

По данным Check Point в конце первого полугодия 2022 года, 1 из 40 организаций во всем мире пострадала от атак программ-вымогателей, что представляет собой тревожное увеличение на 59% по сравнению с прошлым годом. Бизнес программ-вымогателей продолжает расти в гигантских масштабах из-за прибыльных платежей, требуемых и часто получаемых бандами киберпреступников. С добавлением двойного вымогательства атаки программ-вымогателей стали еще более привлекательными: даже если жертва отказывается платить, украденные личные данные могут быть проданы на форуме даркнета за значительную сумму.

Прошли те времена, когда кибератаки осуществлялись энтузиастами-одиночками, которым иногда помогали друзья и единомышленники. Как стало известно из недавней утечки Conti, бэкэнд современной высококлассной киберпреступной операции напоминает структуру гигантских ИТ-компаний, сотрудники которых могут находиться по всему миру с определенными ролями и обязанностями. Судя по вниманию к деталям, которое мы наблюдали в недавнем инциденте с Black Basta, обнаруженном группой реагирования на инциденты Check Point, операторы, стоящие за этой программой-вымогателем, также имеют впечатляющую организационную структуру.

*С мая 2022 года было возбуждено более 89 дел о вымогательстве авторитетных организаций со стороны банды Black Basta. Данные показывают четкую географическую направленность группы на США и Германию; 49% жертв, перечисленных на их сайте, имеют аккаунты в США. Требование выкупа в некоторых случаях превышало 1 миллион долларов США .

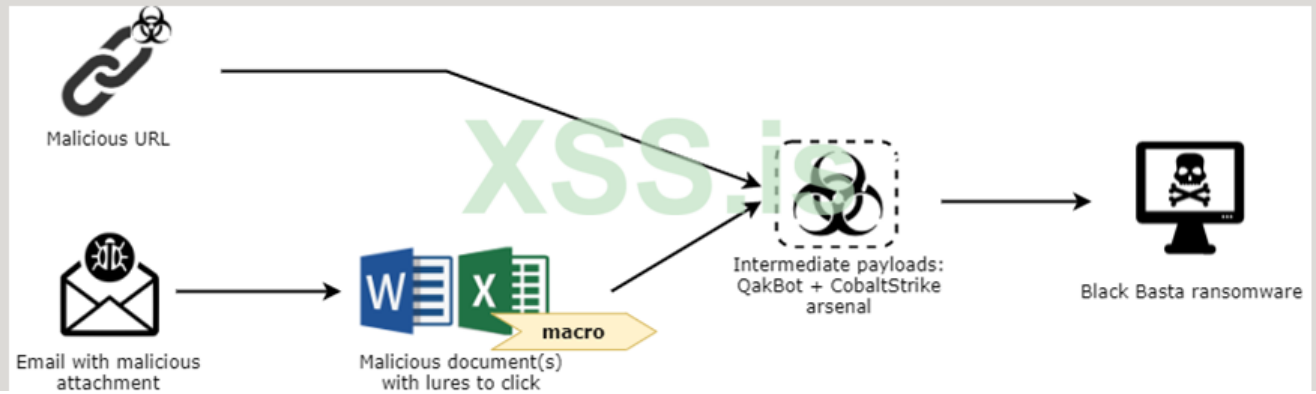
| Country | Number of victims | Percentage of victims |
|---|---|---|
| **United States** | 44 | 38% |
| **Germany** | 16 | 14% |
| **United Kingdom** | 4 | 3.5% |
| **Austria** | 3 | 2.6% |
| **Canada** | 3 | 2.6% |
| **Switzerland** | 3 | 2.6% |
| **Denmark** | 2 | 1.74% |
| **France** | 2 | 1.74% |
| **India** | 2 | 1.74% |
| **Italy** | 2 | 1.74% |
| **Other** | 6 | 5.22% |
| Total: | 87 | 100% |



В статье ниже мы описываем внутреннюю работу кампании Black Basta и уделяем особое внимание этапу доставки, на котором выполняются основные приготовления для беспрепятственного запуска программы-вымогателя. Рассказываем обо всех многочисленных приемах обхода и антианализа, которые мешают эмуляторам и песочницам обнаруживать и анализировать угрозу в автоматическом режиме. Мы даем ссылки на наши энциклопедии Anti-Debug и Evasions в каждой соответствующей статье: эти сайты являются основными источниками многочисленных методов, сгруппированных по категориям, с примерами кода и возможными контрмерами. Наконец, что не менее важно, мы представляем обзор того, как Black Basta шифрует файлы в системе и как он способен к боковому перемещению .

**Технические подробности**

Прежде чем начнется фактическое выполнение программы-вымогателя, программа-вымогатель должна быть доставлена на компьютер жертвы. Благодаря творчеству и развитым социальным навыкам членов синдиката киберпреступников у дроппера есть разные способы доставить свою полезную нагрузку на машину выбранной жертвы. Также может быть цепочка выполнения модулей дроппера (мы наблюдали комбинацию полезной нагрузки QakBot и Cobalt Strike ), что в конечном итоге приводит к выполнению программы-вымогателя.

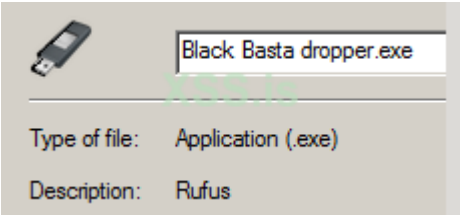



Мы заметили, что дропперы могут быть гораздо более сложными, чем просто технически более простая полезная нагрузка программы-вымогателя. Ниже мы описываем заключительный этап доставки программы-вымогателя Black Basta.

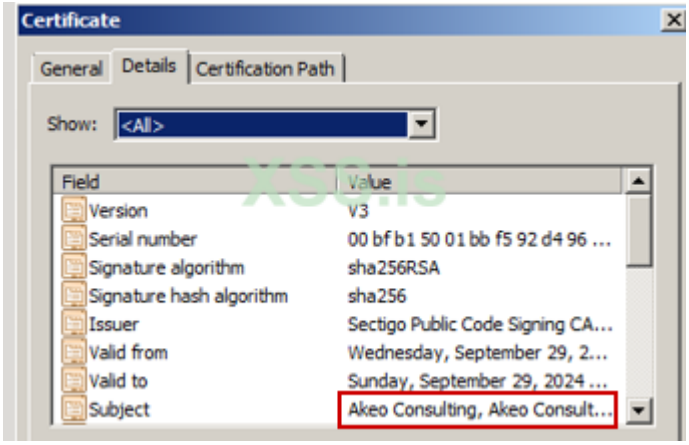
Этап доставки

Дроппер Black Basta имитирует приложение для создания загрузочных USB-накопителей, размещенное на этом сайте - (https://rufus.ie/en/):

Приложение имеет цифровую подпись с тем же сертификатом (выданным "Акео Консалтинг"), который используется для законных исполняемых файлов с веб-сайта Rufus:




Подробнее о том, как создать вредоносное приложение с проверенной цифровой подписью, см. в специальной статье исследовательской группы Check Point (https://research.checkpoint.com/202... signature-verification-putting-users-at-risk/) .




Методы уклонения и антианализа

В дроппере Black Basta реализовано немало антиотладочных приемов, перечисленных ниже, сгруппированных по категориям. Щелкните ссылки для получения дополнительной информации.

```
if ( rl_ev_peb_being_debugged() != 18534 )
  return 28790;
sub_404636();
sub_403009();
if ( rl_ev_check_remote_debugger() != 18534 )
  return 6515;
if ( rl_ev_findwindow_AwtFrame() != 18534 )
  return 8825;
if ( rl_ev_is_debugger_present() != 18534 )
  return 16465;
if ( rl_ev_peb_nt_global_flag() != 18534 )
  return 28790;
if ( rl_ev_check_debug_port() != 18534 )
  return 30295;
if ( rl_ev_GetWriteWatch() != 18534 )
  return 9286;
if ( rl_ev_close_bad_handle() != 18534 )
  return 21266;
```

```
if ( rl_ev_eeflags() != 18534 )
  return 30020;
if ( rl_ev_debugbreak() != 18534 )
  return 30497;
if ( rl_ev_int_2d() != 18534 )
  return 4969;
sub_40552F();
if ( rl_ev_int3() != 18534 )
  return 9528;
if ( rl_ev_outputdebugstring() != 18534 )
  return 17715;
if ( rl_ev_rdtsc() != 18534 )
  return 26450;
if ( rl_ev_queryperfcounter() != 18534 )
  return 9730;
if ( rl_ev_gettickcount() != 18534 )
  return 16722;
if ( rl_ev_check_hw_regs() != 18534 )
  return 26914;
if ( rl_ev_check_trap_flag() != 18534 )
  return 22913;
```

```
sub_4045FC();
if ( rl_log_pollute_beep() != 22647 )
  return 24614;
sub_401867();
sub_402B6E();
sub_40229B();
sub_403EAB();
sub_401B34();
if ( rl_ev_load_libs_check() != 22647 )
  return 18584;
sub_4023F8();
sub_404996();
sub_403A7B();
sub_4030A0();
if ( rl_ev_kernel_debugger_check() == 22647 )
  return 22647;
else
  return 28977;
```

Если какой-либо из этих методов успешно обнаруживает отладчик и/или среду эмуляции, дроппер останавливает свое выполнение и завершает работу, не запуская Black Basta.

Системные флаги

Эта группа методов защиты от отладки опирается на внутрипроцессные структуры для проверки состояния: выполняется ли отладка.

**- PEB: is debugger present**
**- PEB: being debugged**
**- PEB: NtGlobalFlag**
**- CheckRemoteDebugger**
**- Check kernel debugger**

**Регистры процессора**

Методы, сгруппированные ниже, используют регистры ЦП для проверки того, отлаживается ли процесс.

**- Set trap flag**
**- Check trap flag, same as above. Flag is not set, just checked**
**- Check HW breakpoints (method 1 in the link)**

**Инструкции процессора**

Эти методы используют инструкции ЦП через прямые вызовы или оболочки, чтобы проверить, отлаживается ли процесс.

**- DebugBreak**
**- INT 2D**
**- INT3**

**Проверка времени:**

Эти методы выполняют проверки времени, чтобы увидеть различия между отлаживаемым процессом и тем, который выполняется без отладчика.

**- RDTSC**
**- QueryPerformanceCounter**
**- GetTickCount**

**Библиотечные проверки:**

Этот метод основан на предположении, что в обычной системе есть некоторые общие системные библиотеки, которые могут быть загружены без проблем, и что есть также некоторые необычные библиотеки, которые не должны присутствовать в типичной системе. Однако в среде песочницы при попытке загрузить необычную библиотеку может быть возвращен предопределенный код вместо того, что в этих случаях возвращается на неэмулируемой машине. Различия в возвращаемом коде может быть достаточно для обнаружения песочницы.

Библиотеки, которые необходимо загрузить:

**- kernel32.dll**
**- networkexplorer.dll**
**- NlsData0000.dll**

Библиотеки, которые нельзя загружать:

**- NetProjW.dll**
**- Ghofr.dll**
**- fg122.dll**

**Проверки API Windows**

Следующая группа методов использует функции Windows API для определения того, отлаживается ли процесс.

**- VirtualAlloc в сочетании с GetWriteWatch**
**- CloseHandle с неверным дескриптором**
**- OutputDebugString для проверки последней системной ошибки**

**Загрязнение журнала:**

Этот метод на самом деле не является анти-отладчиком, но усложняет анализ журнала. Суть в том, чтобы сделать случайное количество вызовов функции kernel32.beep. Вы можете увидеть больше в этом анализе песочницы (https://www.joesecurity.org/blog/9048980422564630717#) .

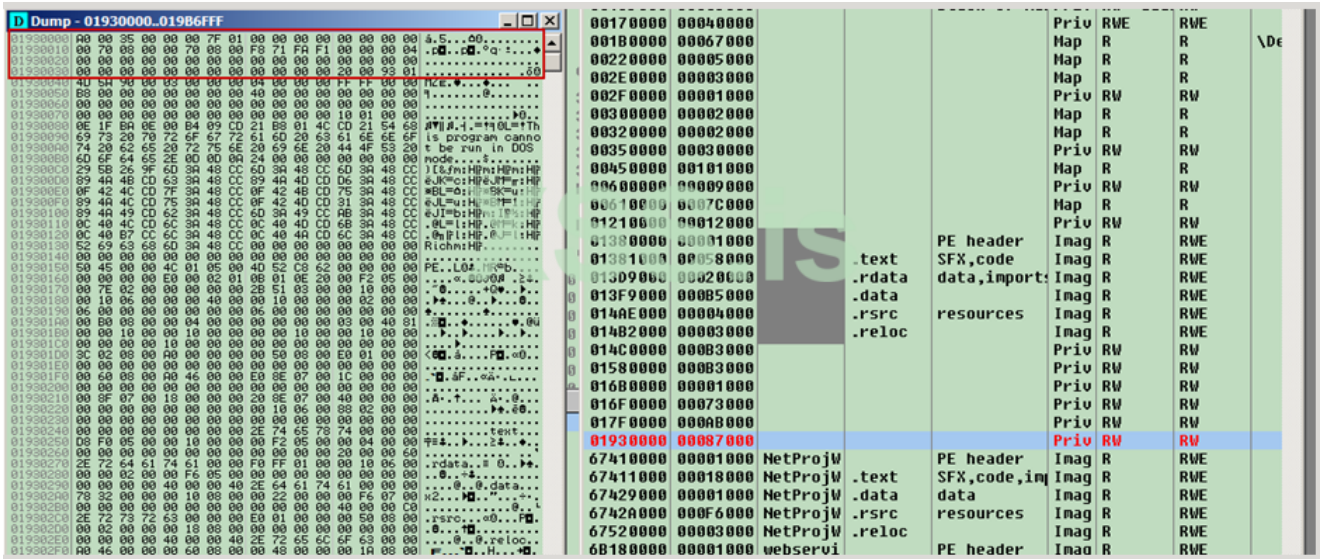
Сбой проверки из-за ошибки кода

Предполагается, что эти проверки используют особенности либо среды эмуляции, либо отлаженного процесса, но не работают должным образом из-за ошибок в коде.

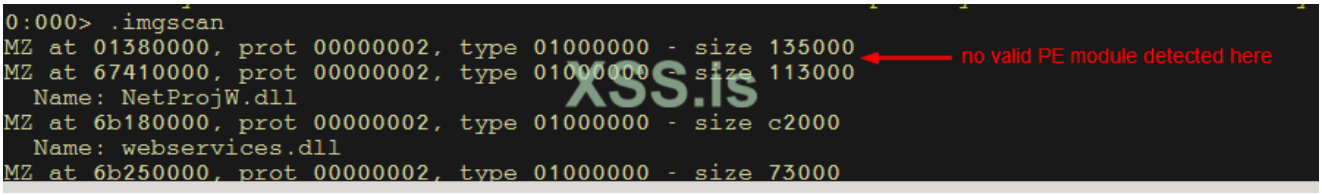
**- FindWindow (имя класса: —unAwtFrame ) — первый символ в имени неправильный; это должен быть SunAwtFrame**
**- NtQueryInformationProcess, проверьте DebugPort — не работает из-за неправильного имени dll**

**Непонятный дамп**

После успешного прохождения этапа техники уклонения у дроппера есть еще одна хитрость. Полезная нагрузка Black Basta не просто распаковывается и выполняется в памяти; есть данные, расположенные перед PE-заголовком программы-вымогателя, чтобы автоматические сканеры не могли легко идентифицировать вредоносную полезную нагрузку.




Как и ожидалось, команда .imgscan в WinDbg не может обнаружить PE-модуль Black Basta в памяти процесса дроппера.




После того, как все эти шаги пройдены, выполняется фактическая полезная нагрузка Black Basta.

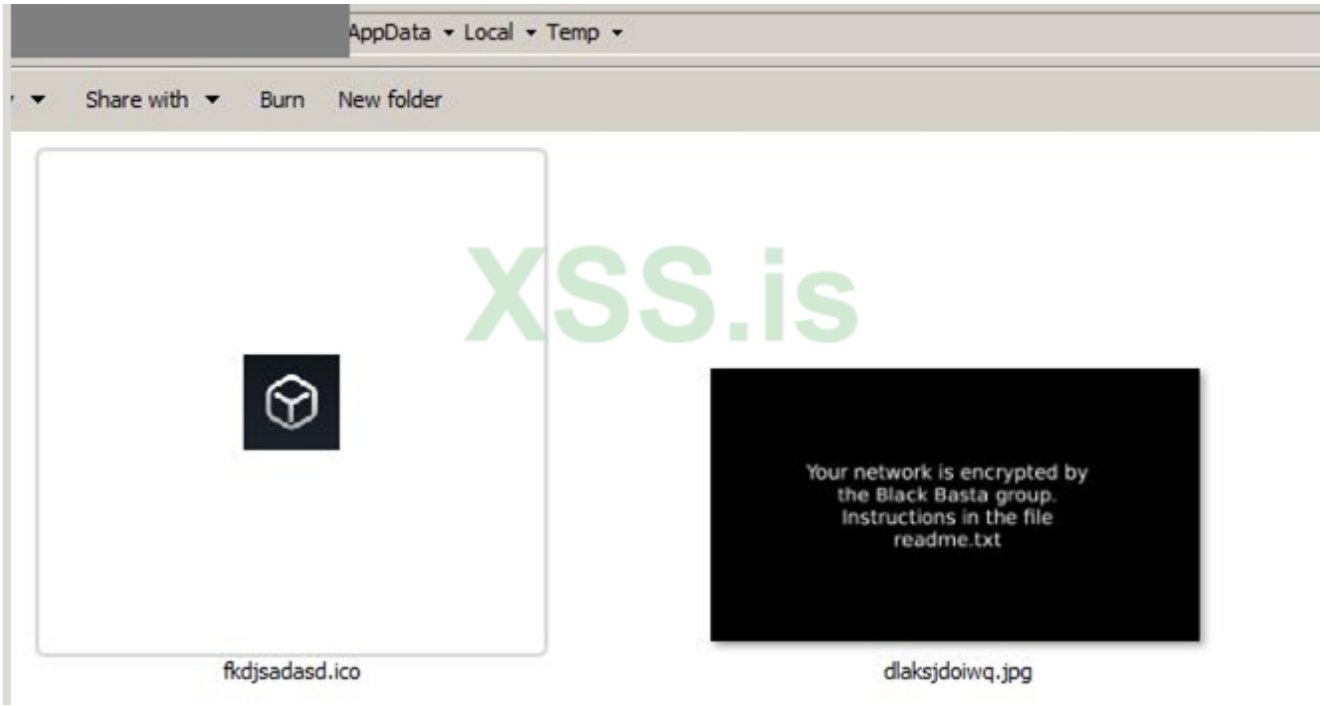
Полезная нагрузка Black Basta

В начале выполнения программы-вымогателя создается мьютекс, чтобы убедиться, что активна только одна копия вредоносного ПО:

```
.text:0040C9C3 push    offset Name     ; "dsajdhas.0"
.text:0040C9C8 push    eax             ; bInitialOwner
.text:0040C9C9 push    eax             ; lpMutexAttributes
.text:0040C9CA call    ds:CreateMutexW
```

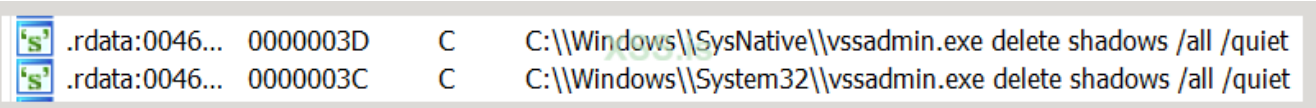
В примере, который мы описываем, имя мьютекса — «dsajdhas.o».

Затем вредоносная программа устанавливает обои и присваивает файлам с расширением ".basta" собственный значок.




Образы взяты из каталога TEMP, куда их распаковывает Black Basta.

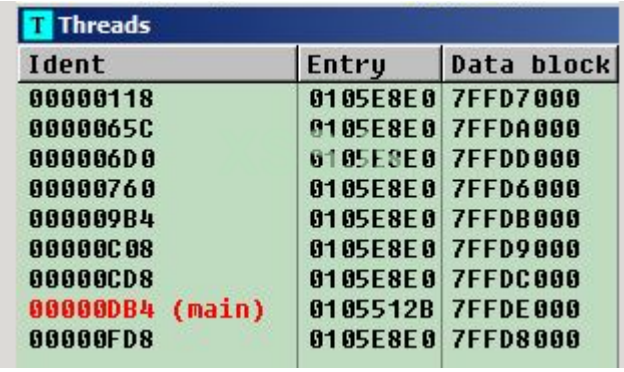
Программа-вымогатель также пытается удалить все копии теневых томов, как показано на рисунке ниже:




**Шифрование**

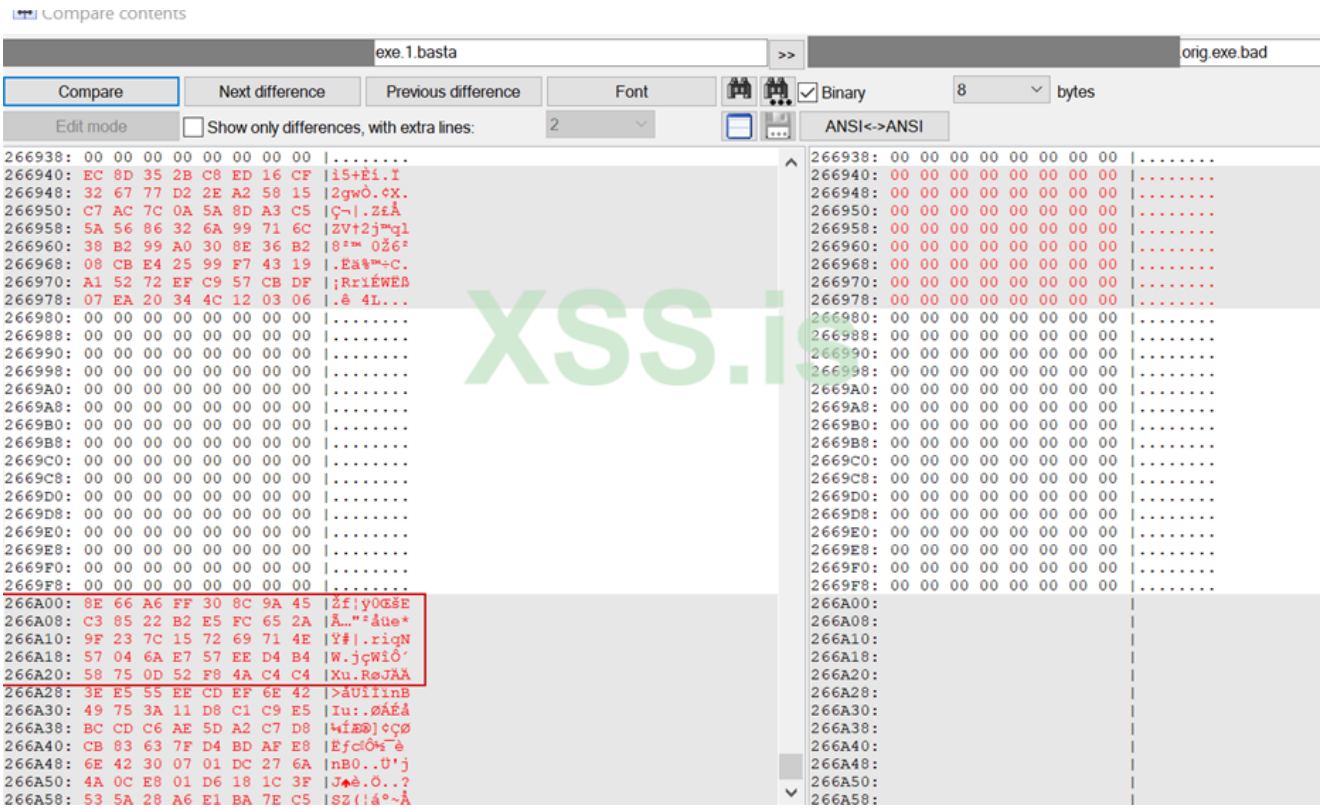
Несколько потоков создаются для создания многопоточного процесса шифрования:

Вредоносная программа шифрует все файлы, найденные на дисках, за исключением тех, в путях которых есть следующие строки:




**Windows**
**Documents and Settings**
**Local Settings**
**Application Data**
**txt**
**Boot**
**txt**
**jpg**
**DAT**
**ico**

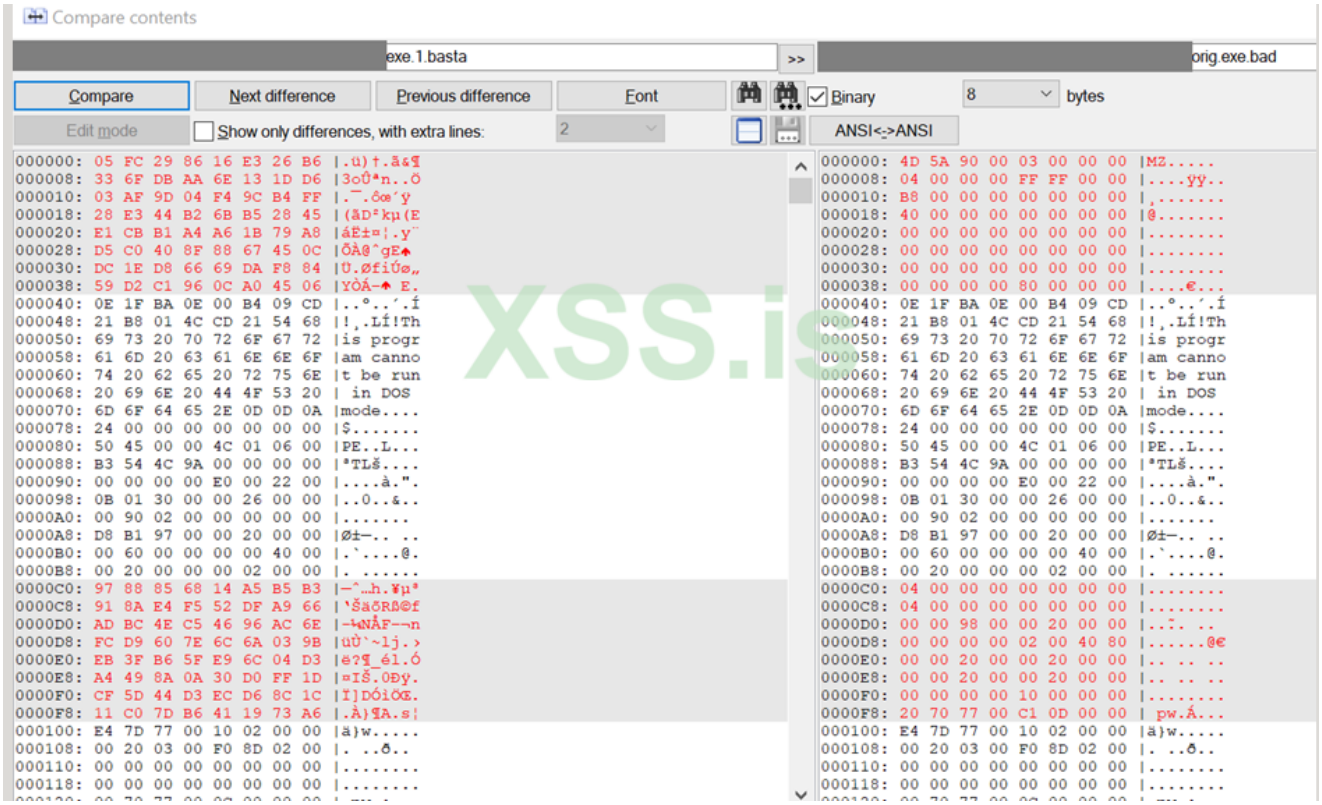
Потоковый шифр ChaCha20 (который, как сообщается в независимых исследованиях, быстрее, чем AES ) https://crypto.stackexchange.com/questions/34455/whats-the-appeal-of-using-chacha20-instead-of-aes используется для шифрования с ключом, сгенерированным случайным образом для каждого зашифрованного файла. Затем этот ключ передается в шифрование RSA с жестко закодированным открытым ключом для извлечения 512 байт зашифрованного ключа ChaCha20. Этот ключ добавляется в конец зашифрованного файла:




В конце блока также указана длина зашифрованного ключа (0x200):

```
266BE0: B0 D1 65 09 88 87 C7 42 |°Ñe.ˆ‡ÇB
266BE8: 48 F6 4C 37 0E 38 CB 51 |HöL7.8ËQ
266BF0: CC D8 2E 47 F4 69 91 40 |ÌØ.Gôi'@
266BF8: 72 22 8C 31 61 45 7B F8 |r"Œ1aE{ø
266C00: 00 02 00 00             |....
```

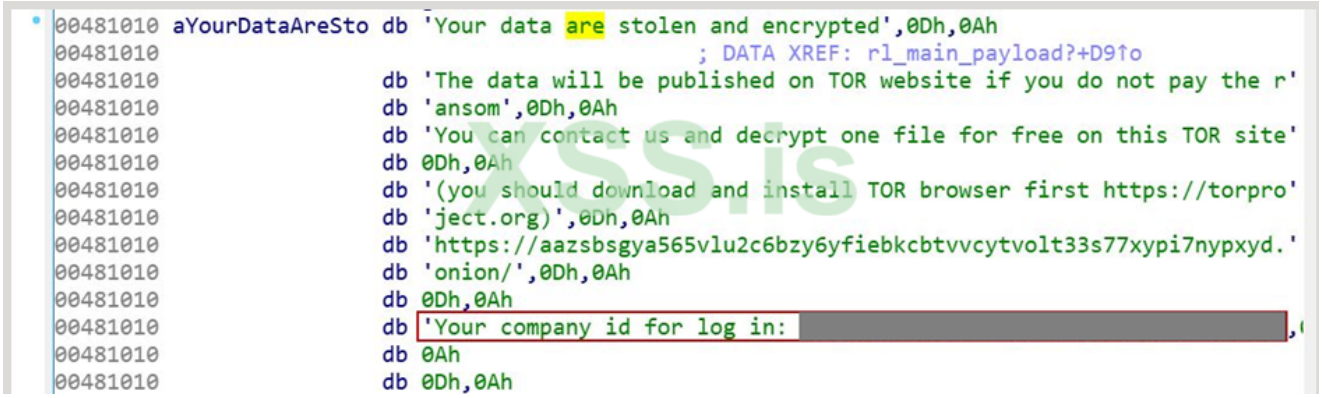
Обратите внимание, что не весь файл шифруется. Зловред нацелен на каждый третий блок из 64 байт:




Для обработки файла используются обычные функции kernel32:

**- CreateFile**
**- ReadFile**
**- WriteFile**
**- MoveFile (to rename an encrypted file)**

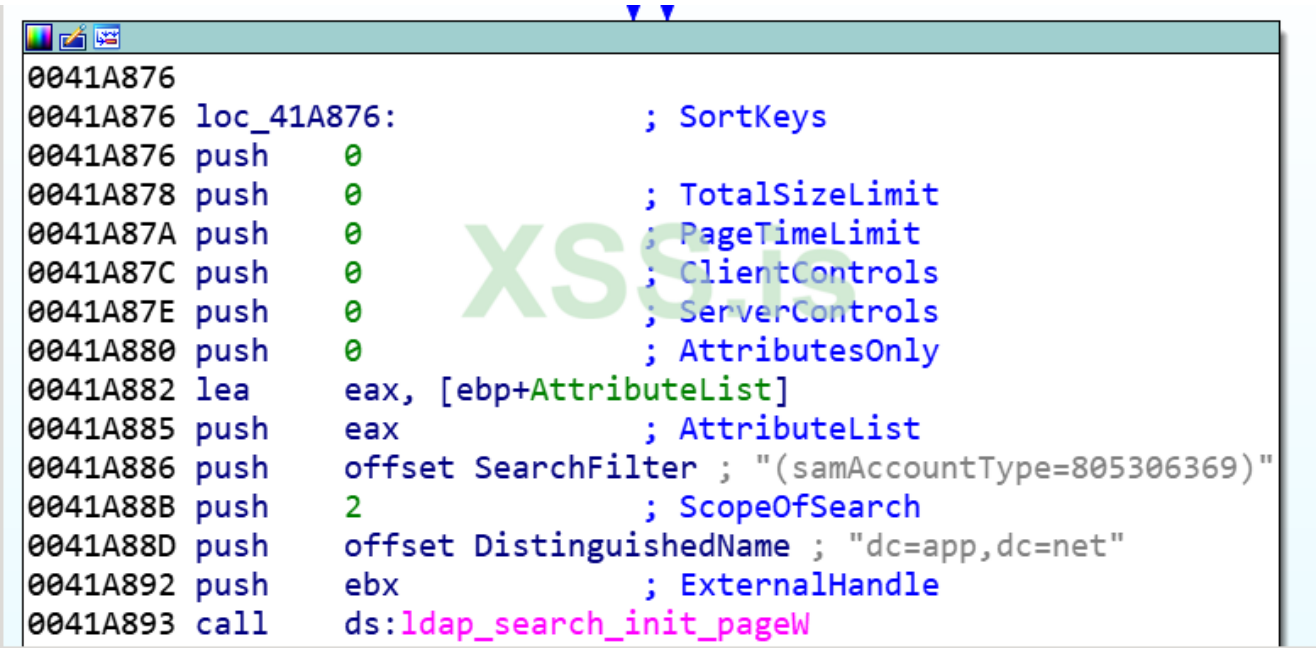
После завершения шифрования программа-вымогатель помещает примечание о выкупе в файл "readme.txt" на рабочем столе. В записке с требованием выкупа жестко прописан идентификатор компании, что является признаком целенаправленной и подготовленной атаки:




Нет очевидного способа расшифровать файлы, не зная закрытого ключа RSA.

**Автоматическое распространение**

В Black Basta встроен функционал автоматической раздачи в сети, если функций дропперов недостаточно для поставленной задачи. Программа-вымогатель пытается подключиться к AD с помощью LDAP API и перебирает подключенные рабочие станции, используя строку фильтра (samAccountType=805306369):




После получения списка рабочих станций вымогатель пытается скопировать себя на удаленные машины по пути \\c$\\Windows\\tmp.exe. Затем с помощью COM-объектов objectIWbemClassObject (CLSID: 4590F812-1D3A-11D0-891F-00AA004B2E24) и IWbemServices->Win32_Process через метод Create запускается скопированный на предыдущем этапе исполняемый файл .

**Вывод**

Атаки программ-вымогателей являются одной из самых серьезных угроз, с которыми может столкнуться жертва. Современные атаки программ-вымогателей имеют опыт многочисленных успешных вымогательств и могут перемещаться по сети в горизонтальном направлении, что приводит к все большему и большему гарантированному вознаграждению при использовании схемы двойного вымогательства.

Недавно появившаяся Black Basta уже является успешным игроком в программах-вымогателях, который принимает различные меры предосторожности и выполняет фактическое шифрование данных, о чем свидетельствуют применяемые методы

защиты от отладки и уклонения. Сочетание технических навыков, продемонстрированных бандой Black Basta, при успешном применении в атаке программ-вымогателей может привести к поистине разрушительным результатам.

Как видно из статьи, программа-вымогатель не только сама спроектирована таким образом, чтобы нанести максимальный ущерб за минимально возможное время, но и этап ее доставки скрыт, изощрен и эффективен. Black Basta без сомнения знает, что окружающая среда безопасна, и имеет все шансы выполнить шифрование.

Чтобы снизить вероятность стать жертвой этой и подобных атак, работодатели должны принять следующие меры:

**- Обучите своих сотрудников тому, как оставаться в безопасности в сфере кибербезопасности.**
**- Не открывайте некорпоративные вложения от неожиданных отправителей.**
**- Обновите и улучшите безопасность вашей киберинфраструктуры.**

**Регулярно делайте резервные копии конфиденциальных данных и храните их на внешних дисках.**
**- Держите свои системы в актуальном состоянии с помощью последних доступных обновлений.**

**Переведено специально для XSS.IS**
**Автор перевода: yashechka**
**Источник: https://research.checkpoint.com/2022/black-basta-and-the-unnoticed-delivery/**