Банковский троян Hesperbot – детальный анализ

Мы уже писали про Hesperbot, эта угроза представляет из себя новое банковское вредоносное ПО и имеет модульную архитектуру. Злоумышленники использовали его для проведения атак на пользователей различных стран, включая Турцию, Чехию, Португалию и Великобританию. Основной целью атак было похищение конфиденциальных данных онлайн-банкинга пользователей и установка мобильного компонента вредоносного кода на устройства под управлением Symbian, Android, Blackberry. Киберпреступники использовали убедительную схему фишинга для заманивания пользователей на вредоносные ссылки, что делало их подход еще более практичным.

Компрометация пользователя происходит при переходе по вредоносной ссылке. Ниже показана схема, используемая злоумышленниками.

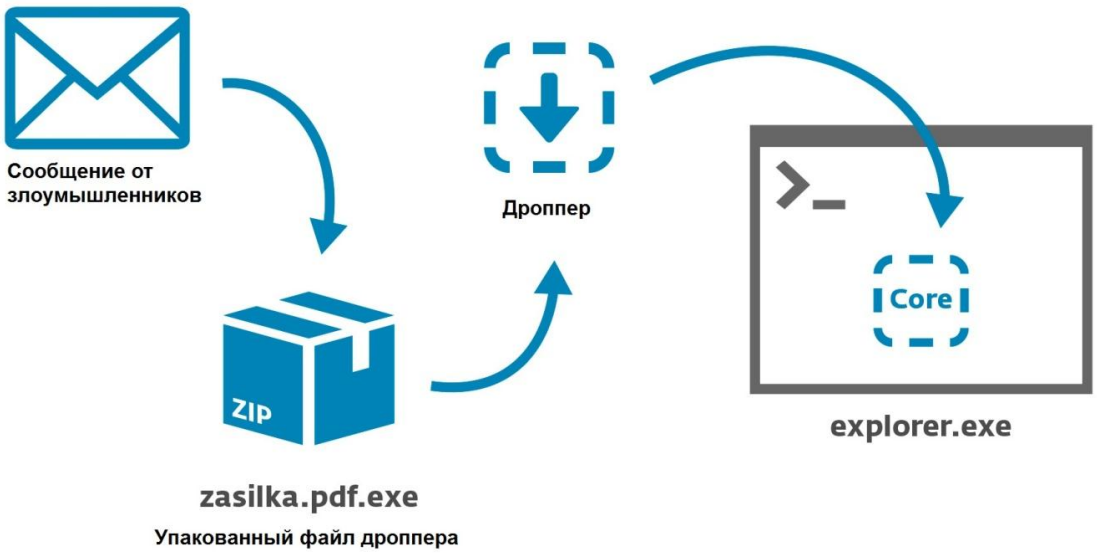


Рис. Схема компрометации пользователя.

Основная задача дроппера заключается во внедрении полезной нагрузки (payload или core) в контекст доверенного процесса explorer.exe. Этот код затем скачивает из сети и исполняет дополнительные модули.

Модуль	Описание
dropper	Используется для внедрения компонента **core** в контекст explorer.exe
core	Взаимодействует с управляющим C&C сервером и скачивает дополнительные модули
nethk	Перехват сетевого трафика, веб-инжекты, захват видео и скриншотов рабочего стола
httphk	
httpi	
keylog	Кейлоггер
hvnc	VNC-сервер
sch	Вспомогательный модуль установки перехватов
socks	SOCKS5 прокси-сервер

Win32/Spy.Hesperbot является кросс-платформенной угрозой и различные перечисленные модули доступны в x86 и x64 вариантах. Различные внутренние функции одних модулей доступны для использования в других модулях через специальную таблицу виртуальных методов (vtable). Бо́льшая часть кода Hesperbot написана на C и скомпилирована с использованием Visual Studio 2010 без использования библиотеки run-time.

Полезная нагрузка

Дроппер может использовать один из следующих методов для внедрения компонента core в адресное пространство explorer.exe:

- Запустить новый экземпляр explorer.exe и исправить (patching) байты его точки входа с использованием *ntdll!NtGetContextThread* и *ntdll!NtWriteProcessMemory*. После этого точка входа будет указывать на код вредоносной программы.
- Внедрить свой код в уже существующий процесс explorer.exe с использованием метода, основанного на функциях *Shell_TrayWnd/SetWindowLong/SendNotifyMessage*. Данный метод использовался во вредоносном коде PowerLoader, который ранее был описан Александром Матросовым [1, 2 ,3].

- Внедрить код в explorer.exe с использованием обычной функции *CreateRemoteThread*.

После успешного внедрения, компонент core начинает в системе свою работу: отвечает за взаимодействие с управляющим C&C сервером, запуск вспомогательных модулей и запись параметров автозагрузки вредоносного кода в системный реестр. Для работы с C&C, Hesperbot использует список жестко зашитых в тело URL-адресов (которые различаются в случае разных стран и ботнетов) или генерирует список новых адресов с использованием специального DGA-алгоритма. Вредоносный код отсылает следующую информацию на C&C сервер:

- Имя бота, которое основано на Computer Name.
- Название ботнета, которое формируется исходя из страны, на которую угроза была нацелена злоумышленниками. Например, «cz-botnet», «tr-botnet», «pt-botnet», «uk-botnet» и «super-botnet» (последний использовался в ранних бета-версиях).
- IP-адреса сетевых адаптеров.
- Названия установленных смарт-карт.
- Информация об установленных плагинах Hesperbot.

```
push    ebx
push    12h
push    offset aCzBotnet ; "cz-botnet"
push    4
mov     eax, esi
call    container_add_value
```

Рис. Идентификатор чешского ботнета.

Ответ сервера может содержать следующую информацию или файлы:

- Конфигурационный файл.
- Файлы плагинов.
- Произвольный исполняемый файл для последующего запуска.
- Новая версия Hesperbot.

Вредоносный код получает список активных смарт-карт в системе с использованием следующих API: *SCardEstablishContext*, *SCardListReaders* и *SCardConnect*. В отличие от других, более сложных случаев угроз [1, 2], Win32/Spy.Hesperbot собирает только названия смарт-карт и не имеет возможности с ними взаимодействовать.

Загружаемые файлы (файлы конфигурации и исполняемые файлы плагинов) шифруются с использованием алгоритма Twofish, 256-битный хэш которого основан на данных:

- Имени компьютера (Computer Name).
- Параметра «InstallDate» раздела реестра [HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion].
- Версии ОС.
- Архитектуры процессора (x86, x64, IA64).
- Параметра «MachineGuid» раздела реестра [HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Cryptography].

- Параметра «DigitalProductId» раздела реестра [HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion].

Для хранения загруженных с C&C данных и других вспомогательных файлов (например, лог-файла кейлоггера), Hesperbot создает директорию с произвольным именем в папке %APPDATA%.

Модуль Core обладает возможностью внедрения своего кода не только в explorer.exe, но и в другие запущенные процессы. Вредоносный код использует недокументированный трюк с перехватом UserNotifyProcessCreate внутри csrss.exe для того чтобы быть уверенным в активности своего кода в каждом запущенном процессе.

Компонент кейлоггера осуществляет перехват нажатий клавиш через функции GetMessage и TranslateMessage. Далее символы сохраняются в лог-файл с информацией о процессе и названием окна, в котором они вводились. После чего файл отправляется на сервер злоумышленников.

Модуль захвата скриншотов рабочего стола и видео называется httpi. Захват видео был использован и в банковской троянской программе Zeus. Данная возможность предоставляет злоумышленникам видеть более полную картину происходящего у пользователя в системе. Возможность реализуется с использованием API *AVIFileCreateStream*, *AVIFileMakeCompressedStream*, *AVIStreamWrite* из библиотеки Avifil32.dll.

```
push    0
lea     eax, [ebp+var_30]
push    eax
mov     [ebp+var_24], edi
push    ecx
lea     edi, [esi+84h]
push    edi
mov     [ebp+var_30], 'sdiv'
mov     [ebp+var_2C], 'CVSM'
mov     [ebp+var_28], 2710h
mov     [ebp+var_1C], 4
call    AVIMakeCompressedStream
test    eax, eax
```

Рис. Код создания видео.

Создание скриншотов рабочего стола производится с использованием функций библиотеки Gdi32.dll.

Возможность VNC-соединения ранее наблюдалась в известной троянской программе Carberp. Она позволяет создать VNC-сервер на стороне зараженного компьютера, к которому будет иметь доступ злоумышленник. Поскольку, в отличие от RDP, VNC не выполняет операцию переключения пользователя (user log off), атакующий может подключиться к жертве в процессе ее работы. Это достигается за счет создания вспомогательного рабочего стола (*CreateDesktop*), который невидим пользователю. Этот модуль предоставляет злоумышленнику возможность запускать процесс браузера, установленного в системе. В таком случае злоумышленник получает доступ к его сохраненным данным (cookie, сессии и прочее).

Сетевое взаимодействие и веб-инжекты

Hesperbot не является первым банковским вредоносным ПО, которое использует перехваты функций библиотеки WinSock (*send*, *WSASend* и прочие) и библиотеки WinInet (*HttpSendRequest*, *InternetReadFile* и прочие), для получения полного контроля за сетевым HTTP/HTTPS трафиком пользователя. Схожие механизмы использовались в известых троянах Zeus и SpyEye. Использование такого подхода дает возможность злоумышленникам осуществлять веб-инжекты (web-injects), получать введенные в веб-формы данные (form-grabbing) и проводить другие операции по похищению нужной злоумышленникам информации, которую пользователь вводит в страницы, открываемые в веб-браузере. Перечисленные методы получили название «Man-in-the-Browser» attack. Win32/Spy.Hesperbot использует несколько иной подход при проведении атаки на пользователя, который мы уже наблюдали в другом банковском трояне Win32/Gataka.

Перехват трафика и внедрение HTML-кода осуществляется модулями nethk, httphk и httpi, которые упоминались выше в таблице компонентов.

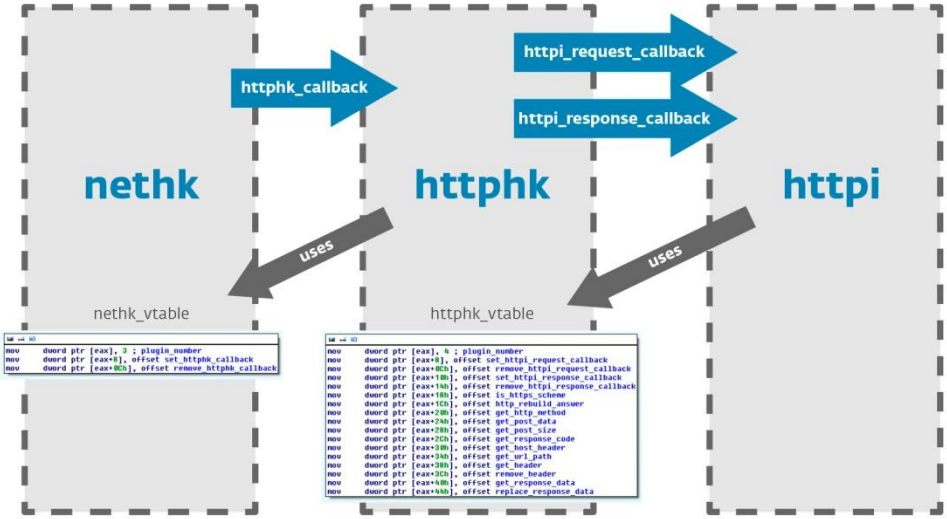


Рис. Взаимодействие различных модулей Hesperbot.

- **nethk** – используется для развертывания локального прокси, перехватывает функции для работы с сокетами при отслеживании соединений; содержит перехваты для функций проверки SSL-сертификатов в браузере.
- **httphk** – используется при разборе HTTP-трафика.
- **httpi** – используется для снятия скриншотов, захвата видео, снятие данных форм и внедрение HTML-кода (web-injects).

Модуль **nethk** является первым скачиваемым плагином вредоносного кода. Его скачивание осуществляется компонентом **core**. Win32/Spy.Hesperbot осуществляет атаку man-in-the-middle через создание локального прокси (local proxy), через который он может контролировать все соединения, инициируемые процессом браузера.

```
push    edx
push    offset a127_0_1_1D ; "127.0.1.1:%d"
lea     edx, [ebp+proxy_addr]
call    sub_10005590
mov     edx, [ebx+WSPPROC_TABLE.lpWSPStringToAddress]
add     esp, 8
```

Рис. Код создания прокси.

TCPView - Sysinternals: www.sysinternals.com

File Options Process View Help

Process	PID	Protocol	Local Address	Local Port	Remote Address	Remote Port	State
alg.exe	1988	TCP	127.0.0.1	1028	0.0.0.0	0	LISTENING
iexplore.exe	3180	UDP	127.0.0.1	1035	*	*	
iexplore.exe	3108	UDP	127.0.0.1	1036	*	*	
iexplore.exe	3180	TCP	127.0.1.1	44719	0.0.0.0	0	LISTENING
iexplore.exe	3108	TCP	127.0.1.1	16255	0.0.0.0	0	LISTENING
lsass.exe	868	UDP	0.0.0.0	500	*	*	
lsass.exe	868	UDP	0.0.0.0	4500	*	*	
svchost.exe	1120	TCP	0.0.0.0	135	0.0.0.0	0	LISTENING
svchost.exe	1260	UDP	127.0.0.1	1026	*	*	
svchost.exe	1260	UDP	127.0.0.1	123	*	*	
svchost.exe	1320	UDP	0.0.0.0	1025	*	*	
svchost.exe	1260	UDP	192.168.80.128	123	*	*	
svchost.exe	1384	UDP	127.0.0.1	1900	*	*	
svchost.exe	1384	UDP	192.168.80.128	1900	*	*	
System	4	TCP	192.168.80.128	139	0.0.0.0	0	LISTENING
System	4	TCP	0.0.0.0	445	0.0.0.0	0	LISTENING
System	4	UDP	192.168.80.128	137	*	*	
System	4	UDP	192.168.80.128	138	*	*	
System	4	UDP	0.0.0.0	445	*	*	

Endpoints: 19 | Established: 0 | Listening: 6 | Time Wait: 0 | Close Wait: 0

Рис. Сетевые соединения Internet Explorer через установленный прокси Hesperbot.

Троянская программа с использованием модуля **nethk** создает прокси на произвольном порту с адресом 127.0.1.1 и перехватывает следующие функции библиотеки mswsock.dll: *WSPSocket*, *WSPIoctl*, *WSPConnect*, *WSPCloseSocket*. Указатели на эти функции модифицируются в таблице *WSPPROC_TABLE*. Для понимания того, как работает это перенаправления ниже представлена вредоносная версия *WSPConnect*.

```
.text:10003640 ; int __stdcall hooked_WSPConnect(SOCKET s, const struct sockaddr *name, int namelen, LPWSABUF lpCallerData, LPWSABUF lpCalleeData, LPQOS lpSQOS, LPQOS lpGQOS, LPINT lpErrno)
.text:10003640 hooked_WSPConnect proc near             ; DATA XREF: WSPSocket_hooks_install+45↓o
.text:10003640
.text:10003640 s               = dword ptr  8
.text:10003640 name            = dword ptr  0Ch
.text:10003640 namelen         = dword ptr  10h
.text:10003640 lpCallerData    = dword ptr  14h
.text:10003640 lpCalleeData    = dword ptr  18h
.text:10003640 lpSQOS          = dword ptr  1Ch
.text:10003640 lpGQOS          = dword ptr  20h
.text:10003640 lpErrno         = dword ptr  24h
.text:10003640
.text:10003640                 push    ebp
.text:10003641                 mov     ebp, esp
.text:10003643                 cmp     is_WSPSocket_hooked, 0
.text:1000364A                 push    ebx
.text:1000364B                 mov     ebx, [ebp+namelen]
.text:1000364E                 push    esi
.text:1000364F                 push    edi
.text:10003650                 mov     edi, [ebp+lpErrno]
.text:10003653                 jz      short loc_10003688
.text:10003655                 mov     eax, [ebp+s]
.text:10003658                 call    get_connection
.text:1000365D                 mov     esi, eax
.text:1000365F                 test    esi, esi
.text:10003661                 jz      short loc_10003688
.text:10003663                 mov     ecx, [ebp+name]
.text:10003666                 push    edi             ; int
.text:10003667                 push    esi             ; lpParameter
.text:10003668                 mov     eax, ebx
.text:1000366A                 call    connect_to_proxy
.text:1000366F                 add     esp, 8
.text:10003672                 test    eax, eax
.text:10003674                 jz      short loc_100036A8
.text:10003676                 cmp     dword ptr [edi], WSAEWOULDBLOCK
.text:1000367C                 jz      short loc_100036A8
.text:1000367E                 add     esi, 0FFFFFFF8h
.text:10003681                 push    esi             ; lpAddend
.text:10003682                 call    ds:InterlockedDecrement
.text:10003688
.text:10003688 loc_10003688:                           ; CODE XREF: hooked_WSPConnect+13↑j
.text:10003688                                         ; hooked_WSPConnect+21↑j
.text:10003688                 mov     eax, [ebp+lpGQOS]
.text:1000368B                 mov     ecx, [ebp+lpSQOS]
.text:1000368E                 mov     edx, [ebp+lpCalleeData]
.text:10003691                 push    edi
.text:10003692                 push    eax
.text:10003693                 mov     eax, [ebp+lpCallerData]
.text:10003696                 push    ecx
.text:10003697                 mov     ecx, [ebp+name]
.text:10003698                 push    edx
.text:1000369B                 mov     edx, [ebp+s]
.text:1000369E                 push    eax
.text:1000369F                 push    ebx
.text:100036A0                 push    ecx
.text:100036A1                 push    edx
.text:100036A2                 call    original_WSPConnect
.text:100036A8
.text:100036A8 loc_100036A8:                           ; CODE XREF: hooked_WSPConnect+34↑j
.text:100036A8                                         ; hooked_WSPConnect+3C↑j
```

Рис. Вредоносный вариант *WSPConnect* API.

Например, если пользователь захотел войти в систему онлайн-банкинга и браузер пытается установить защищенное соединение с веб-сайтом банка, запрос на соединение будет перехвачен кодом Hesperbot для создания локального прокси, после чего система онлайн-банкинга будет иметь дело с сокетом злоумышленников.

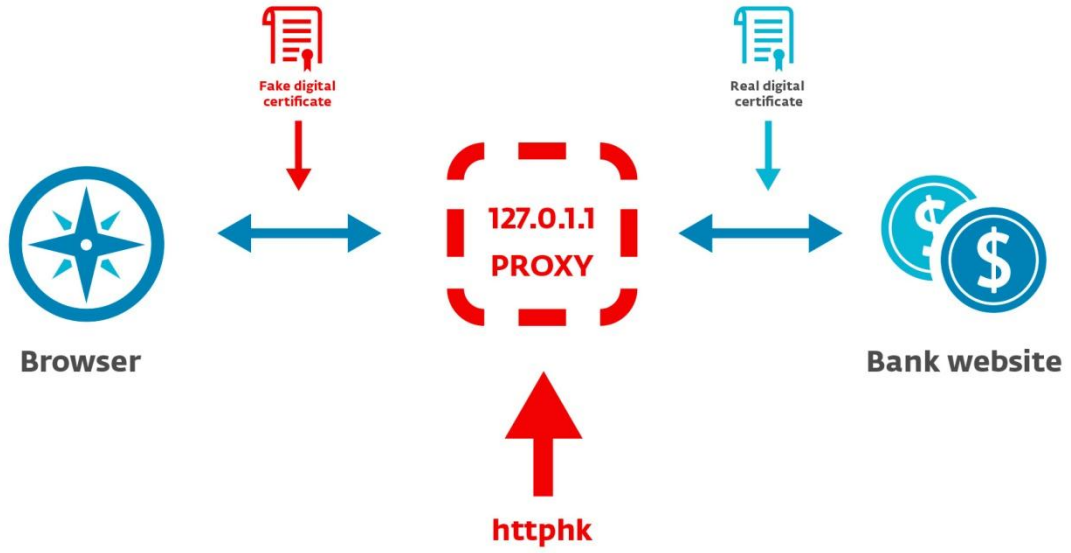


Рис. Схема работы вредоносного прокси.

Callback-функция модуля httphk вызывается каждый раз, когда прокси перехватывает запрос от браузера перед его передачей реальному серверу. Кроме этого, httphk вызывается каждый раз, когда прокси перехватывает ответ на запрос от реального сервера, перед тем как передать его браузеру. Далее httphk продолжает работу с трафиком.

Существуют различия в обработке HTTP и HTTPS трафика. В случае простого HTTP-соединения, данные в режиме запрос/ответ просто передаются от httphk и обратно. В случае же защищенного HTTPS соединения, вредоносный код использует возможности модуля **nethk** для того, чтобы «избавиться от шифрования». Когда HTTPS запрос, инициированный браузером, перехвачен (данные запроса уже зашифрованы с использованием фальшивого сертификата, см. ниже как), он расшифровывается и эти данные отправляются модулю httphk через callback-функцию и затем шифруются уже с помощью настоящего сертификата сервера (т. е. сертификата веб-сайта банка), после чего отправляются по назначению. С приходом ответа сервера, т. е. HTTPS-ответа, прокси расшифровывает его с использованием настоящего сертификата и передает эти данные к httphk для шифрования поддельным сертификатом перед передачей браузеру.

Таким образом, используя «man-in-the-middle» прокси, Win32/Spy.Hesperbot может получить доступ к исходящему HTTPS-трафику жертвы до того как он будет зашифрован настоящим сертификатом. Аналогичный подход использую Zeus и SpyEye, но их реализация MitB отличается от используемой в Hesperbot.

Вредоносный код отдает браузеру свой сертификат для имитации HTTPS-соединения. Модуль **nethk** имеет на борту собственный self-signed SSL сертификат.

```
-----BEGIN CERTIFICATE-----
MIIDhzCCAm+gAwIBAgIBADANBgkqhkiG9w0BAQUFADA7MQswCQYDVQQGEwJOTDER
MA8GA1UEChMIUG9sYXJTU0wxGTAXBgNVBAMTEFBvbGFyU1NMIFRlc3QgQ0EwHhcN
MTEwMjEyMTQ0NDAwWhcNMjEwMjEyMTQ0NDAwWjA7MQswCQYDVQQGEwJOTDERMA8G
A1UEChMIUG9sYXJTU0wxGTAXBgNVBAMTEFBvbGFyU1NMIFRlc3QgQ0EwggEiMA0G
CSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQDA3zf8F7vglp0/ht6WMn1EpRagzSHx
mdTs6st8GFgIlKXsm8WL3xoemTiZhx57wI053zhdcHgH057Zk+i5clHFzgMwUqny
50BwFMtEonILwuVA+T7lpg6z+exKY8C4KQB0nFc7qKUEkHHxvYPZP9al4jwqj+8n
YMPGn8u67GB9t+aEMr5P+igmIgNb1LTV+/Xjli5wwOQuvfwu7uJBVcA0Ln0kcmnL
R7EUQIN9Z/SG9jGr8XmksrUuEvmEF/Bibyc+E1ixVA0hmnM3oTDPb5Lc9un8rNsu
KNF+AksjoBXyOGVkCeoMbo4bF6BxyLObyavpw/LPh5aPgAIynplYb6LVAgMBAAGj
gZUwgZIwDAYDVR0TBAUwAwEB/zAdBgNVHQ4EFgQUtFrkpbPe0lL2udWmlQ/rPrzH
/f8wYwYDVR0jBFwwWoAUtFrkpbPe0lL2udWmlQ/rPrzH/f+hP6Q9MDsxCzAJBgNV
BAYTAk5MMREwDwYDVQQKEwhQb2xhclNTTDEZMBcGA1UEAxMQUG9sYXJTU0wgVGVz
dCBDQYIBADANBgkqhkiG9w0BAQUFAAOCAQEAuP1U2ABUkIslsCfdlc2i94QHHYeJ
SsR4EdgHtdciUI5I62J6Mom+YOdT/7a+8S6MVMCZP6C5NyNyXw1GWY/YR82XTJ8H
DBJiCTok5DbZ6SzaONBzdWHXwWwmi5vg1dxn7YxrM9d0IjxM27WNKs4sDQhZBQkF
pjmfs2cb4oPl4Y9T9meTx/lvdkRYEug61Jfn6cA+qHpyPYdTH+UshITnmp5/Ztkf
m/UTSLBNFNHesiTZeH31NcxYGdHSme9Nc/gfidRaOFLOCfWxRlFqAI47zG9jAQCZ
7Z2mCGDNMhjQc+BYcdnl0lPXjdDK6V0qCg1dVewhUBcW5gZKzV7e9+DpVA==
-----END CERTIFICATE-----
      -----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: DES-EDE3-CBC,A8A95B05D5B7206B

9Qd9GeArejl1GDVh2lLV1bHt0cPtfbh5h/5zVpAVaFpqtSPMrElp50Rntn9et+JA
7VOyboR+Iy2t/HU4WvA687k3Bppe9GwKHjHhtl//8xFKwZr3Xb5yO5JUP8AUctQq
Nb8CLlZyuUC+52REAAthdWgsX+7dJO4yabzUcQ22Tp9JSDOhiL43BlkWYUNK3dAo
PZlmiptjnzVTjg1MxsBSydZinWOLBV8/JQgxSPo2yD4uEfig28qbvQ2wNIn0pnAb
GxnSAOazkongEGfvcjIIs+LZN9gXFhxcOh6kc4Q/c99B7QWETwLLkYgZ+z1a9VY9
gEU7CwCxYCD+h9hY6FPmsK0/lC4O7aeRKpYq00rPPxs6i7phiexg6ax6yTMmArQq
QmK3TAsJm8V/J5AWpLEV6jAFgRGymGGHnof0DXzVWZidrcZJWTNuGEX90nB3ee2w
PXJEFWKoD3K3aFcSLdHYr3mLGxP7H9ThQai9VsycxZKS5kwvBKQ//YMrmFfwPk8x
vTeY4KZMaUrveEel5tWZC94RSMKgxR6cyE1nBXyTQnDOGbfpNNgBKxyKbINWoOJU
WJZAwlsQn+QzCDwpri7+sV1mS3gBE6UY7aQmnmiiaC2V3Hbphxct/en5QsfDOt1X
JczSfpRWLlbPznZg8OQh/VgCMA58N5Dj0zTIK7sJJ5r+94ZBTCpgAMbF588f0NTR
KCe4yrxGJR7X02M4nvD4IwOlpsQ8xQxZtOSgXv4LkxvdU9XJJKWZ/XNKJeWztxSe
Z1vdTc2YfsDBA2SEv33vxHx2g1vqtw8SjDRT2RaQSS0QuSaMJimdOX6mTOCBKk1J
9Q5mXTrER+/LnK0jEmXsBXWA5bqqVZIyahXSx4VYZ7l7w/PHiUDtDgyRhMMKi4n2
iQvQcWSQTjrpnlJbca1/DkpRt3YwrvJwdqb8asZU2VrNETh5x0QVefDRLFiVpif/
tUaeAe/P1F8OkS7OIZDs1SUbv/sD2vMbhNkUoCms3/PvNtdnvgL4F0zhaDpKCmlT
P8vx49E7v5CyRNmED9zZg4o3wmMqrQO93PtTug3Eu9oVx1zPQM1NVMyBa2+f29DL
1nuTCeXdo9+ni45xx+jAI4DCwrRdhJ9uzZyC6962H37H6D+5naNvClFR1s6li1Gb
nqPoiy/OBsEx9CaDGcqQBp5Wme/3XW+6z1ISOx+igwNTVCT14mHdBMbya0eIKft5
X+GnwtgEMyCYyyWuUct8g4RzErcY9+yW9Om5Hzpx4zOuW4NPZgPDTgK+t2RSL/Yq
rE1njrgeGYcVeG3f+OftH4s6fPbq7t1A5ZgUscbLMBqr9tK+OqygR4EgKBPsH6Cz
L6zlv/2RV0qAHvVuDJcIDIgwY5rJtINEm32rhOeFNJwZS5MNIC1czXZx5//ugX7l
I4sy5nbVhwSjtAk8Xg5dZbdTZ6mIrb7xqH+fdakZor1khG7bC2uIwibD3cSl2XkR
wN48slbHnqqagr6Xm1nNOSVl8C/6kbJEsMpLhAezfRtGwvOucoaE+WbeUNolGde
P/eQiddSf0brnpiLJRh7qZrl9XuqYdpUqnoEdMAfotDOID8OtV7gt8a48ad8VPW2
-----END RSA PRIVATE KEY-----
```


Рис. SSL сертификат внутри nethk.

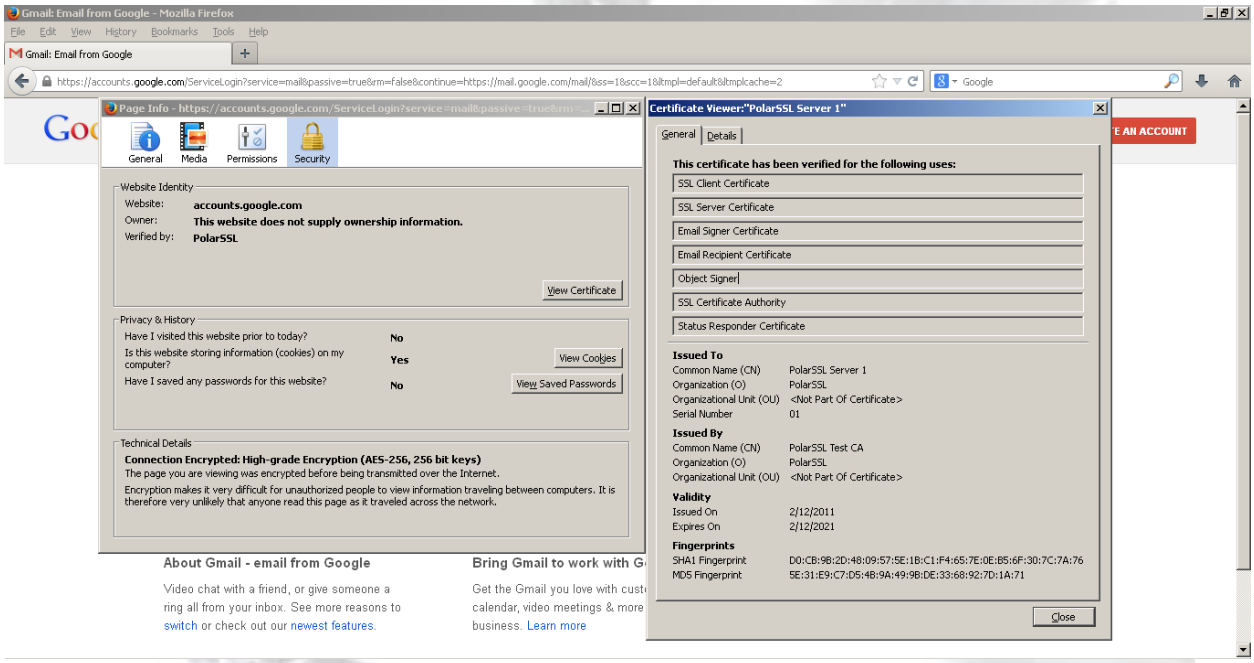


Рис. Пример использования поддельного сертификата Hesperbot. На обычной системе можно увидеть реальный сертификат Google вместо поддельного.

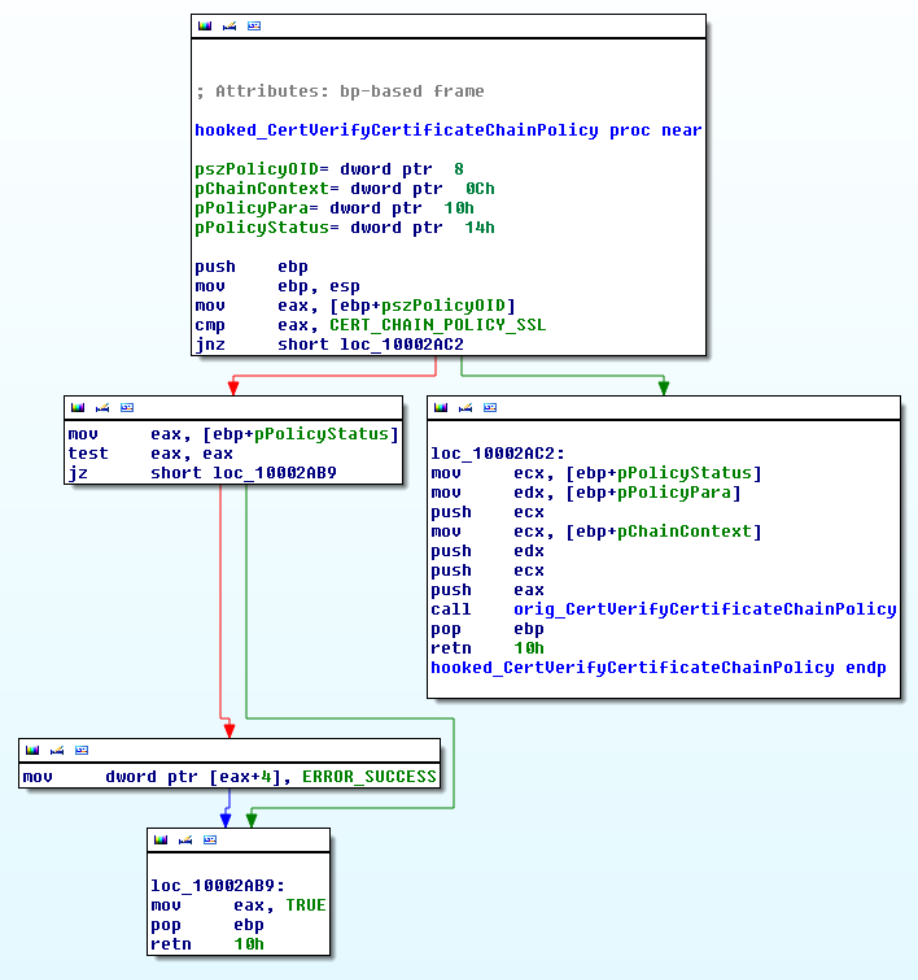
Для того, чтобы обмануть компонент проверки действительности сертификата браузера, вредоносный код перехватывает соответствующие функции проверки сертификата. Релизация подобных перехватов зависит от браузера жертвы. Ниже в таблице представлены браузеры, поддерживаемые Win32/Spy.Hesperbot, и перехватываемые функции.

Процесс браузера	**Перехватываемая функция**
iexplore.exe	*crypt32!CertVerifyCertificateChainPolicy* и *crypt32!CertGetCertificateChain*
maxthon.exe	
avant.exe	
sleipnir.exe	
webkit2webprocess.exe	
browser.exe	
chrome.exe	
deepnet.exe	
firefox.exe	*nss3!CERT_VerifyCertificate*, *nss3!CERT_VerifyCert*, *nss3!CERT_VerifyCertificateNow*, *nss3!CERT_VerifyCertNow*, *nss3!CERT_VerifyCertName*
seamonkey.exe	
k-meleon.exe	
opera.exe	Функции из opera.dll

Авторы вредоносного кода использовали хэши названий процессов, вместо сравнения на основе строк для усложнения анализа.

```
call    calc_hash
mov     process_hash, eax
cmp     eax, 76379A9Ah  ; firefox.exe
ja      short loc_100029BA
jz      short loc_10002990
cmp     eax, 537B492Fh  ; iexplore.exe
ja      short loc_1000299C
jz      short loc_100029E4
cmp     eax, 2771AA06h  ; maxthon.exe
jz      short loc_100029E4
cmp     eax, 30B15DB3h  ; seamonkey.exe
jz      short loc_10002990
cmp     eax, 532A495Fh  ; k-meleon.exe
jnz     short loc_100029E9
```

Скриншот ниже показывает код перехвата *CertVerifyCertificateChainPolicy*.



Модуль **httphk** отвечает за анализ данных протокола HTTP. Когда происходит вызов callback-функции этого модуля, он выполняет разбор заголовков HTTP, данных и заполняет свои внутренние структуры данных. В дальнейшем, эта структура будет доступна через модуль **httpi**. httphk предоставляет две callback-функции для вызова httpi: httpi_request_callback и httpi_response_callback.

Модуль **httpi** выполняет модификацию данных HTTP протокола в соответствии с настройками файла конфигурации. Когда он вызывает httpi_request_callback, выполняются следующие действия.

- Захват видео и скриншотов рабочего стола. Вредоносный код читает данные конфигурационного файла и проверяет запрашиваемый URL. В случае совпадение происходит захват видео или создаются скриншоты.
- Извлечение данных форм (form grabbing). Вредоносный код проверяет параметр content-type в заголовке POST-запроса HTTPS протокола на совпадение с «application/x-www-form-urlencoded» или «text/plain». В случае совпадения он полагается, что происходит передача данных для login-формы. Если в файле конфигурации задан этот URL, то данные записываются в лог-файл.

При вызове httpi_response_callback троянская программа проверяет HTTP код ответа на значение 200 (OK). После этого осуществляется чтение конфигурационного файла и если для сайта заданы веб-инжекты, они вставляются в веб-страницу.

На рисунке ниже представлен расшифрованный файл конфигурации, который использовался в португальском ботнете. Можно заметить, что первая группа доменов будет игнорироваться модулем httpi. Это говорит о том, что данные аккаунтов Facebook и Google считаются для злоумышленников не такими ценными как данные аккаунтов систем онлайн-банкинга, которые следуют ниже.

Похоже, что люди, писавшие код скриптов web-inject говорят на русском, о чем свидетельствуют сопровождающие комментарии. Однако следует отметить, что сценарии могли быть написаны и совсем другими людьми, отличными от авторов Win32/Spy.Hesperbot. Скрипты web-inject достаточно похожи у разных семейств вредоносных программ и их использование является довольно обычным делом у киберпреступников.

Мобильный компонент

Hesperbot не является первым подобным вредоносным ПО, которое использует мобильные компоненты (ZitMo или SpitMo) для обхода системы подтверждения аутентификации на основе SMS-сообщений (Mobile Transaction Authentication Number). Подобная система

подтверждения банковской транзакции широко используется во многих странах Европы и в России.

Hesperbot осуществляет внедрение специальных JS-скриптов в веб-страницы, через которые пользователю предлагается установить приложение для мобильного телефона. Скомпрометированному пользователю предлагается специальный список моделей телефонов и после ввода номера телефона, пользователю передается ссылка на скачивание вредоносного мобильного приложения. Приложение существует в вариантах для Symbian, Android и Blackberry.

```
switch (INJ.phone_os_type) {
    case 'android':
        jq('#inj_activate_os').text('Android OS');
        break;
    case 'symbian':
        jq('#inj_activate_os').text('Symbian OS');
        break;
    case 'blackberry':
        jq('#inj_activate_os').text('Blackberry OS');
        break;
}
```

Нам удалось проанализировать вредоносные компоненты для платформ Symbian и Android. Они обладают схожими возможностями. Во-первых, это «процедура активации» (activation procedure). Специальный веб-инжект на зараженном компьютере генерирует случайный «номер активации», который отображается пользователю. Пользователю нужно будет продублировать этот номер на своем устройстве, когда такое действие будет предложено мобильным приложением. Далее мобильное приложение отображает «код ответа», который рассчитывается через код активации. Этот код пользователь должен ввести обратно в веб-страницу для подтверждения. Вредоносный скрипт снабжен тем же алгоритмом для расчета кода, что и мобильный. Таким образом, через имитацию доверенного приложения на мобильном устройстве пользователя и используя поддельную веб-страницу на его компьютере, злоумышленники могут обойти систему подтверждения кодов для проведения операций, связанных с онлайн-банкингом.

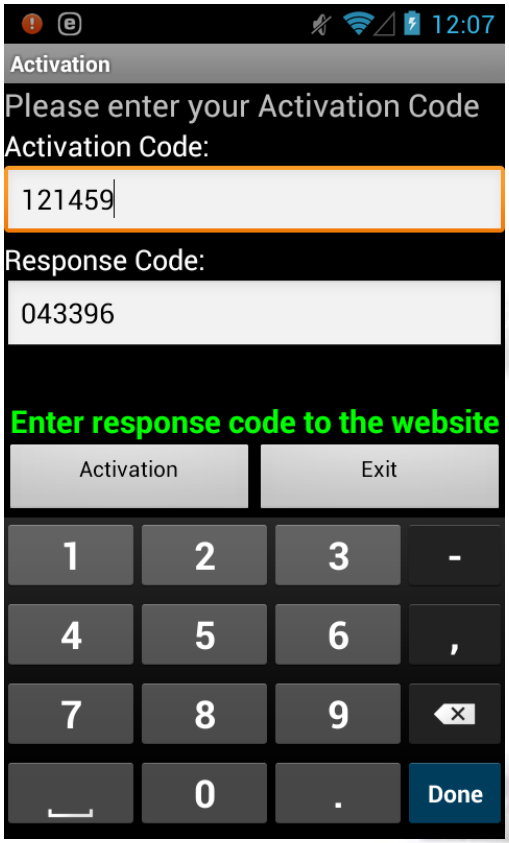


Рис. Скриншот вредоносного компонента для Android [**Android/Spy.Hesperbot.A**].

Это вредоносное мобильное приложение регистрирует специальный сервис, который ожидает входящих SMS-сообщений и передает их на номер злоумышленника. Таким образом, злоумышленник получит настоящий код для проведения транзакции или иной банковской операции.

Антивирусные приложения ESET обнаруживают мобильный вредоносный код как **Android/Spy.Hesperbot.A** и **SymbOS9/Spy.Hesperbot.A**.