



NetRipper – Smart traffic sniffing for penetration testers

Ionut Popescu – Senior Application Security Engineer @ 1&1 Romania



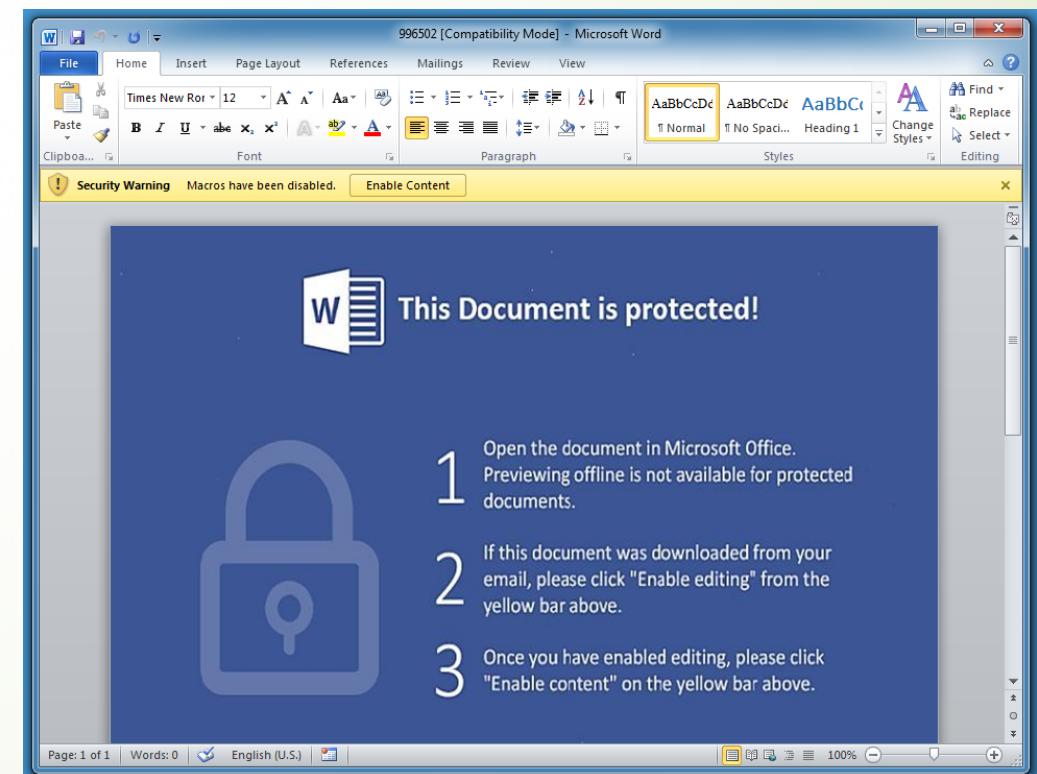
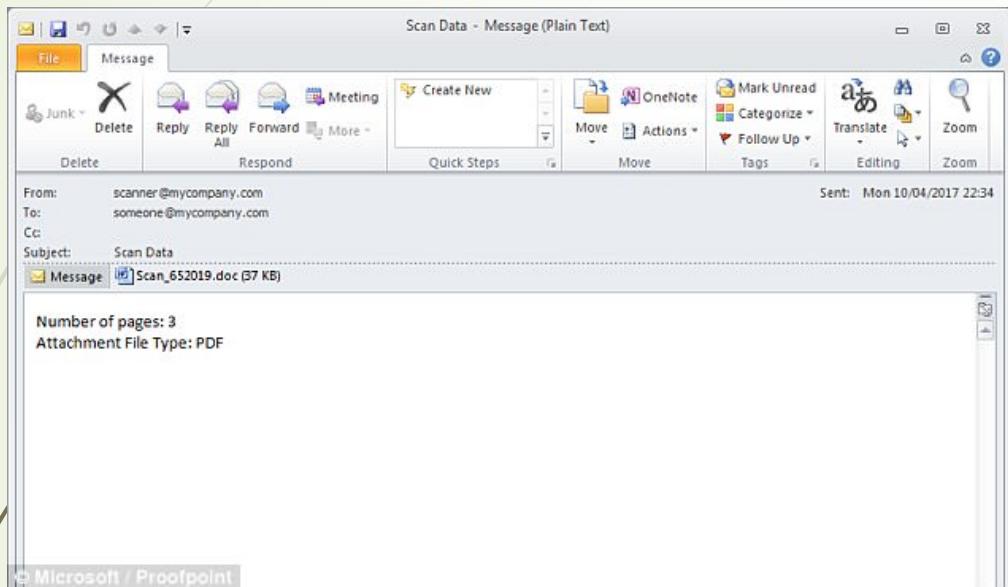
About me

- ▶ Blogger @ <https://nytrosecurity.com/>
- ▶ GitHub @ <https://github.com/NytroRST>
- ▶ Twitter @ <https://twitter.com/NytroRST>
- ▶ Admin @ <https://rstforums.com/forum/>

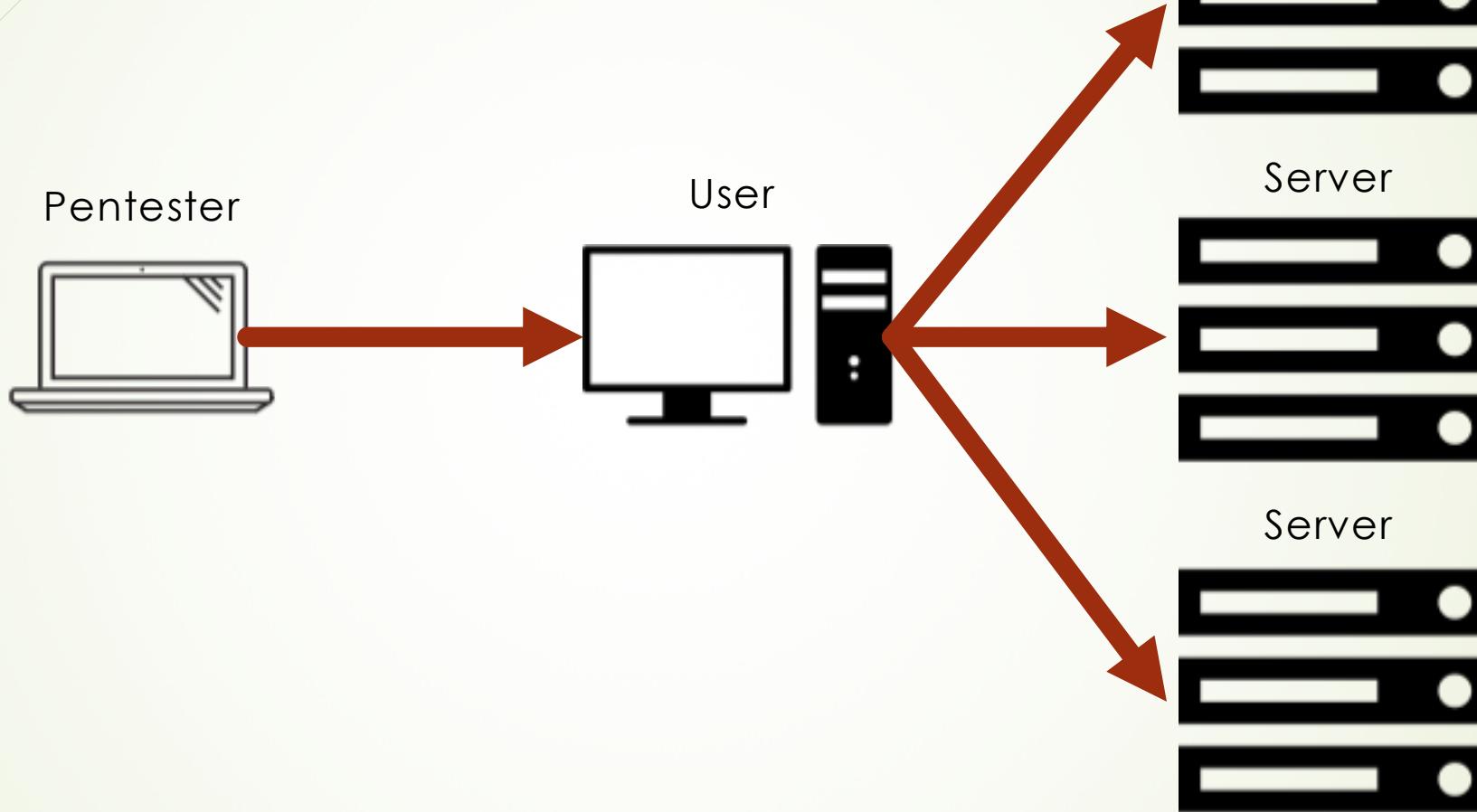
NetRipper

- ▶ Personal project
- ▶ Released at Defcon 23 (2015)
- ▶ Presented at BlackHat Asia Arsenal (2018)
- ▶ For penetration testers
- ▶ For anyone

Getting access to a workstation



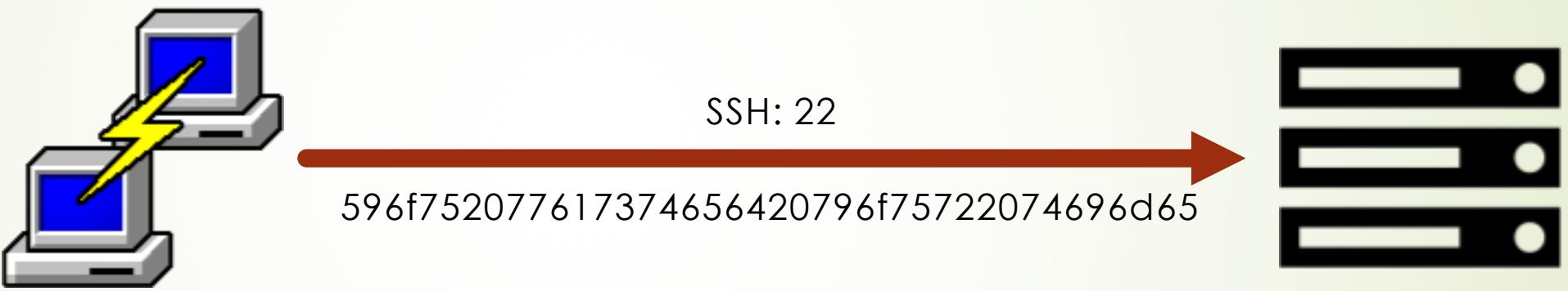
What's next?



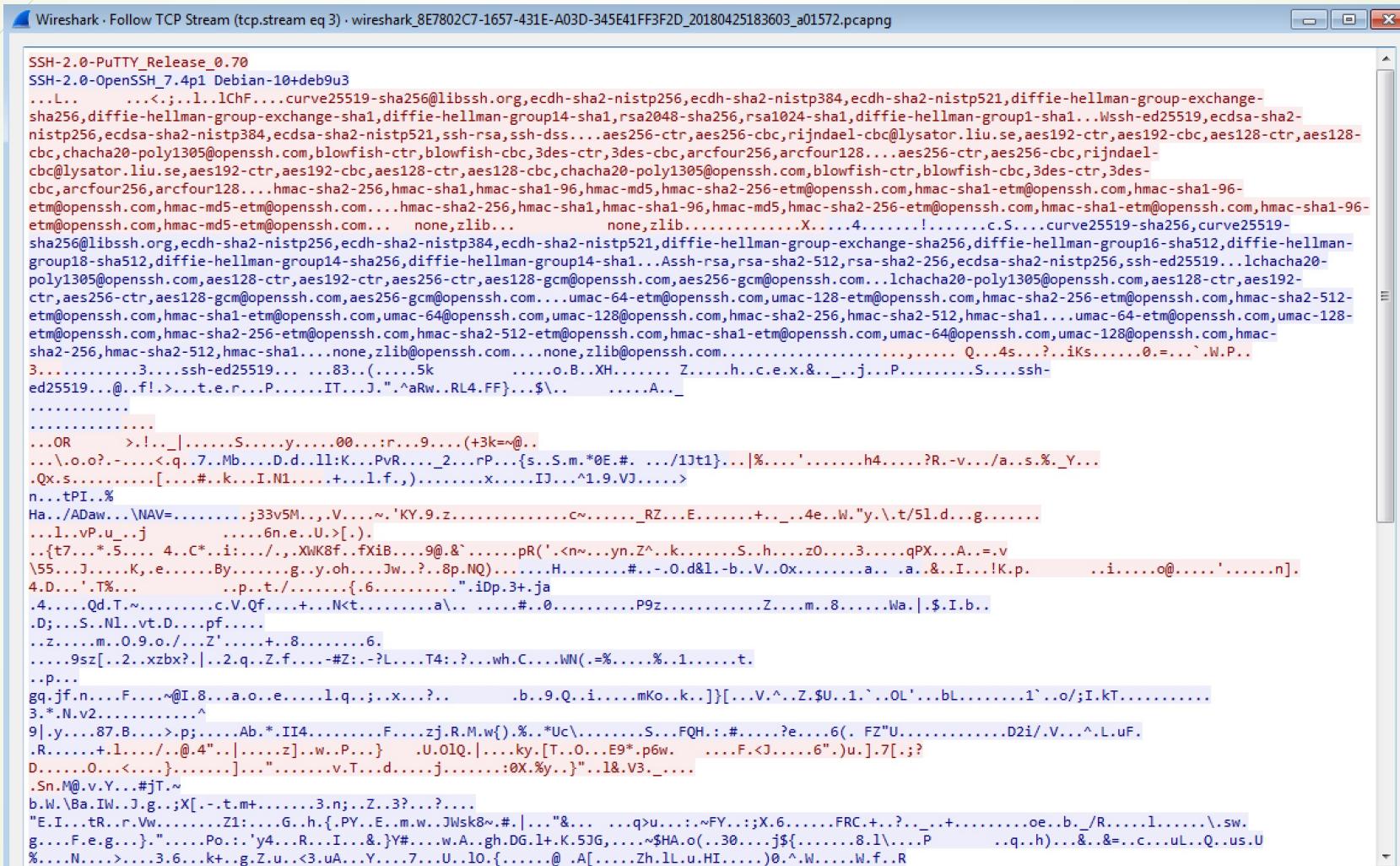
How to connect to servers?



Connection example

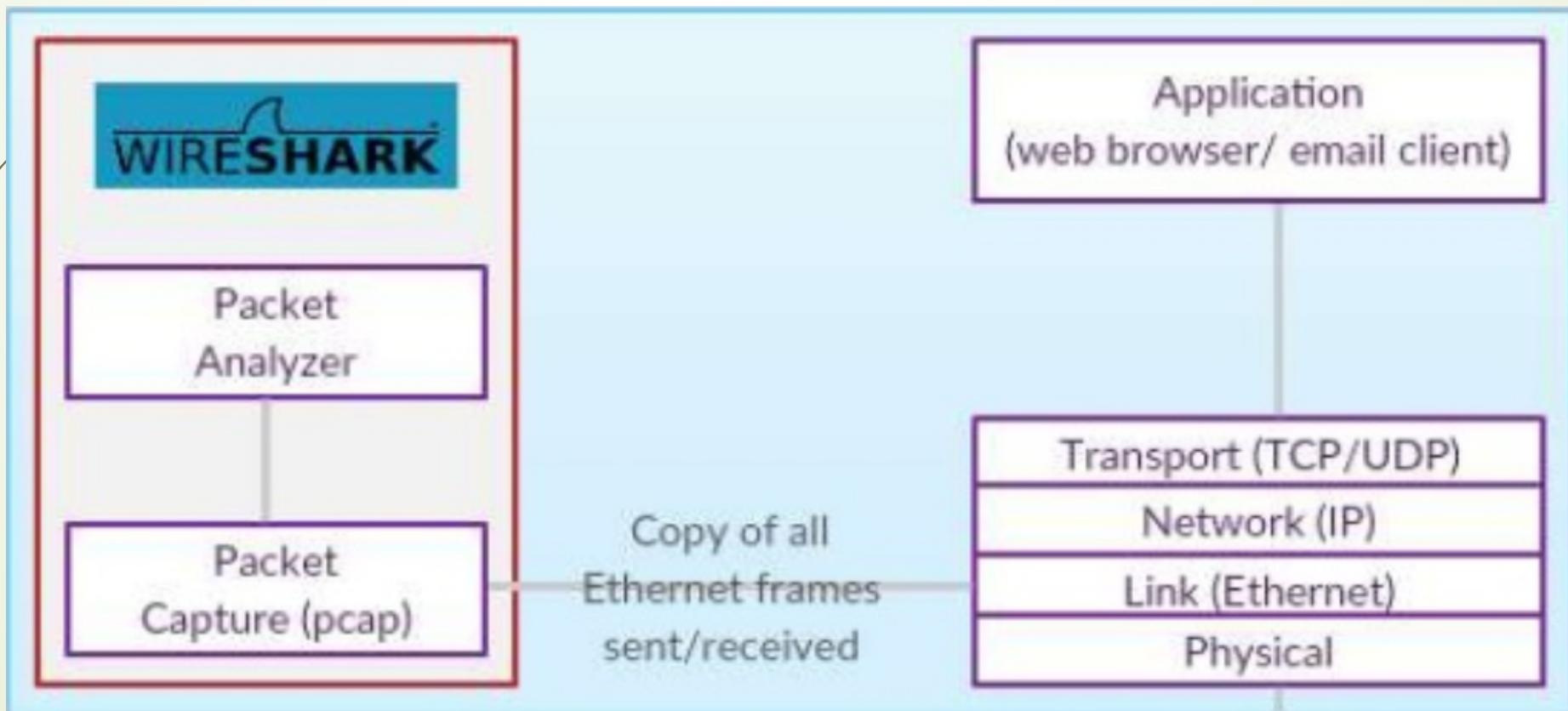


Traffic sniffing

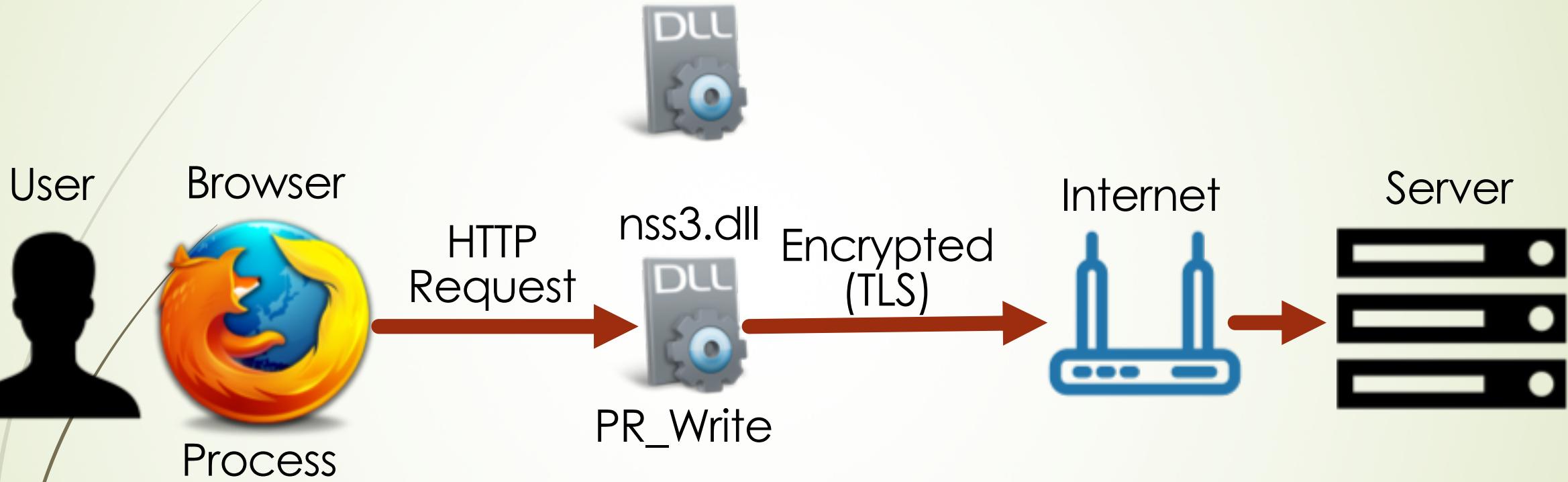


How a sniffer works

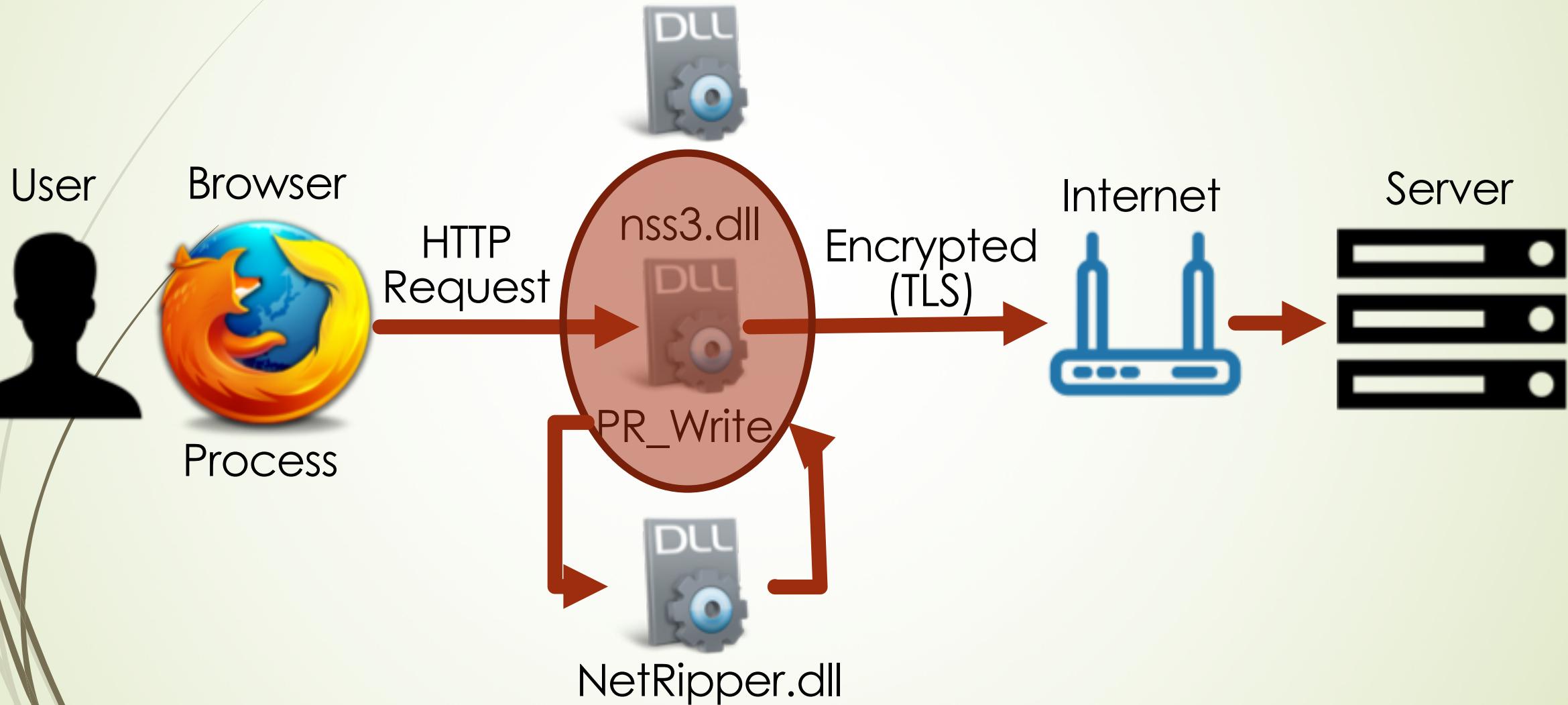
- ▶ It requires administrative privileges
- ▶ Useless for encrypted data (e.g. HTTPS, SSH)



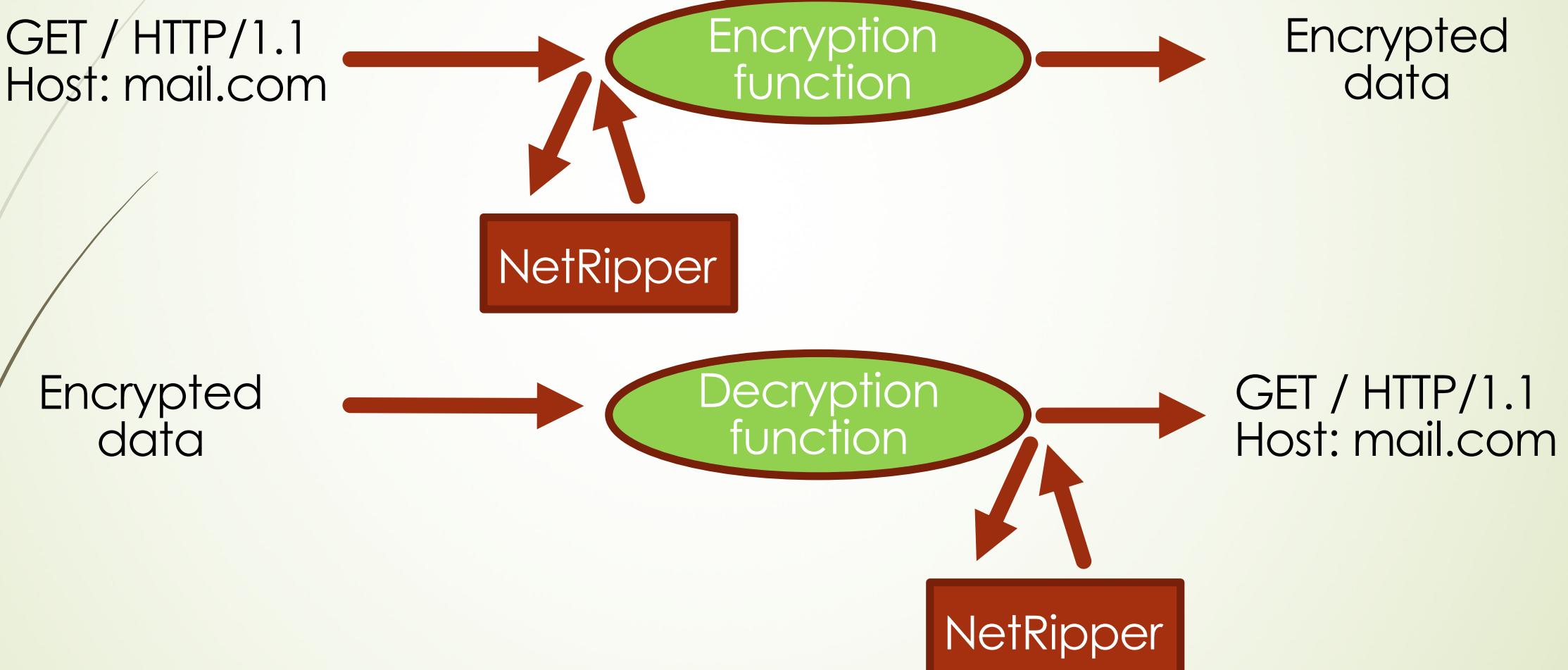
How a browser works



How NetRipper works



API Hooking



NetRipper components

- ▶ NetRipper.dll – Main component (API hooking)
- ▶ NetRipper.exe – DLL configurator and injector
- ▶ netripper.rb – Metasploit module of DLL configurator and injector

NetRipper plugins

- ▶ PlainText – Save only plaintext data
- ▶ DataLimit – Limit „packet“ size
- ▶ StringFinder – Find strings



What's new?

- ▶ Cross-compilation on Linux
- ▶ Support for PCAP files

Cross-compilation on Linux

- ▶ Requires mingw-w64
- ▶ Compiled DLLs are big
- ▶ Has limitations
- ▶ Will be improved



PCAP files

- ▶ Can easily follow requests and responses
- ▶ Can be used with Wireshark (supports multiple protocols)
- ▶ Can be used with other tools supporting PCAP files
- ▶ Can get IP addresses and TCP ports (limited)
- ▶ Will be improved



Where is the problem?

- ▶ Some applications are statically linked (no exported functions, reverse-engineering required)
- ▶ Support has to be added for each of them
- ▶ Examples: Putty, Google Chrome

Google Chrome

Immunity Debugger interface showing assembly, registers, and memory dump panes for Google Chrome.

Assembly pane (left):

```
68902F5E 55          push ebp
68902F5F 53          mov  ebp,esp
68902F61 57          push ebx
68902F62 57          push edi
68902F63 56          push esi
68902F64 83 EC 20    sub  esp,20
68902F67 A1 98 EC 44 6C  mov  eax,dword ptr ds:[6C44]
68902F68 89 CE        mov  esi,ecx
68902F69 31 E8        xor  eax,ebp
68902F6A 89 45 F0    mov  dword ptr ss:[ebp-10],eax
68902F6B 88 46 18    mov  eax,dword ptr ds:[esi+18]
68902F6C C7 80 98 00 00 00 01  mov  dword ptr ds:[eax+98],01
68902F6D E8 9C CE 3E FE  call chrome.69CEFE21
68902F6E E8 45 06 6A FE  call chrome.69FA35CF
68902F6F 83 7E 14 00    cmp  dword ptr ds:[esi+14],je chrome.68902FAC
68902F70 88 46 18    mov  eax,dword ptr ds:[esi+18]
68902F71 83 B8 84 00 00 00 02  cmp  dword ptr ds:[eax+84],jne chrome.68902FCC
68902F72 75 30        push  dword ptr ds:[eax+8C]
68902F73 FF B0 8C 00 00 00 00  call chrome.6A90DC67
68902F74 E8 C0 AC 00 FF  add   esp,4
68902F75 83 C4 04    add   esp,4
68902F76 EB 18        jmp   chrome.68902FC7
68902F77 68 C2 03 00 00 00 00  push  3C2
68902F78 68 24 24 35 6C 00 00 00  push  chrome.6C352424
68902F79 68 E2 00 00 00 00 00 00  push  E2
68902F7A 6A 00        push  0
68902F7B 6A 10        push  10
68902F7C E8 D4 A9 00 FF  call  chrome.6A90D998
68902F7D 83 C4 14    add   esp,4
68902F7E 31 FF        xor   edi,edi
68902F7F 45           add   esp,4
```

Registers pane (right):

EAX	0C71FCE4
EBX	00000000
ECX	00000004
EDX	00000000
EBP	0C71FD0C
ESP	0C71FCA0
ESI	00000BEC
EDI	0C71FCE8

EIP = 77D6F901

Memory dump pane (bottom):

06DEF274	6BF4D000/mojo/public/cpp/system/simple
06DEF278	0000006E	
06DEF27C	69CF019E	return to chrome.69CF019E from chro
06DEF280	00000013	
06DEF284	00000017	
06DEF288	EC6C4EDE	
06DEF28C	06DEF2C0	
06DEF290	0D4COA98	
06DEF294	00000000	
06DEF298	00000000	
06DEF29C	06DEF2RC	

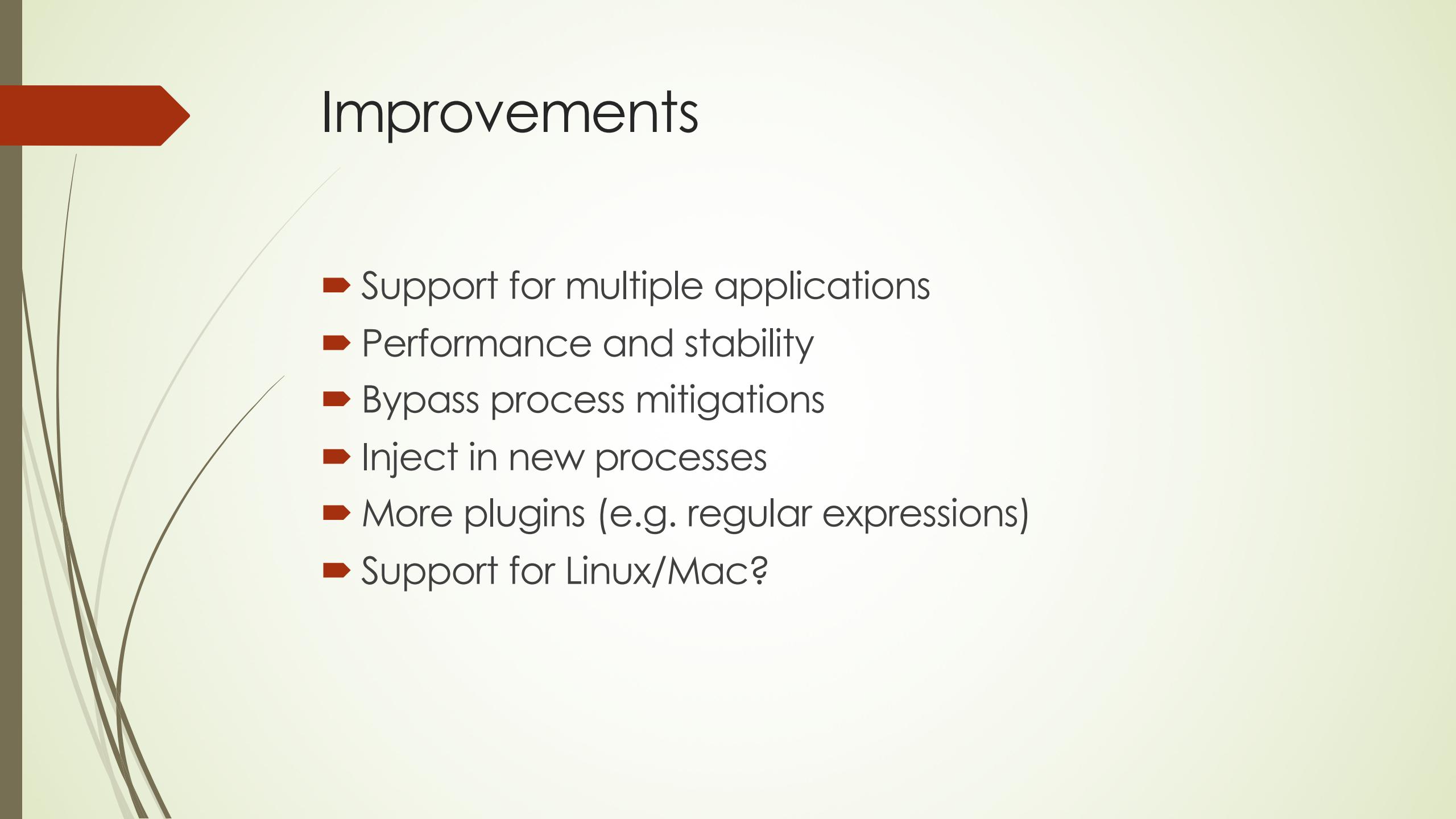


Use cases

- ▶ Penetration testers
- ▶ Bug bounty hunters
- ▶ Attackers
- ▶ Any other users



DEMO



Improvements

- ▶ Support for multiple applications
- ▶ Performance and stability
- ▶ Bypass process mitigations
- ▶ Inject in new processes
- ▶ More plugins (e.g. regular expressions)
- ▶ Support for Linux/Mac?

Conclusion

- ▶ Open-source tool for Windows
- ▶ Captures traffic before encryption and after decryption
- ▶ Supports multiple applications
- ▶ Easy to use
- ▶ It can be improved



Questions?

<https://github.com/NytroRST/NetRipper>

ionut.popescu@outlook.com