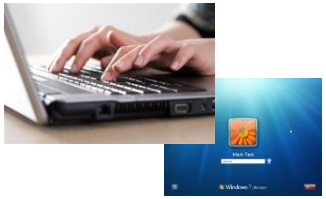
Post-Exploitation with WCE v1.2

Pass-the-Hash. Pass-the-ticket & more…

Date: 0**1-07-2011**

Author:
Hernan Ochoa
<hernan@ampliasecurity.com>

Windows Authentication



cgarcia
pwd1

h1 = LMHash("pwd1")
h2 = NTHash("pwd1")

cmp([h1,h2], SAM)

SAM Database

User	Hash LM	Hash NT
cgarcia	A6BCD..	B0FD1..
ajuarez	B90DF..	CCDF..
…	…	…

Network

- **The cleartext password is not sent**
- **The NTLM protocol is used**

NTLM Authentication

lmhash = LMHash("pwd1")
nthash = NTHash("pwd1")

Init connection

Responds *C* = challenge random

Sends *cgarcia*, *R*

cgarcia
pwd1

R = f(*lmhash*/*nthash*, *C*)

SAM Database

User	Hash LM	Hash NT
cgarcia	A6BCD..	B0FD1..
ajuarez	B90DF..	CCDF..
…	…	…

R' = f(SAM[lmhash/nthash], C)
R' == R => Access Granted
R' <> R => Access Denied

pre-*Pass-the-Hash Attacks*

- After compromising a Windows box…
  - ‘ Dump’ the SAM
    - *pwdump3/3e/4/5/6/7,fgdump*,etc
      - Administrator:500:010203040506070809 0A0B0C0D0E0F10:1020304050607080 90A0B0C0D0E0F10
  - Crack/Brute-Force hashes to obtain cleartext password
    - Takes time.. (e.g.: pentest time is limited)
    - No guarantee the password will be obtained
    - Rainbow tables were not widely used
      - Less computing power, storage, etc
      - Still, not the answer to everything

Pass-The-Hash Technique

Published by Paul Ashton in 1997

http://www.securityfocus.com/bid/233/discuss

amplia security

Pass-the-hash Technique

lmhash = LMHash("pwd1")
nthash = NTHash("pwd1")

Init connection

Responds *C* = challenge random

Sends *cgarcia*, *R*

cgarcia
pwd1

R = f(***lmhash/nthash***, *C*)

SAM Database

User	Hash LM	Hash NT
cgarcia	A6BCD..	B0FD1..
ajuarez	B90DF..	CCDF..
…	…	…

R' = f(SAM[**lmhash/nthash**], C)
R' == R => Access Granted
R' <> R => Access Denied

- Cleartext password is not needed for NTLM auth
- Only lmhash/nthash are needed
- No need to crack/brute-force
- Just use the hashes directly

Pass-the-hash: 'exploitation'

- *Modified Smbclient* (SAMBA)
  - *smbclient //192.168.1.120/c$ -U Administrator -p 4ECC0E7568976B7EAAD3B435B51404EE:551E3B3215FFD87F5E037B3E3523D5F6*
- *Samba-TNG*
- Many 3rd-party *SMB+NTLM* stacks
  - *Python*, *Ruby*, *Java*, etc

3rd-party *SMB+NTLM stacks*: Limitations

- Limited and partial functionality
- Always running behind Windows
  - New functionality has to be implemented, some by reverse engineering
    - Complex, requires time and effort
- Cannot use native Windows applications
  - They ask for username and cleartext password, not hashes…

Enter, *Windows Credentials Editor...*

What is WCE?

- Tool to manipulate Windows logon sessions
  - Add, list, delete, modify
  - Obtain credentials associated with logon sessions
  - **Pass-the-hash (NTLM)**
  - Pass-the-ticket (Kerberos)

Pre-WCE/Pass-the-hash attacks

- Without WCE..
  - Crack/Brute-force hashes to obtain cleartext password
  - Crack/Brute-force ´encrypted´ hashes (C,R->NTLM) to obtain cleartext password
  - Use 3rd-party SMB+NTLM stacks, w/limited and partial functionality
  - More difficult/not possible to do pass-the-hash while pivoting among Windows boxes

Post-WCE/Pass-the-hash attacks

- With WCE…
  - Do *Pass-the-hash* directly with the hashes
    - No need to attempt to crack/brute-force hashes
    - Will be able to use them even if you cannot crack them
  - 3rd-party SMB+NTLM stacks problems eliminated
  - Easier to do *Pass-the-hash* while pivoting among Windows boxes

Demo #1
WCE: Pass-the-hash

WCE: ‘Steal’ Credentials from memory

- New ´attack´ implemented by WCE
  - **It is not** *Pass-the-hash*, it´s another technique..
  - Sometimes the two are confused, but they are not the same..

- Allows you to obtain usernames and NTLM hashes stored in memory

WCE: ‘Steal’ Credentials from memory

- Why are they in memory?
  - NTLM auth package
  - SSO

cgarcia
pwd1

MyDomain/cgarcia
h1 = LMHash(“pwd1”)
h2 = NTHash(“pwd1”)

Network

NTLM

WCE: ‘Steal’ Credentials from memory

When are they stored in memory?

- Interactive logon sessions at the console
- Remote logon sessions via RDP
- RunAs
- Windows Services running under specific user accounts
- Windows APIs used by applications
- Etc.

WCE: ‘Steal’ Credentials from memory

When are they stored in memory?

- Interactive logon sessions at the console

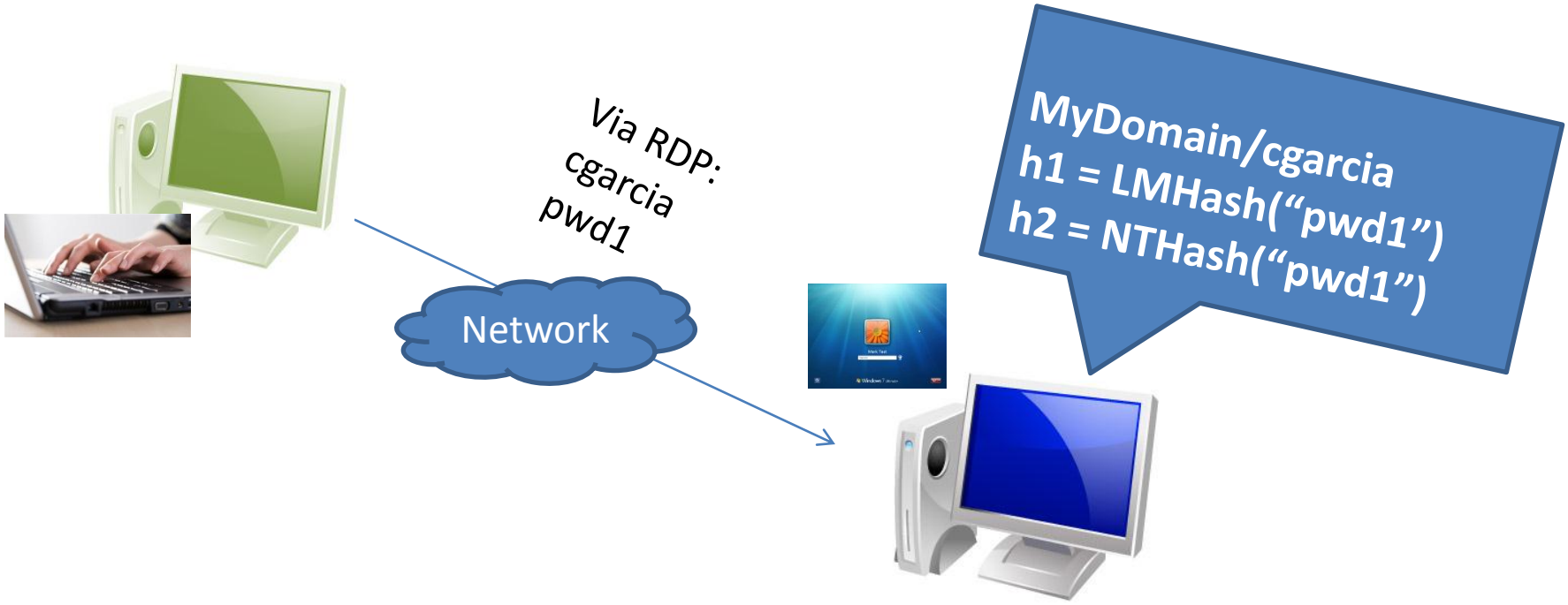
cgarcia
pwd1

MyDomain/cgarcia
h1 = LMHash(“pwd1”)
h2 = NTHash(“pwd1”)

WCE: 'Steal' Credentials from memory

When are they stored in memory?

- Remote logon sessions via RDP

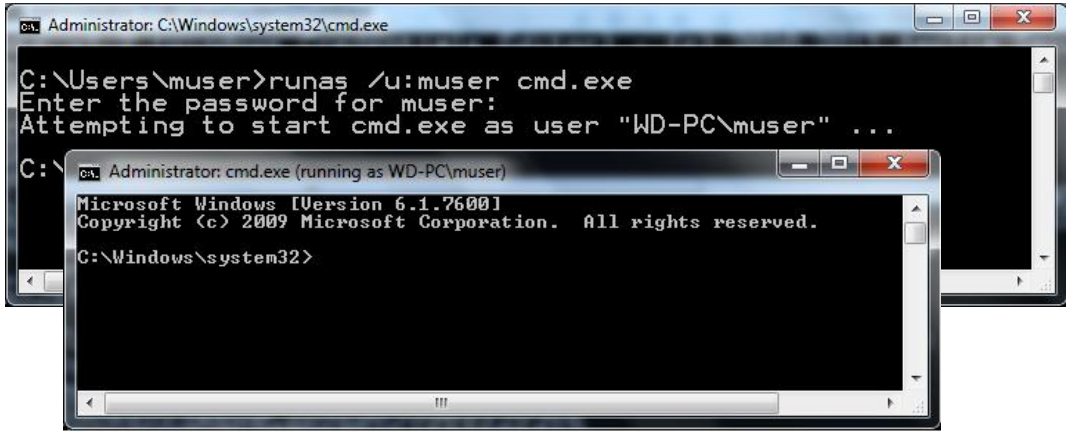


WCE: ‘Steal’ Credentials from memory

When are they stored in memory?

- RunAs

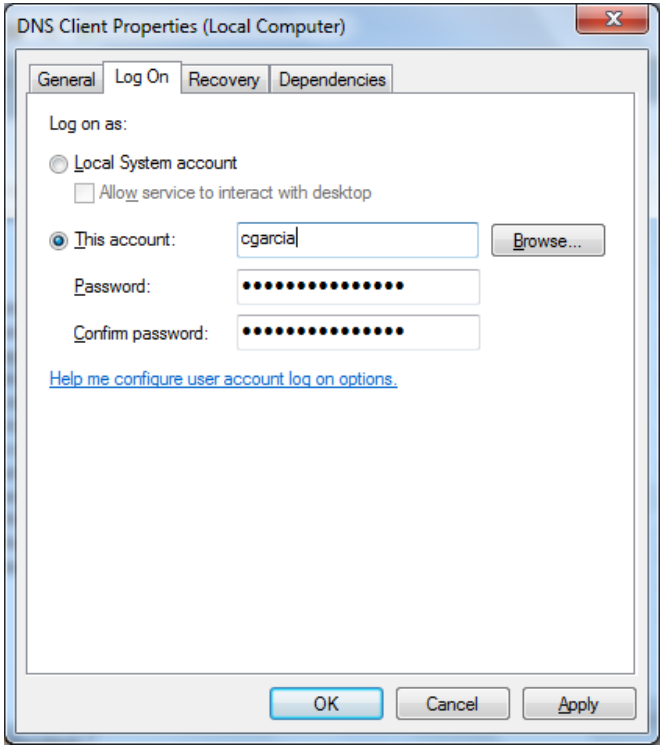
WD-PC/muser
h1 = LMHash(“pwd4”)
h2 = NTHash(“pwd4”)



WCE: 'Steal' Credentials from memory

When are they stored in memory?

- Windows Services



WCE: 'Steal' Credentials from memory

When are they stored in memory?

- Windows APIs used by applications

Example:

LogonUser Function

msdn

The **LogonUser** function attempts to log a user on to the local computer. The local computer is the computer from which **LogonUser** was called. You cannot use **LogonUser** to log on to a remote computer. You specify the user with a user name and domain and *authenticate* the user with a *plaintext* password. If the function succeeds, you receive a handle to a token that represents the logged-on user. You can then use this token handle to impersonate the specified user or, in most cases, to create a *process* that runs in the context of the specified user.

Syntax

```
BOOL LogonUser(
  __in      LPTSTR lpszUsername,
  __in_opt  LPTSTR lpszDomain,
  __in      LPTSTR lpszPassword,
  __in      DWORD dwLogonType,
  __in      DWORD dwLogonProvider,
  __out     PHANDLE phToken
);
```

MyDomain/cgarcia
h1 = LMHash("pwd1")
h2 = NTHash("pwd1")

WCE: ‘Steal’ Credentials from memory

- WCE can obtain the LM Hash..
  - By default, nowadays, Windows does not store the LM hash in the SAM
    - It is weak; “easy” to crack

WCE: ‘Steal’ Credentials from memory

- WCE can obtain the LM Hash from memory
  - Windows generates and stores the hashes in memory, including the LM hash
    - Interactive sessions
    - Others previously mentioned
  - Possible to crack it and obtain cleartext password to use in places where NTLM is not the auth method

WCE: 'Steal' Credentials from memory

Example:

The LM Hash is not in the SAM

```
Administrator:500:NO PASSWORD**********************:NO PASSWORD**********************
***:::
Guest:501:NO PASSWORD**********************:NO PASSWORD**********************:::
muser:1000:NO PASSWORD**********************:9BF617CAEFC9DFE18995B5A300174176:::
Completed.

C:\Users\muser\wce>
```

Pwdump output

The LM Hash is in memory

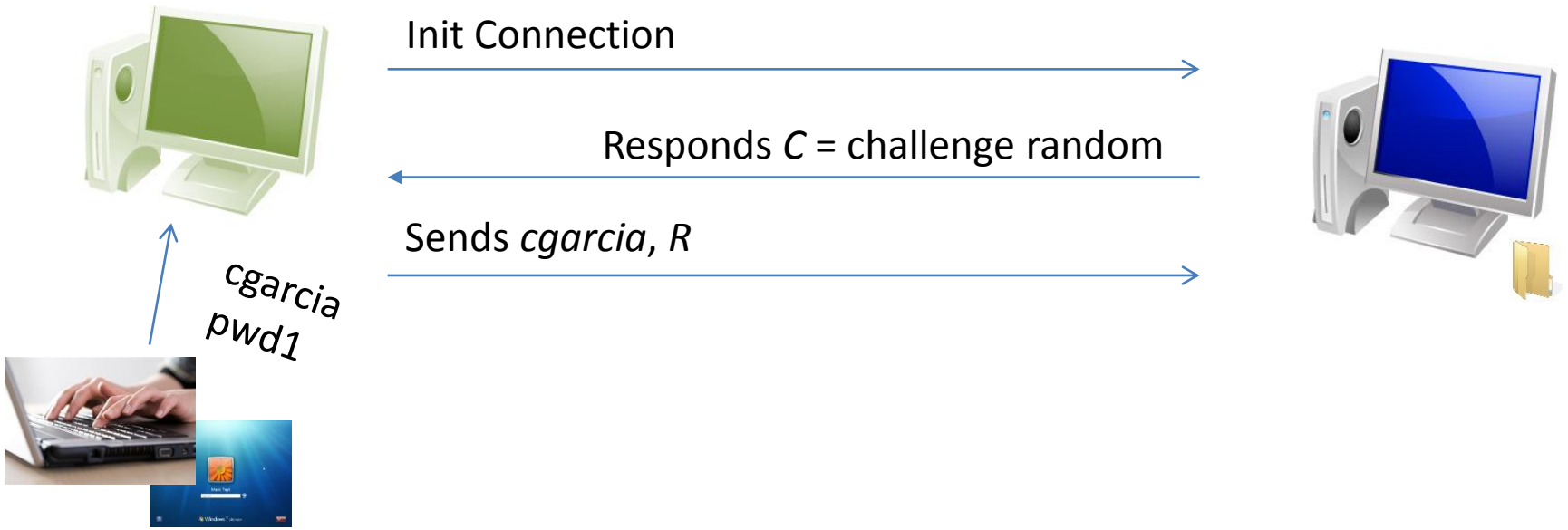
```
C:\Users\muser\wce>wce
WCE v1.2 (Windows Credentials Editor) - (c) 2010,2011 Amplia Security - by Herna
n Ochoa (hernan@ampliasecurity.com)
Use -h for help.

muser:wd-PC:A3283469F98CF766AAD3B435B51404EE:9BF617CAEFC9DFE18995B5A300174176

C:\Users\muser\wce>
```

WCE output

WCE: 'Steal' Credentials from memory

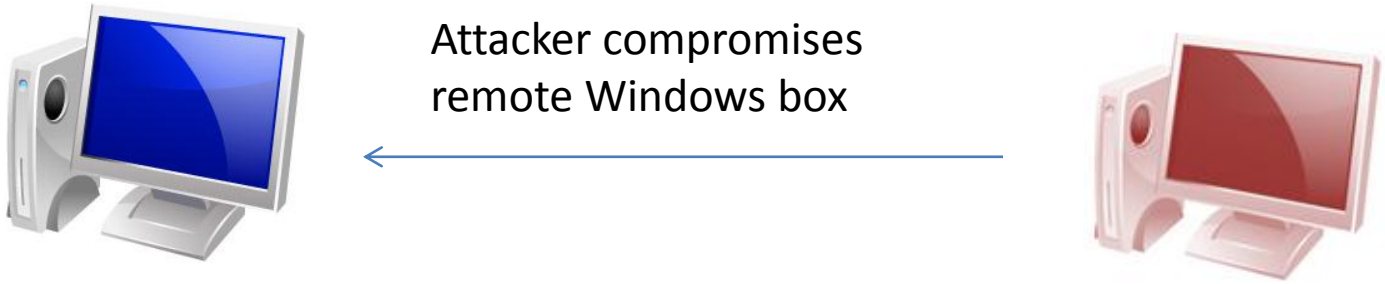
When are they **NOT** stored in memory?

- Network Logons
  - The hashes never reach the remote server



WCE: ‘Steal’ Credentials from memory

Post-explotation attack scenario

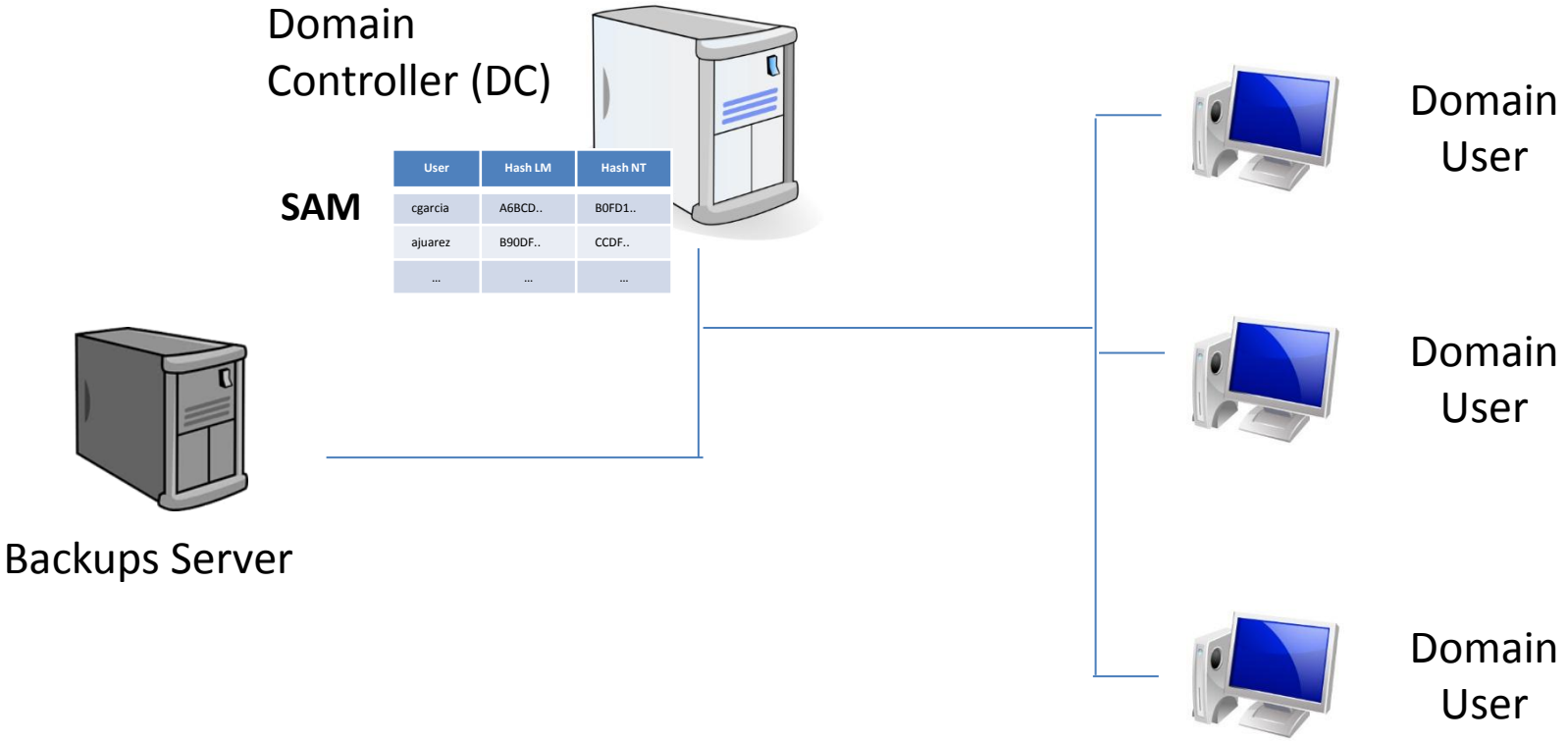


- Run WCE to obtain credentials stored in memory
- Use those hashes to do Pass-The-Hash with WCE

Demo #2
WCE: ‘Steal’ credentials from memory

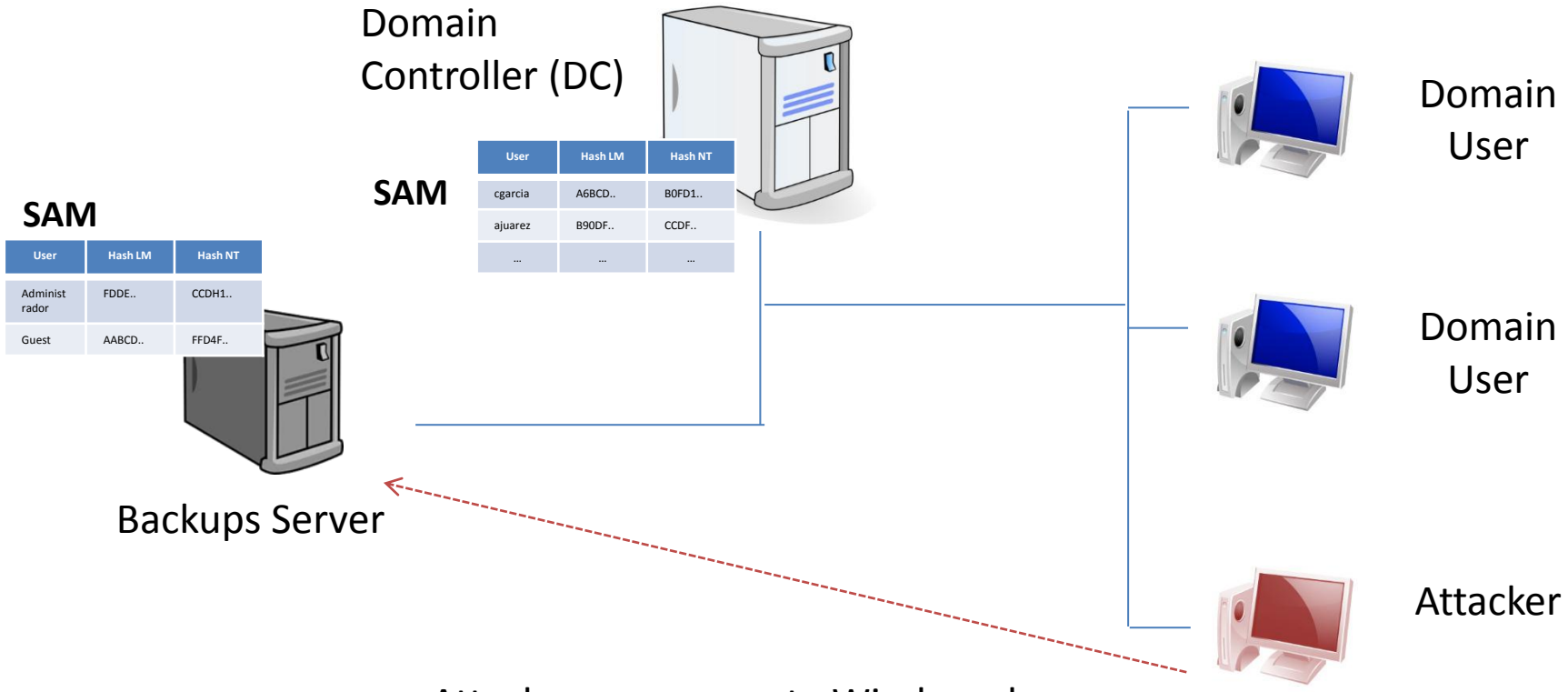
WCE: ‘Steal’ Credentials from memory

Especially interesting in Windows Domain Environments



WCE: ‘Steal’ Credentials from memory

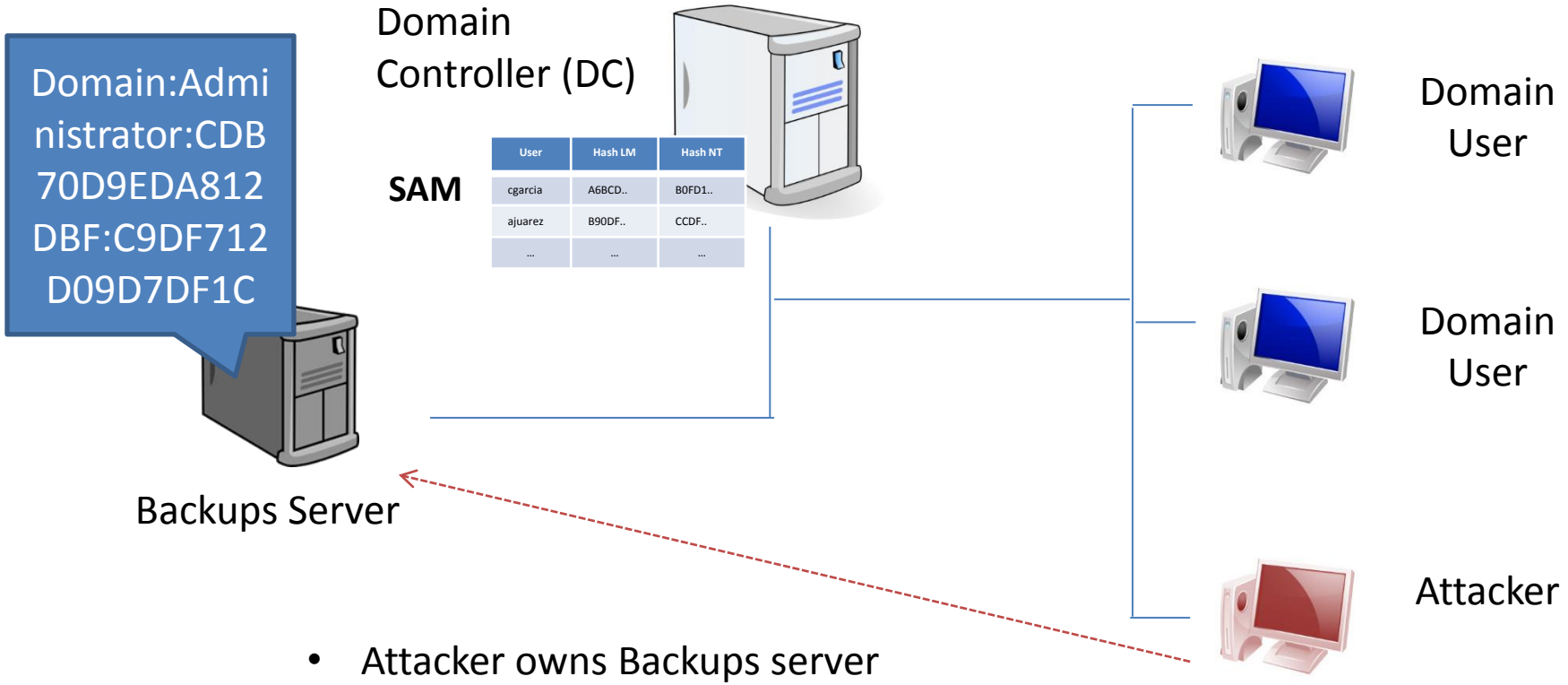
WITHOUT WCE..



- Attacker owns remote Windows box
- Only has access to local SAM..
- No Domain Users there, not very useful…

WCE: ‘Steal’ Credentials from memory

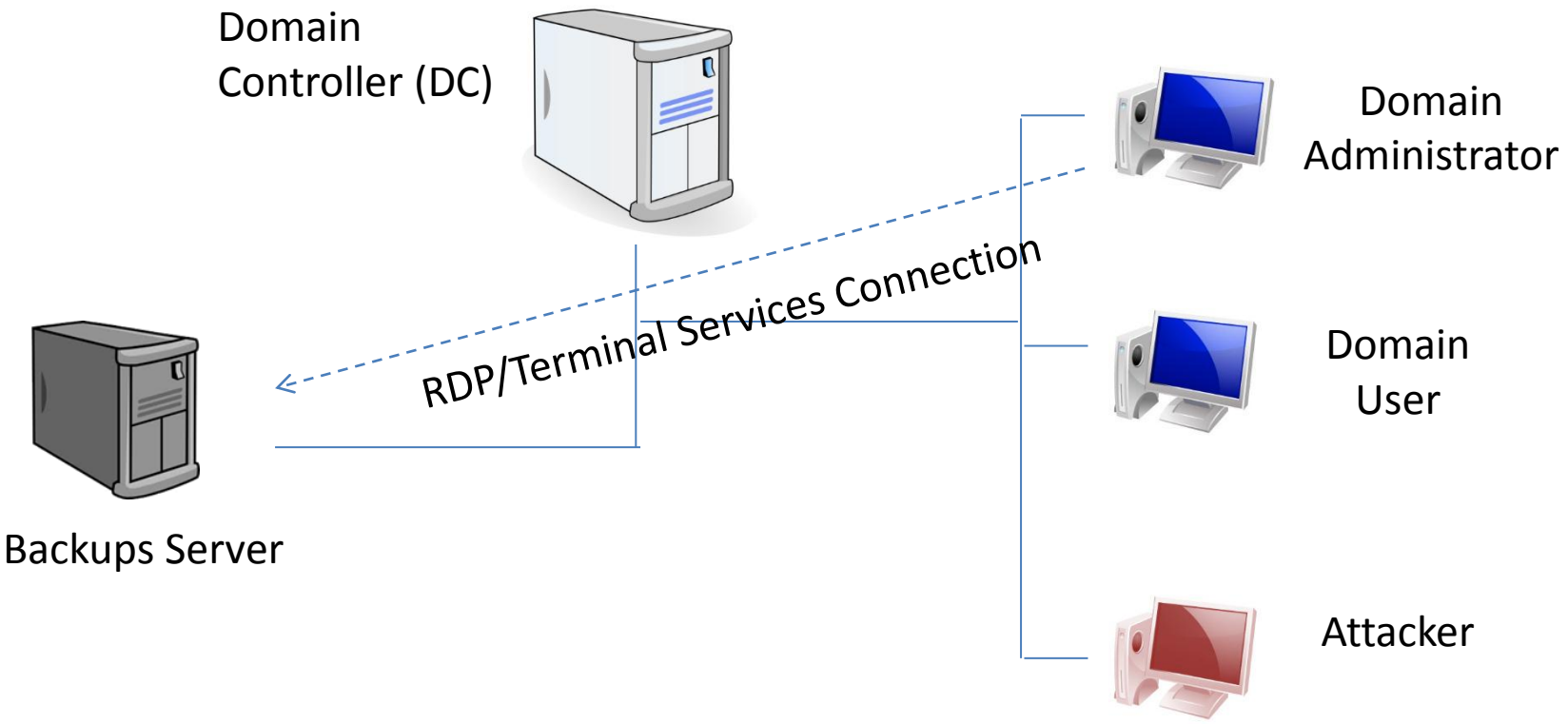
WITH WCE..



- Attacker owns Backups server
- Obtains credentials from memory..
  - Probably will find Domain Users there..
  - Can possibly compromise the whole domain!

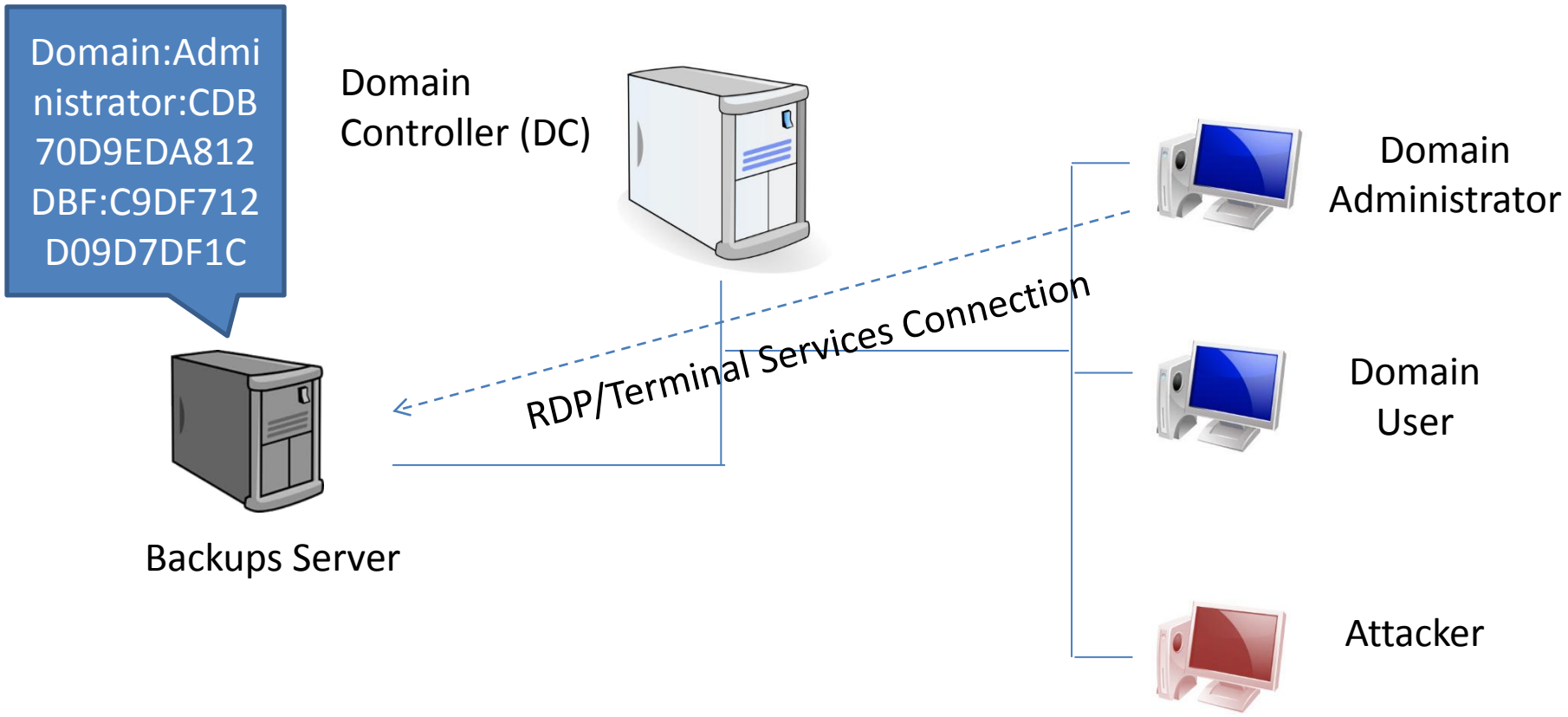
WCE: ‘Steal’ Credentials from memory

Typical Scenario



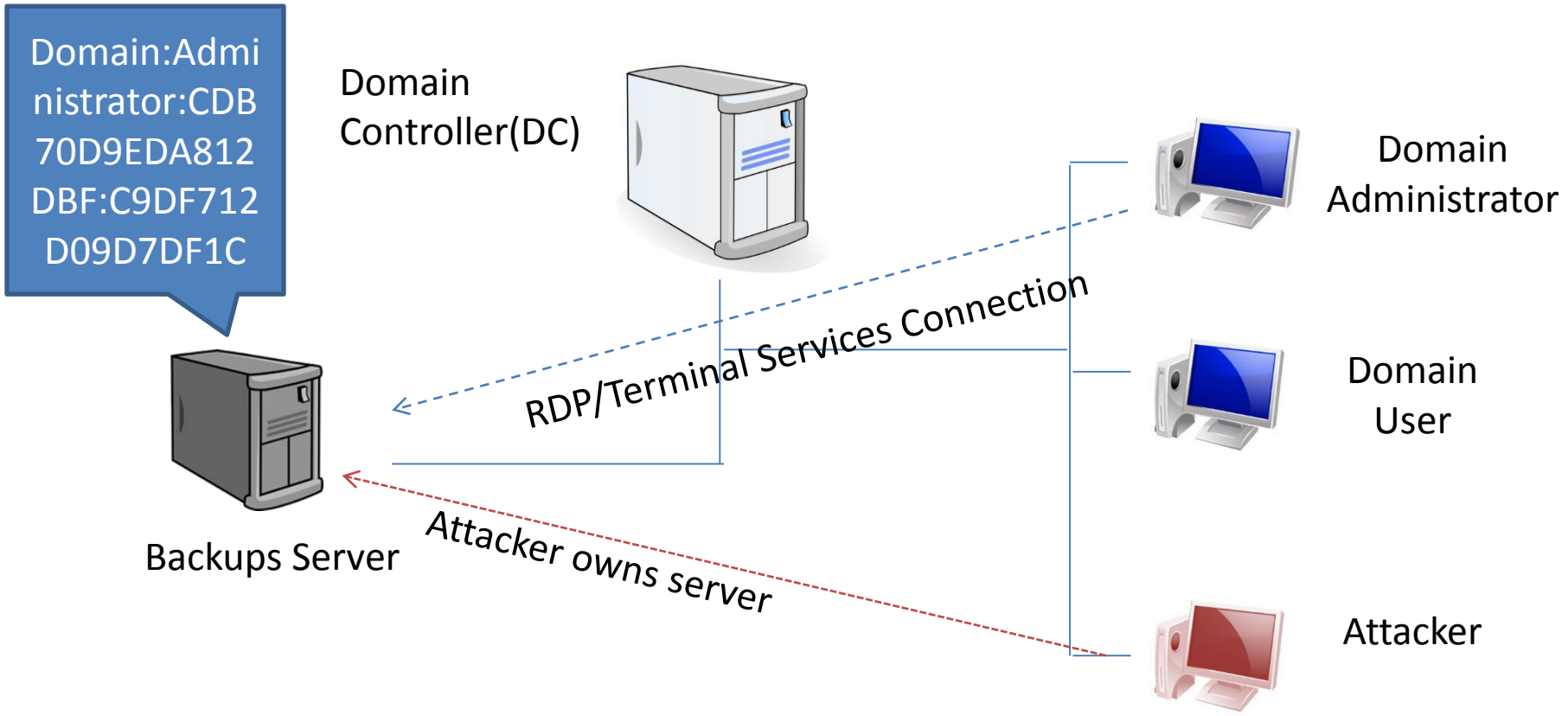
WCE: ‘Steal’ Credentials from memory

Typical Scenario



WCE: ‘Steal’ Credentials from memory

Typical Scenario



- **WCE ‘steals’ the Domain Administrator Credentials!**

WCE: ‘Steal’ Credentials from memory

RDP ´exposes´ credentials

- **When you RDP to a remote box, you leave the NTLM hashes of your password in the remote server´s memory**
  - NTLM hashes are equivalent to the cleartext password (pass-the-hash+wce)
  - So, we could say you are leaving there your password…

WCE: ‘Steal’ Credentials from memory

RDP ´exposes´ credentials

- **Even when using pass-through authentication**
  - Credentials (user+domain+hashes) are stored in Domain Controller
  - But when you use RDP, they are also left in the memory of the remote Windows box you are RDPing to!
  - Be careful where you are RDPing to…

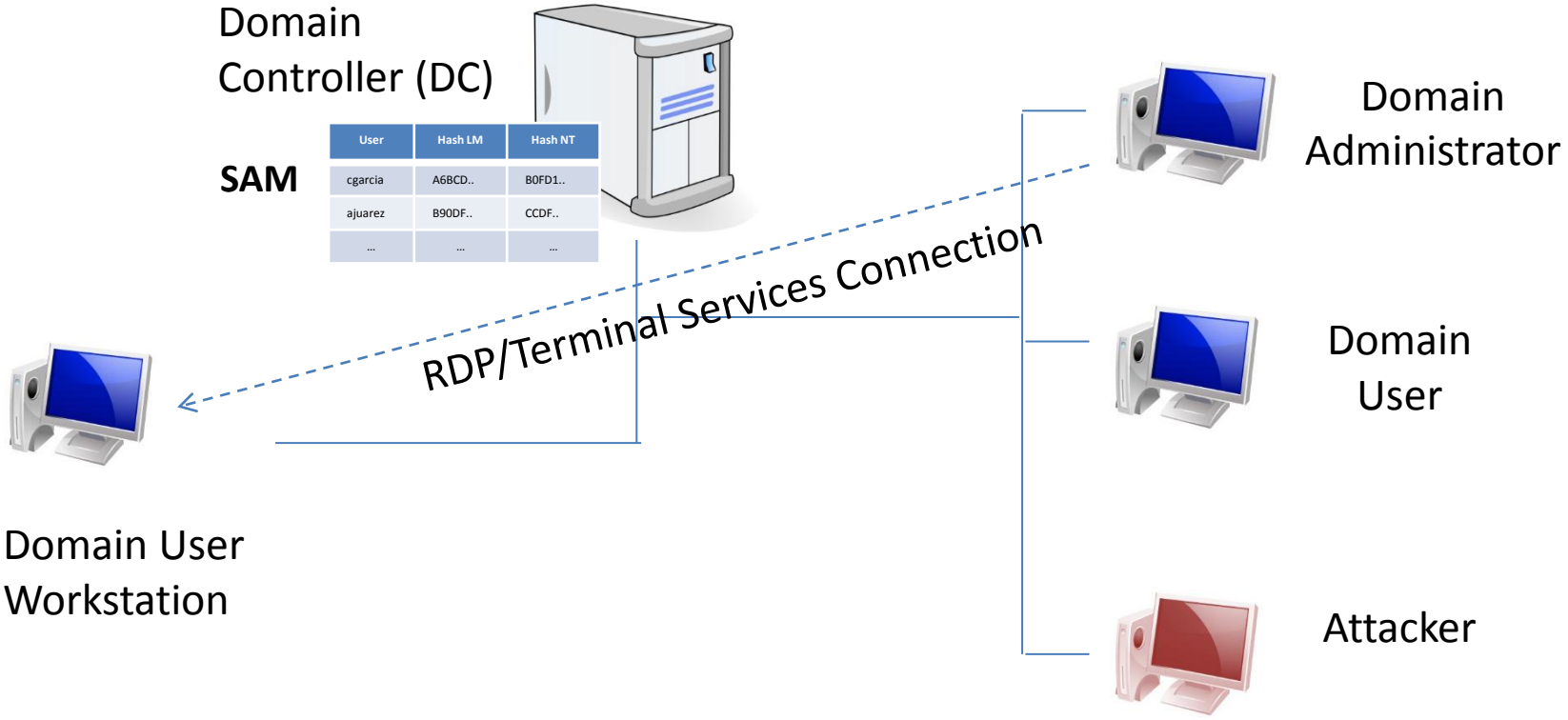
WCE: ‘Steal’ Credentials from memory

RDP ´exposes´ credentials

- Attacker on the remote Windows box can obtain credentials from memory
  - Local Administrator
  - Regular Domain User w/local admin privs
  - Attacker that owns less secured box, regular users are more vulnerable
  - Etc.

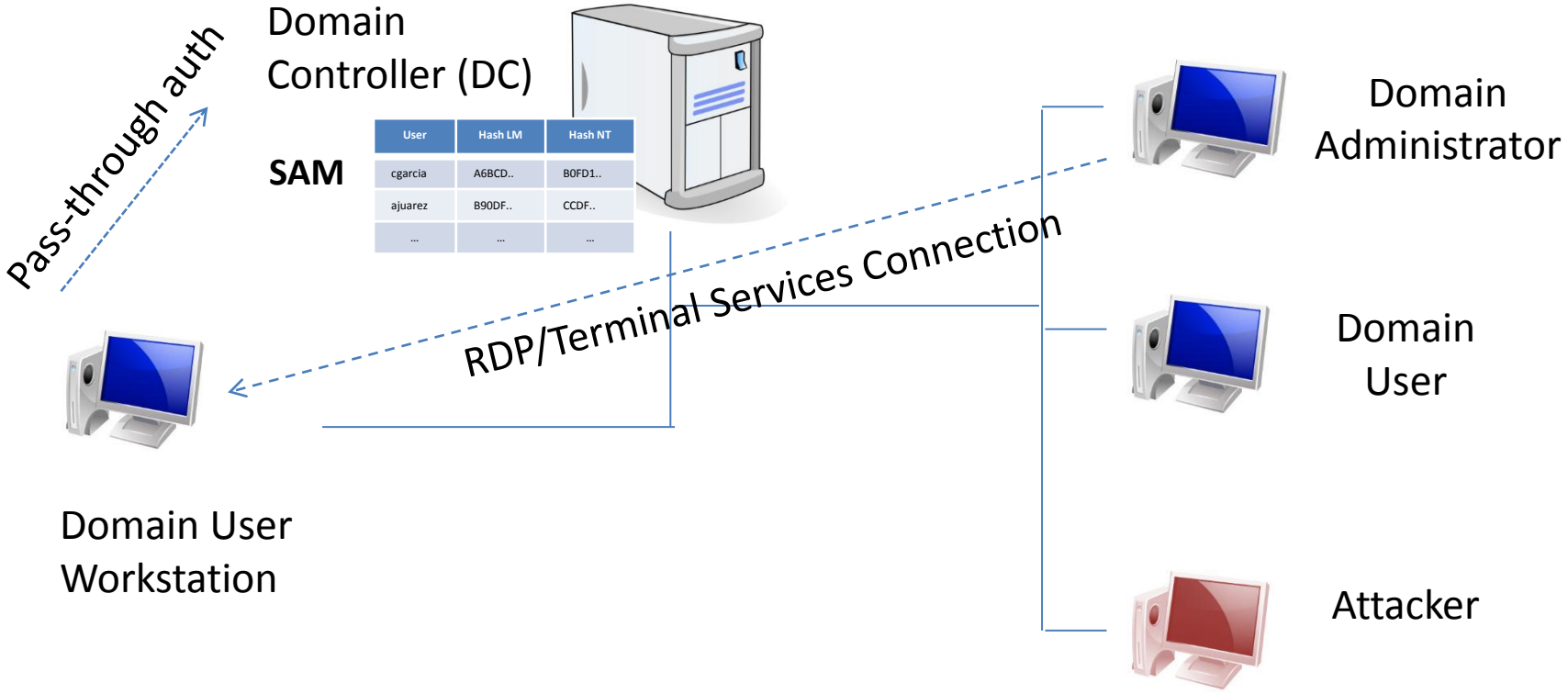
WCE: ‘Steal’ Credentials from memory

RDP ´exposes´ credentials: Example



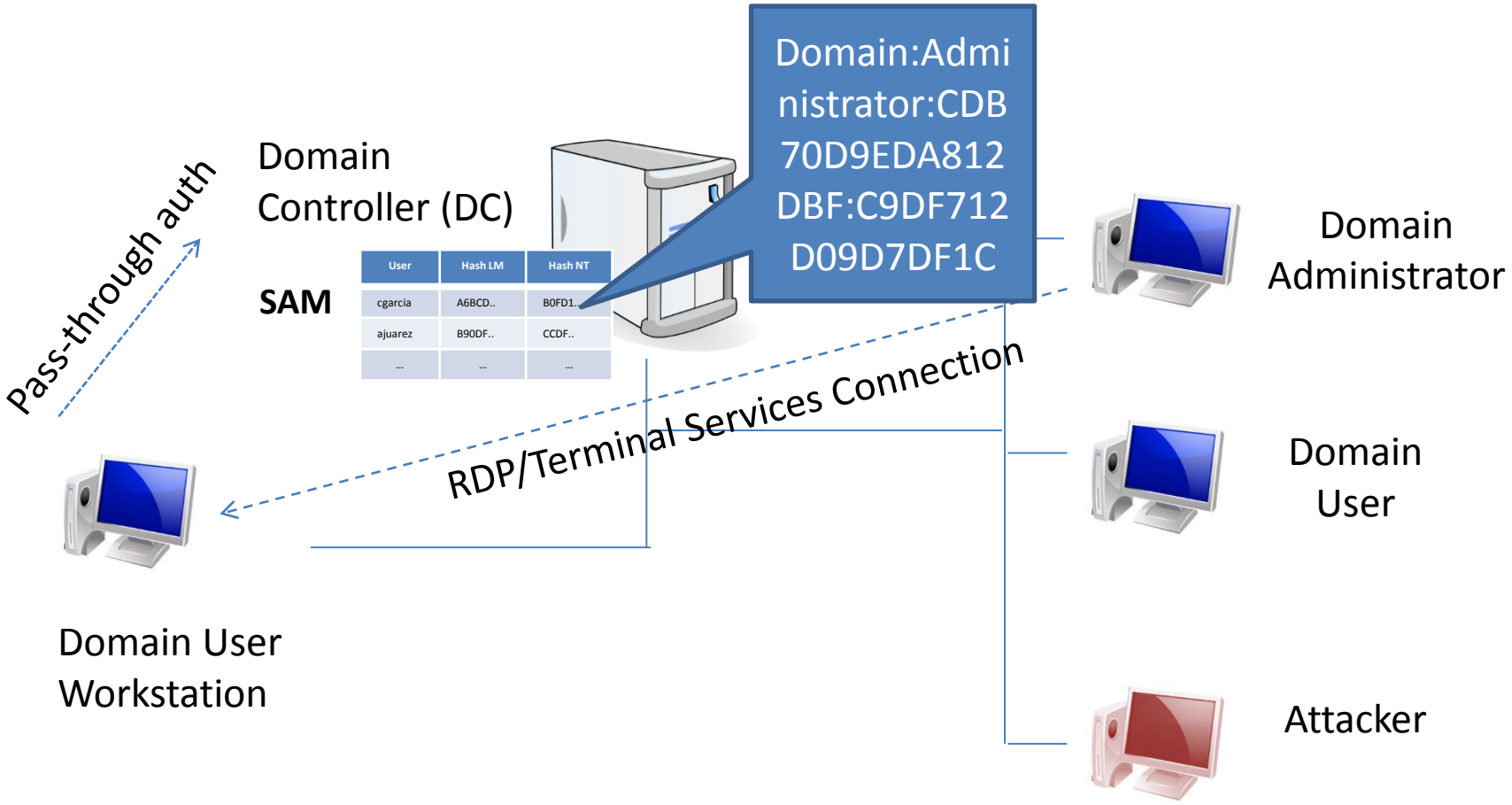
WCE: ‘Steal’ Credentials from memory

RDP ´exposes´ credentials: Example



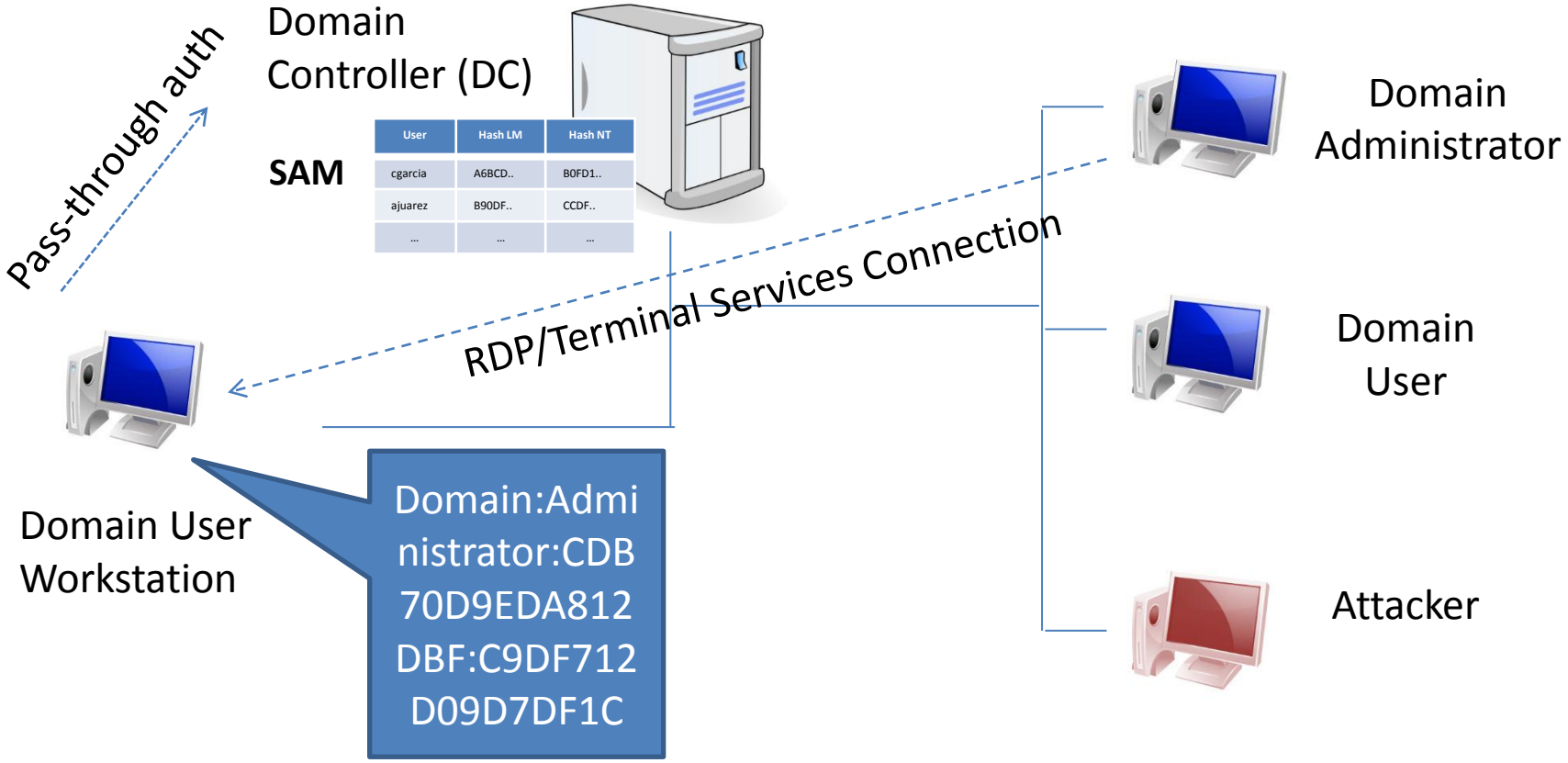
WCE: ‘Steal’ Credentials from memory

RDP ´exposes´ credentials: Example



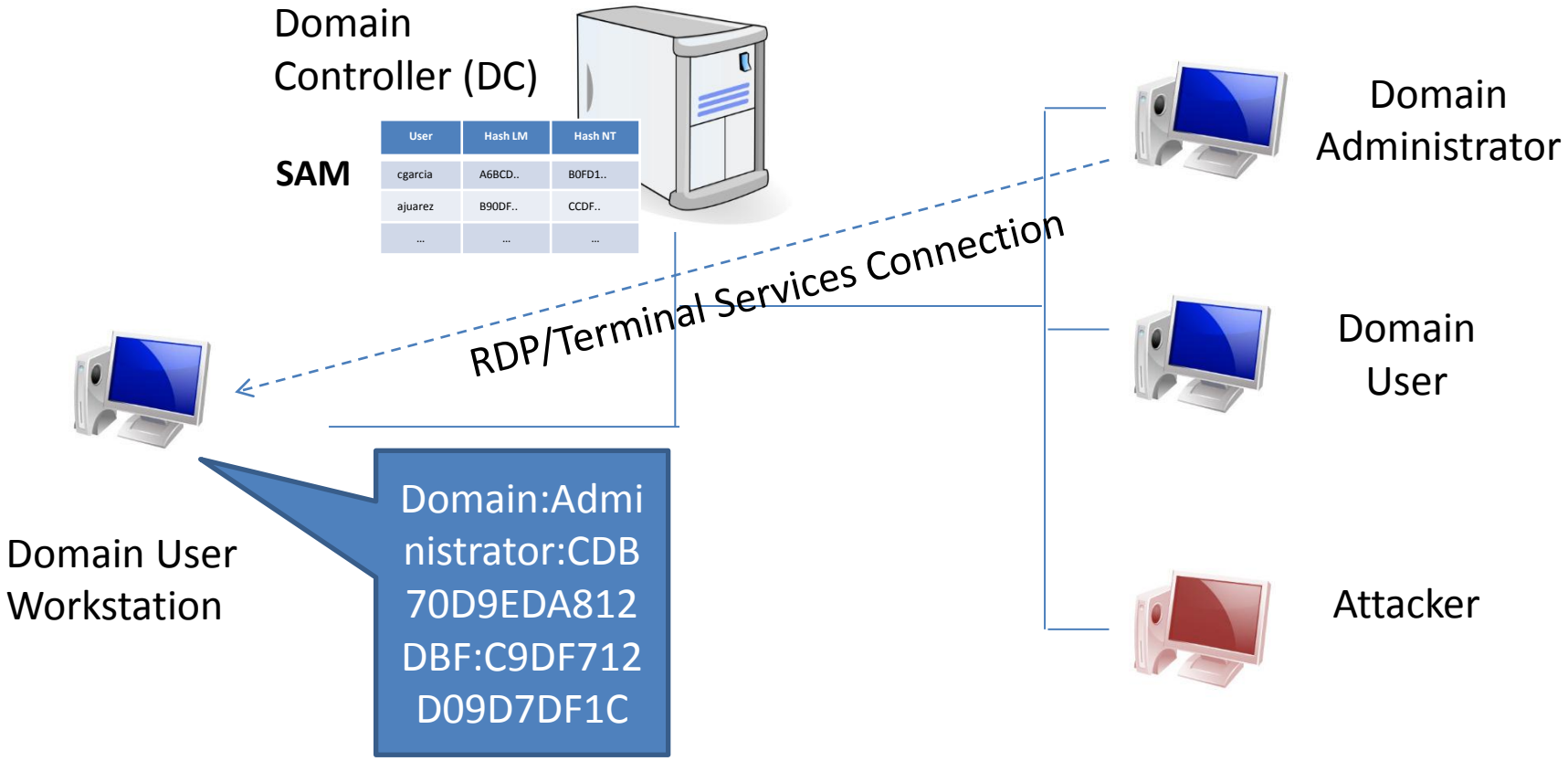
WCE: ‘Steal’ Credentials from memory

RDP ´exposes´ credentials: Example



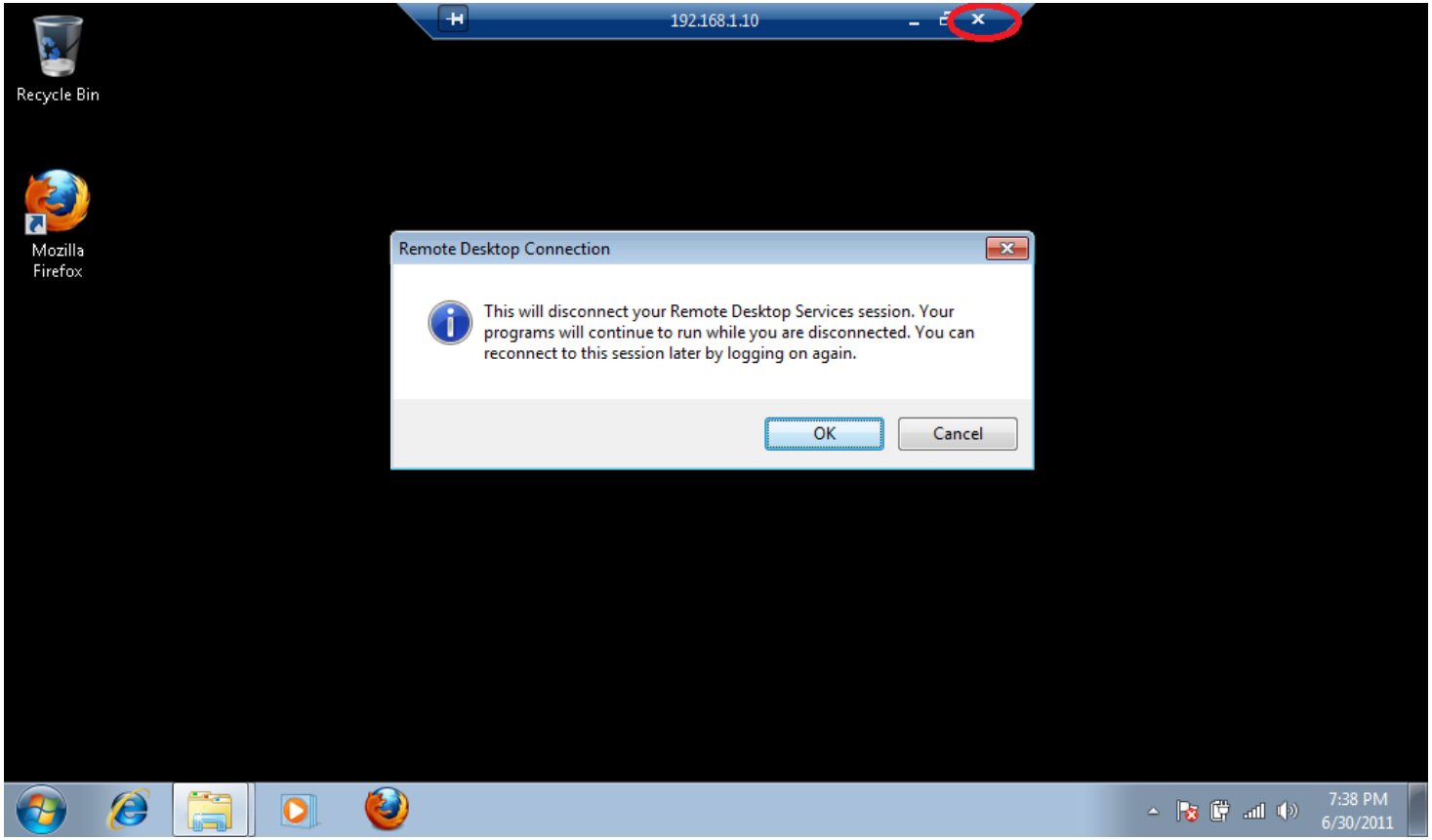
WCE: ‘Steal’ Credentials from memory

RDP ´exposes´ credentials: Example



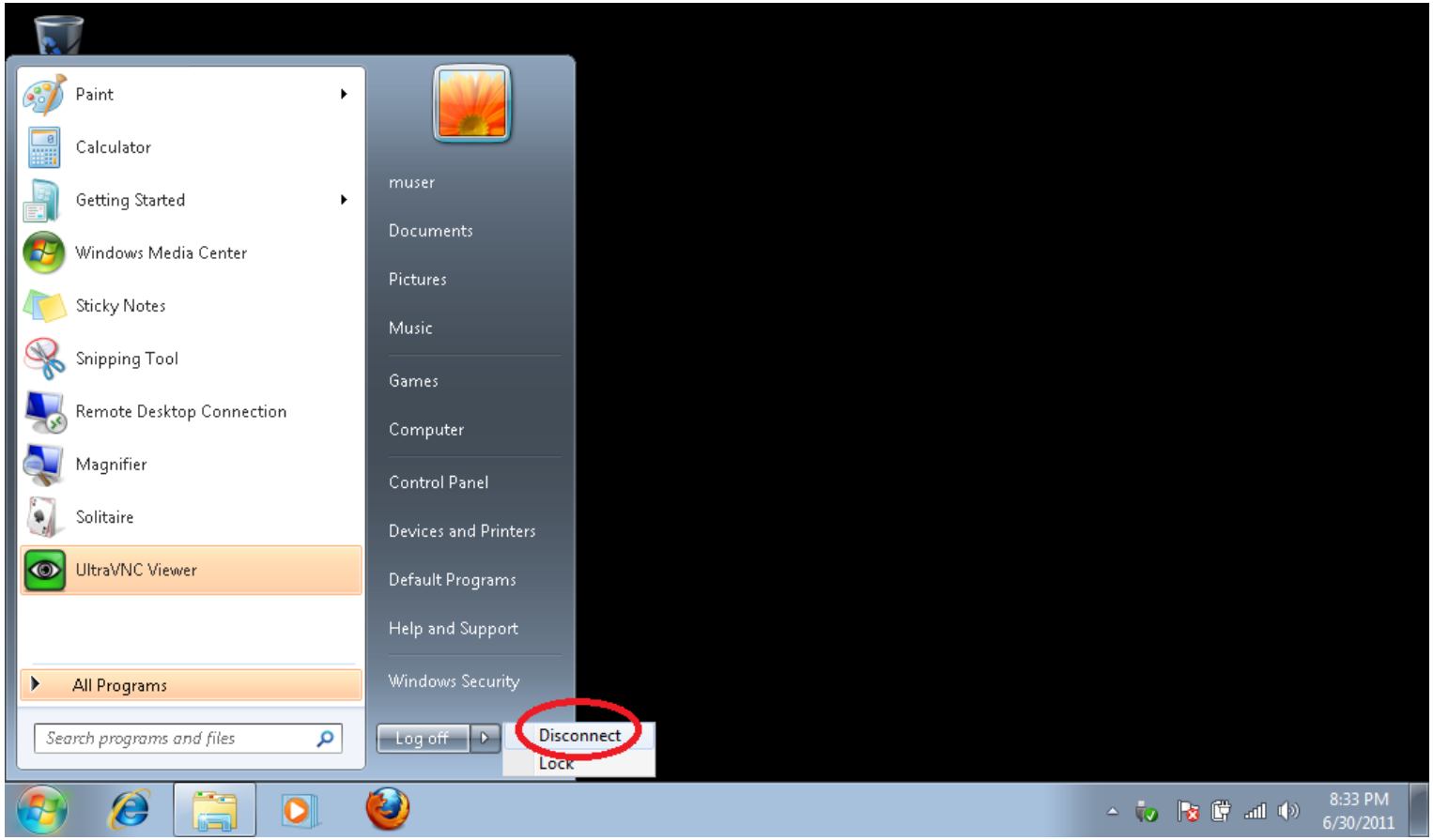
WCE: ‘Steal’ Credentials from memory

RDP: Disconnect != Log Off



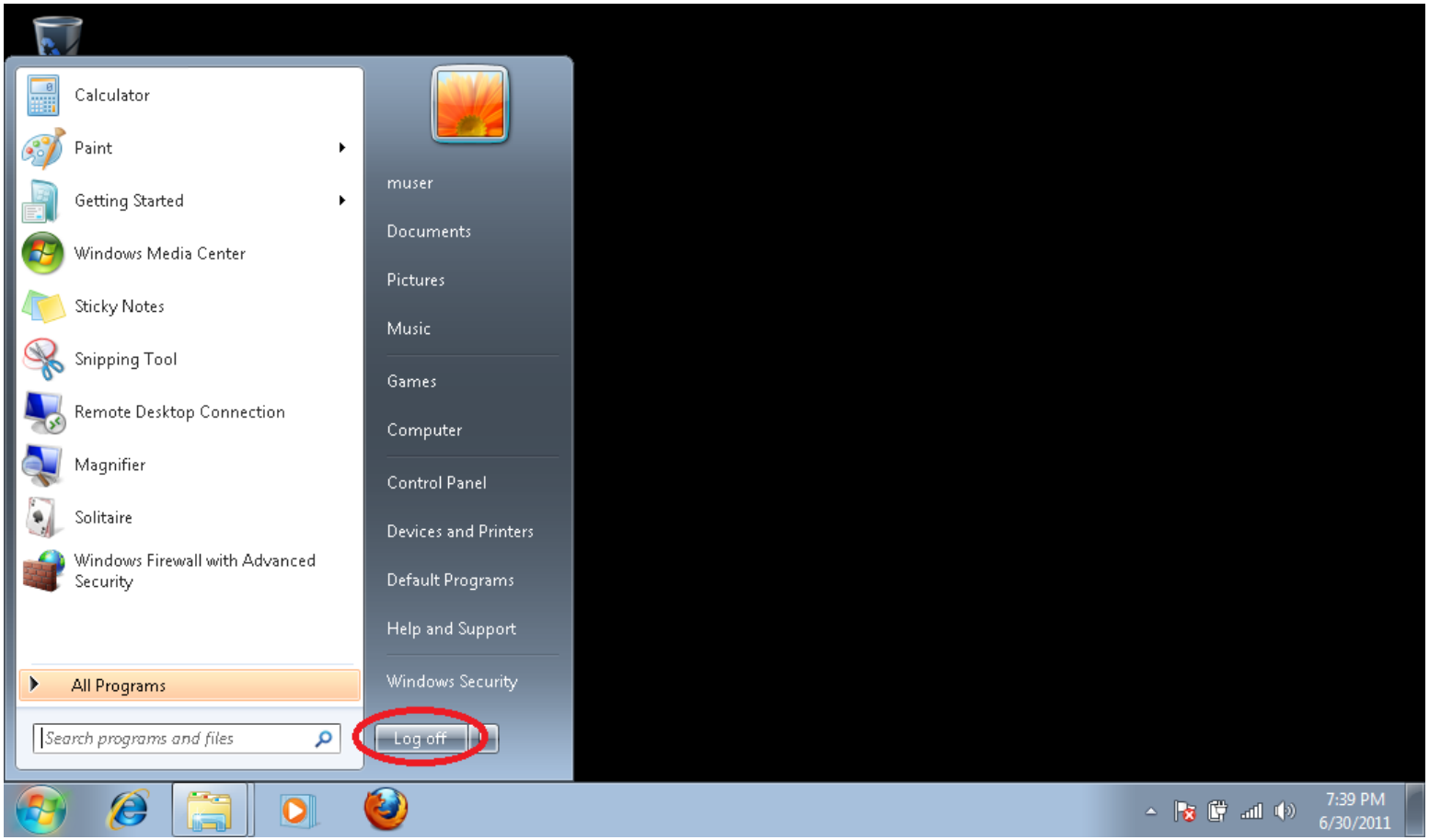
WCE: ‘Steal’ Credentials from memory

RDP: Disconnect != Log Off



WCE: ‘Steal’ Credentials from memory

RDP: Disconnect != Log Off



WCE: ‘Steal’ Credentials from memory

RDP: Disconnect != Log Off

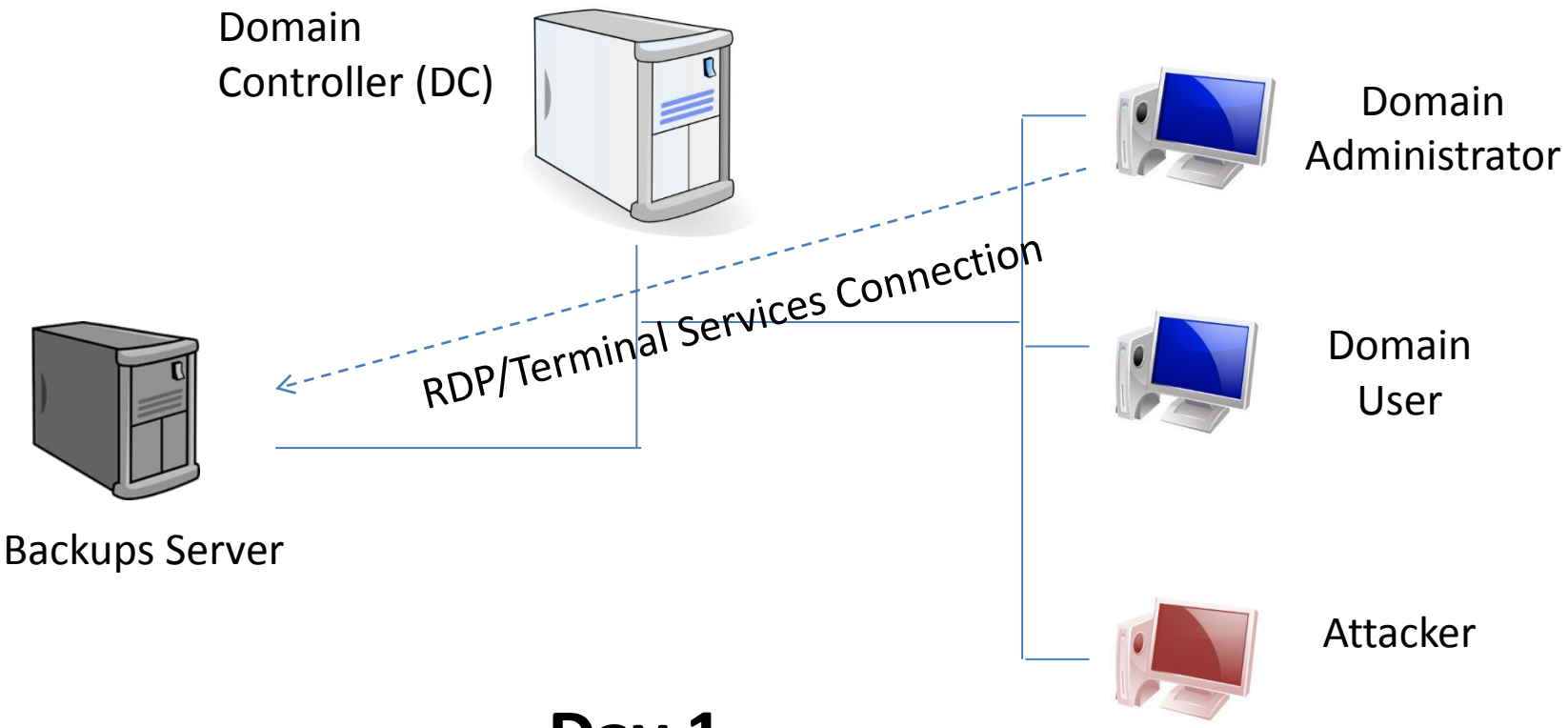
- **´Disconnect´ leaves NTLM hashes in memory**
  - The Logon session is not terminated
- ´Log Off´ terminates the logon session
  - Hashes are erased from memory
- **Always ´Log Off´**!
  - Users tend to just ´Disconnect´…
  - Including Administrators..

WCE: ‘Steal’ Credentials from memory

Zombie Logon Sessions

WCE: ‘Steal’ Credentials from memory

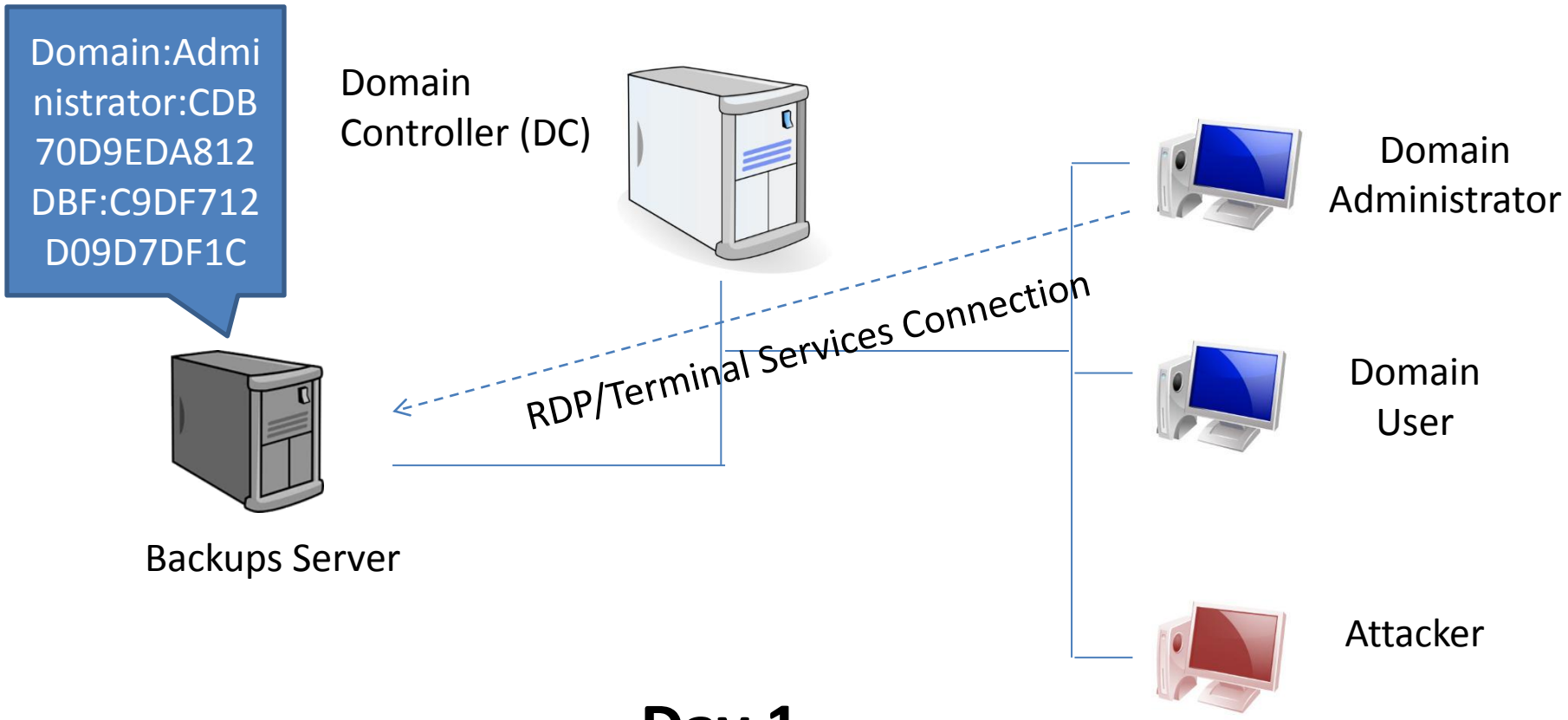
Bug: Zombie Logon Sessions!



Day 1

WCE: ‘Steal’ Credentials from memory

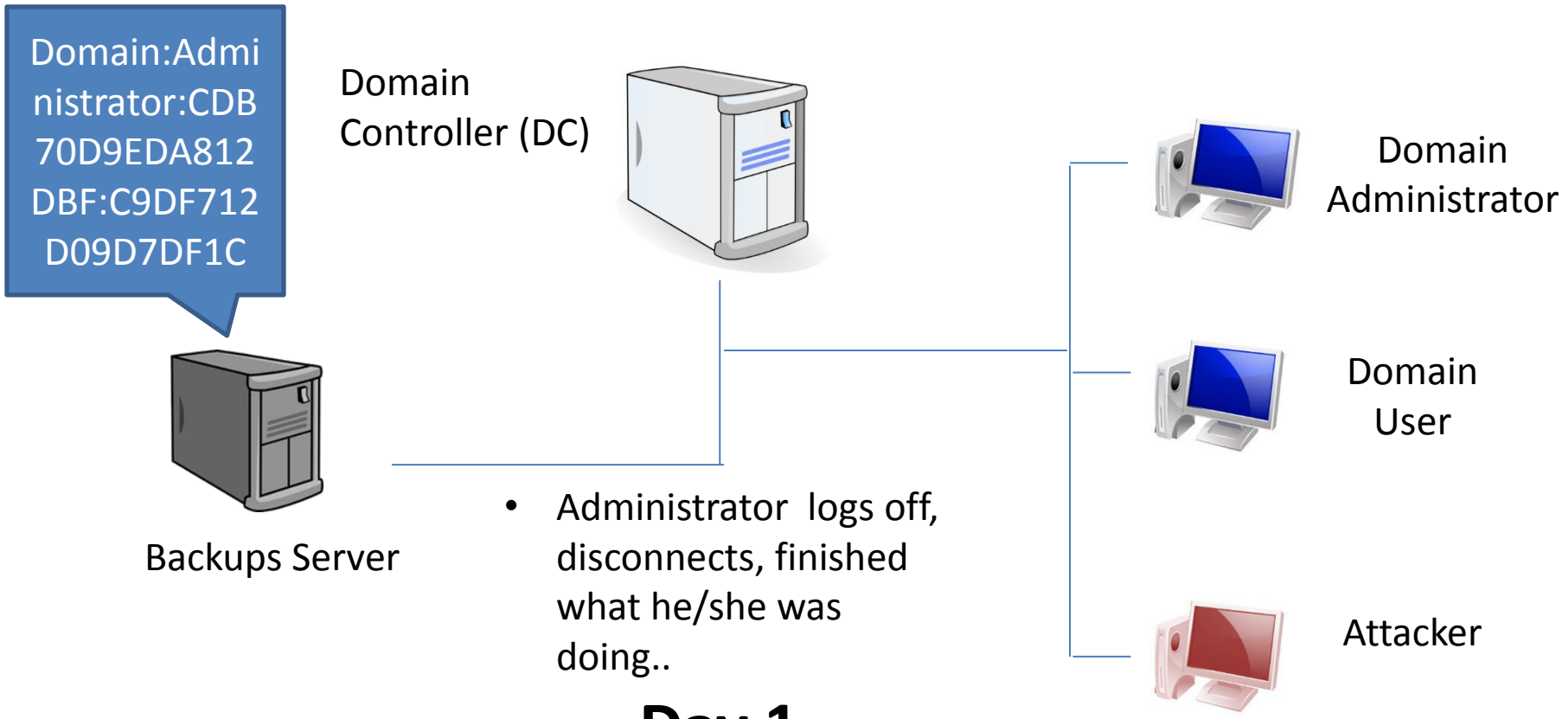
Bug: Zombie Logon Sessions!



Day 1

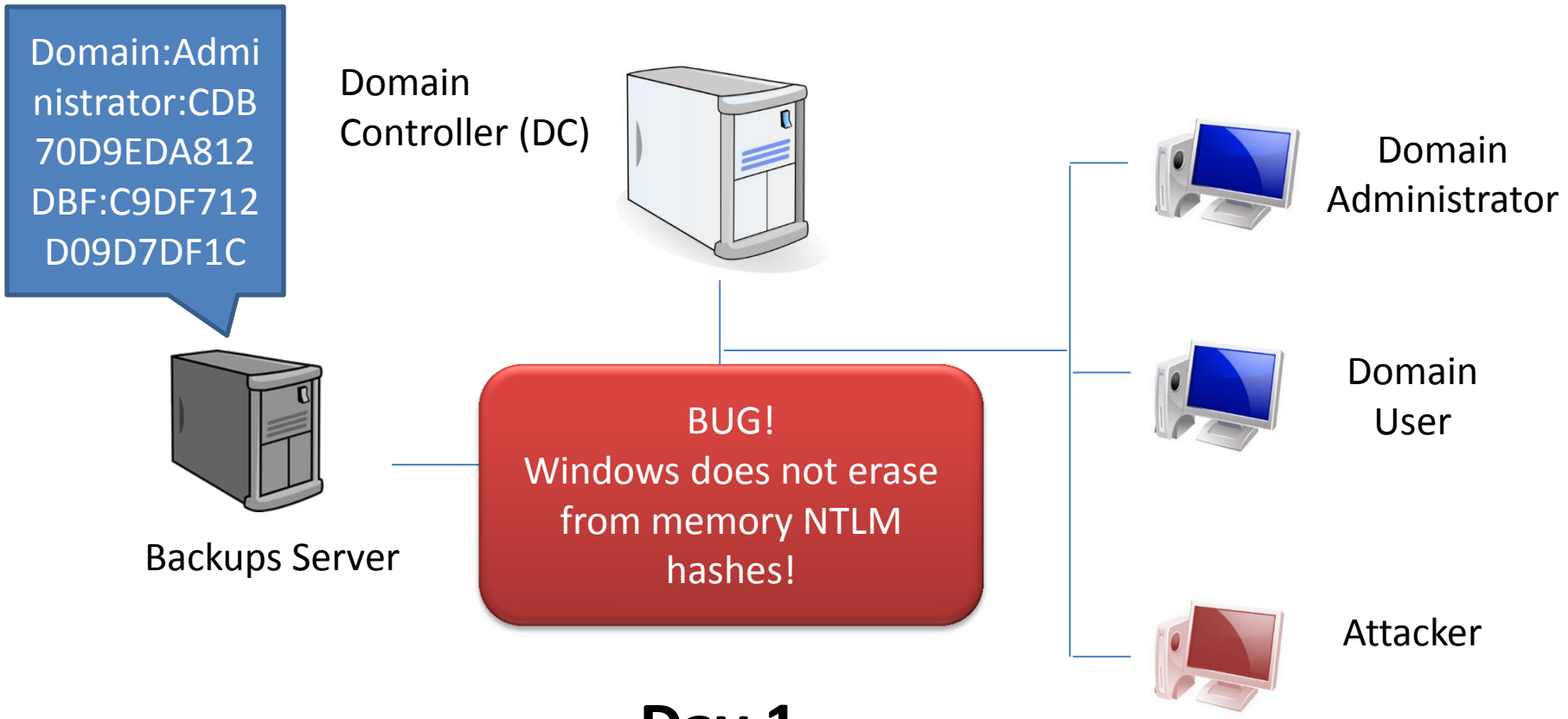
WCE: ‘Steal’ Credentials from memory

Bug: Zombie Logon Sessions!



WCE: ‘Steal’ Credentials from memory

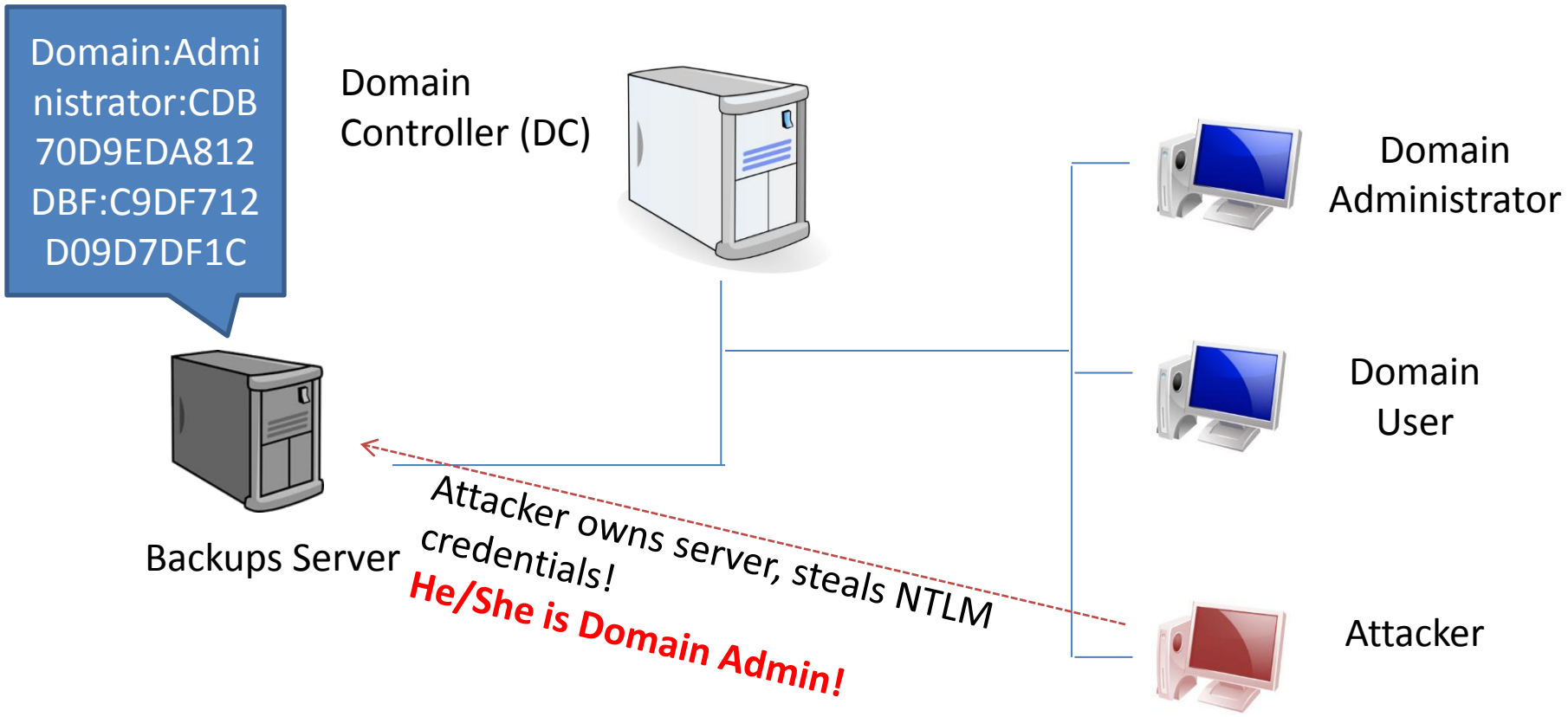
Bug: Zombie Logon Sessions!



Day 1

WCE: 'Steal' Credentials from memory

Bug: Zombie Logon Sessions!



Day 31.. ~ A month later..

WCE: Pass-the-Ticket (Kerberos)

- New attack implemented by WCE v1.2
  - First and only tool that implements this AFAIK
- Post-exploitation
- Equivalent to Pass-The-Hash for NTLM
- You can ‘steal’ Kerberos TGT/tickets & use them in other Windows and *Unix boxes

WCE: Pass-the-Ticket (Kerberos)

- ´Stolen´ tickets can be used to access remote services
  - Example: SMB shares
- The TGT (Ticket Granting Ticket) can be used to create new tickets
  - Gain access to more services / computers

Conclusions

- WCE brings new post-exploitation techniques
  - Pass-the-Hash (NTLM)
  - Steal NTLM from memory
  - Pass-the-ticket (Kerberos)
- Useful for pentests
- You need to know them to defend yourself
  - Not just to attack..

More information

- "WCE Internals" Presentation
  - RootedCon 2011; Madrid, España
  - More technical details about implementation

http://www.ampliasecurity.com/research/WCE_Internals_RootedCon2011_ampliasecurity.pdf

Questions?

Thank you!

hernan@ampliasecurity.com

@hernano
@ampliasecurity

www.ampliasecurity.com