

# THE REMOTE MALICIOUS BUTLER DID IT!

Authors:

Tal Be'ery, Sr. Security Research Manager, Microsoft

Chaim Hoch, Security Researcher, Microsoft

August 2016

©2016 Microsoft Corporation. All rights reserved. This document is provided "as-is." Information and views expressed in this document, including URL and other Internet Web site references, may change without notice. You bear the risk of using it. Some examples are for illustration only and/or are fictitious. No real association is intended or inferred.

This document does not provide you with any legal rights to any intellectual property in any Microsoft product. You may copy and use this document for your internal, reference purposes.

## Contents

Introduction .....	3
The Essentials of Logon in the Windows Domain Environment .....	4
When Domain Controller is Available: The Kerberos Protocol .....	4
Kerberos Message Flow .....	4
Users' and Computers' Logon Process.....	6
When Domain Controller is Unavailable: Cached Credentials.....	7
The Change of Password Procedure in the Domain Environment .....	8
The "Evil Maid" Attack .....	10
The attack scenario .....	10
The "Evil Maid" Attack via Cached Credentials Poisoning .....	10
Anti-"Evil Maid" Attack Patches.....	12
The "Remote Butler" Attack: The "Evil Maid" Meets the Cyber Kill-Chain .....	14
The Cyber Kill-Chain .....	14
A Deeper Look into the Cyber Kill-Chain Initial Phase .....	15
The "Remote Butler" Attack in Details .....	17
The "Remote Butler": RDP Reconnaissance Method .....	18
The "Remote Butler": Compromising the Original User's Domain Credentials.....	19
The "Remote Butler": Clean-up Step .....	20
Mitigation.....	22
Patching .....	22
Hardening.....	22
RDP hardening with Network Level Authentication (NLA) .....	23
Kerberos Armoring.....	24
Defense-in-Depth.....	26
Conclusions .....	28

## Introduction

In the past few years, attacks have hit major organizations world-wide. The perpetrators of these attacks are determined to achieve their goals and are characterized by usually advanced methods and persistence (APTs). A crucial part in the success of the attack involves maneuvering the network of the target and compromising the credentials of the domain administrators, in order to gain complete control of the target network.

At Black Hat Europe 2015, Ian Haken in his talk "Bypassing Local Windows Authentication to Defeat Full Disk Encryption" demonstrated a sophisticated attack that allows the attacker to bypass BitLocker disk encryption in an enterprise domain environment. The attacker can do so by connecting the unattended computer into a rogue Domain Controller and abusing a client side authentication vulnerability. These types of attacks are known as an "Evil Maid" attack, as the attack has to have physical access to the target in order to carry out the attack.

As a result of this "Evil Maid" attack, Microsoft released some patches to fix this vulnerability and mitigate the attack (MS15-122 and MS16-014). While being a clever attack, the physical access requirement for the attack seems to be prohibitive and would prevent it from being used on most APT campaigns. As a result, defenders might not correctly prioritize the importance of patching it.

The "Remote Malicious Butler" attack is an extension of this attack, demonstrating how attackers can utilize the original attack take control over a remote computer and thus enabling an attacker to maneuver in a target network. In this document, we dive into the technical details of the attack including the rogue Domain Controller, the client-side vulnerability and the Kerberos authentication protocol network traffic that ties them. We suggest some practical defensive recommendations, to help defenders protect their networks against such attacks.

## The Essentials of Logon in the Windows Domain Environment

The following section describes some key aspects of the logon process in the Windows Domain environment. Familiarity with these elements is crucial for the understanding of the attacks described on the following sections. Please note that some of the exact technical details may have been simplified for brevity.

### When Domain Controller is Available: The Kerberos Protocol

When the Domain Controller is available, the authentication (e.g. verification of the user's password) is performed against it, via the Kerberos protocol.

The Kerberos protocol is an authentication and authorization protocol, standardized and maintained by the IETF (mainly in RFC 4120<sup>1</sup>) and implemented by many Operating Systems (OS), including but not limited to Windows, Linux and Mac OSX.

The Kerberos protocol enables the transparent Single-Sign-On (SSO) user experience. The SSO enables users to actively authenticate (i.e. provide their password) only once even though they access various services.

### Kerberos Message Flow

The Kerberos authentication and authorization protocol works in the following manner:

---

<sup>1</sup> <https://www.ietf.org/rfc/rfc4120.txt>

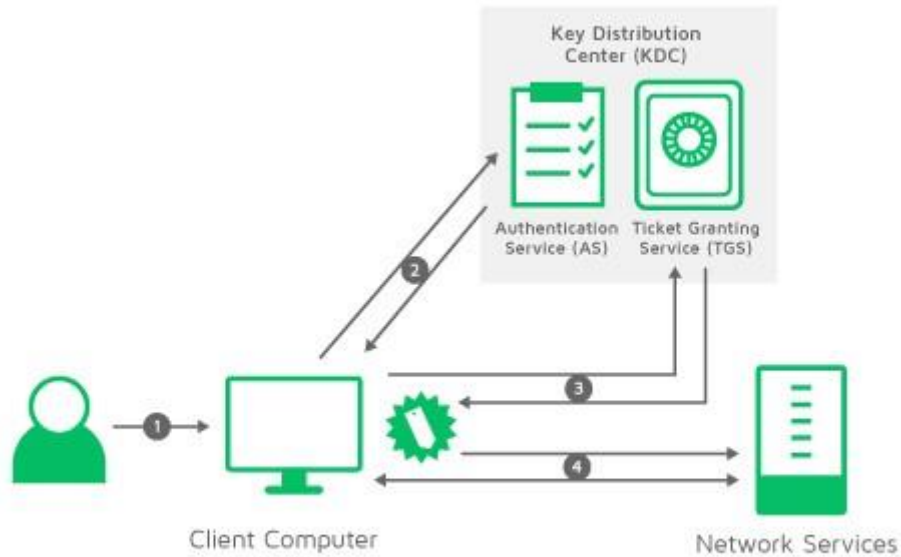


Figure 1 Kerberos Authentication Flow

1. The user provides the Domain Name, user name and password to logon to the computer.
2. The computer authenticates to the Authentication Server (AS) residing on the Kerberos Key Distribution Center (KDC). Accordingly, the KDC provides the computer with a Ticket Granting Ticket (TGT). The TGT is a token which enables the computer to request access to services without having the user to re-supply their credentials.
3. Each time the computer attempts to access a service, it first identifies itself to the Domain Controller (DC), residing on the KDC, with the TGT as provided earlier by the AS. The DC, through its Ticket Granting Server (TGS), provides the user with a ticket for the particular requested service, encrypted with the target service's long term key (derived from its password).
4. The user provides the service ticket to the service. Since the ticket is encrypted with the service's long term key and was validated by the TGS, the service grants access according to the authorization data specified in the ticket. Accordingly, the connection between the user and the service is established.

In Windows networks the KDC is implemented in the Active Directory (AD) service on the Domain Controller (DC) server. Therefore, we will use KDC, AD and DC interchangeably throughout this document.

Note that for subsequent access requests only steps 3 and 4 are repeated and the Authentication Server (AS) is not involved in these transactions. The provided TGT is used as a proof that a successful authentication had taken place.

The user credentials are only used in the preliminary authentication stage. From thereafter, the Kerberos protocol only uses the TGT ticket. On the one hand, this feature improves the efficiency of the protocol. On the other hand, the TGT token now becomes a single point of failure in the authentication and authorization process.

### Users' and Computers' Logon Process

In a domain environment, a computer is assigned with an account name (with a fixed '\$' suffix) and credentials<sup>2</sup>. The computer itself also mutually authenticates against the DC via the Kerberos protocol. **A successful authentication between computer and its DC, is often referred to as "Domain Trust Relationship", or more simply as "Domain Trust"** (The term is often used in a negative manner, for cases in which the computer is unable to authenticate against the DC and the user is prompted with a "The trust relationship between this workstation and the primary domain failed" message<sup>3</sup>).

Info	KerberosString
AS-REQ	aoratoessrv8\$,krbtgt,AORATO.RESEARCH
KRB Error: KRB5KDC_ERR_PREAUTH_REQUIRED	krbtgt,AORATO.RESEARCH
AS-REQ	aoratoessrv8\$,krbtgt,AORATO.RESEARCH
AS-REP	AORATOESSRV8\$,krbtgt,AORATO.RESEARCH

Figure 2 Computer's authentication message flow, as captured on the wire: aoratoessrv8 computer authenticates

**The user logon process is a special case of the Kerberos authentication, in which the target server is the computer the user wants to logon to.** The service ticket is encrypted by the computer's password. As a result, the Domain Trust is validated on a successful user's logon.

Info	KerberosString
AS-REQ	bugsb,krbtgt,aorato.research
KRB Error: KRB5KDC_ERR_PREAUTH_REQUIRED	krbtgt,aorato.research
AS-REQ	bugsb,krbtgt,aorato.research
AS-REP	bugsb,krbtgt,AORATO.RESEARCH
TGS-REQ	krbtgt,AORATO.RESEARCH,host,aoratoessrv8.aorato.research
TGS-REP	bugsb,host,aoratoessrv8.aorato.research

Figure 3 Logon process message flow, as captured on the wire: user bugsb logs on to aoratoessrv8 computer

<sup>2</sup> [https://technet.microsoft.com/en-us/library/cc731641\(v=ws.11\).aspx](https://technet.microsoft.com/en-us/library/cc731641(v=ws.11).aspx)

<sup>3</sup> <https://support.microsoft.com/en-us/kb/2771040>

## When Domain Controller is Unavailable: Cached Credentials

When the Domain Controller is not available, the authentication (e.g. verification of the user's password) must be performed locally. To support this very relevant scenario, Microsoft Windows caches previous users' logon information locally.

When a new user successfully logs on to a computer, a digest of the user's password is created and stored along with additional data in the registry.

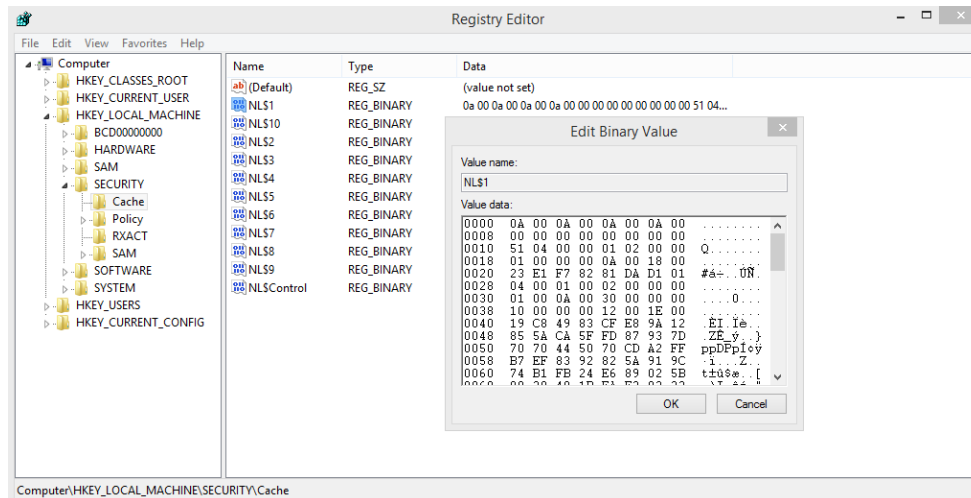


Figure 4 A screenshot of Cached Credentials, stored in the registry

The digest algorithm (for Windows Vista and onwards<sup>4</sup>) is often referred to as MS-Cache2 or MS-DCC2 (Domain Cached Credentials) and is calculated as follows<sup>5</sup>:

1. The password is encoded using UTF-16-LE.
2. The MD4 digest of step 1 is calculated. (The result of this is identical to the nthash digest of the password).
3. The unicode username is converted to lowercase, and encoded using UTF-16-LE. This should be just the plain username (e.g. User not SOMEDOMAIN\\User)
4. The username from step 3 is appended to the digest from step 2; and the MD4 digest of the result is calculated
5. PBKDF2-HMAC-SHA1 is then invoked, using the result of step 4 as the secret, the username from step 3 as the salt, 10240 rounds, and resulting in a 16 byte digest.
6. The result of step 5 is encoded into hexadecimal; this is the DCC2 hash.

<sup>4</sup> The original (older) version of the algorithm is referred to as MS-Cache or MS-DCC

<sup>5</sup> <https://pythonhosted.org/passlib/lib/passlib.hash.msdcc2.html>

The digest is stored in the registry in the HKEY\_LOCAL\_MACHINE\SECURITY\Cache hive. This hive is only accessible by the System Account and its contents are encrypted.

## The Change of Password Procedure in the Domain Environment

The change of password procedure is an important part of managing the user's credentials lifecycle.

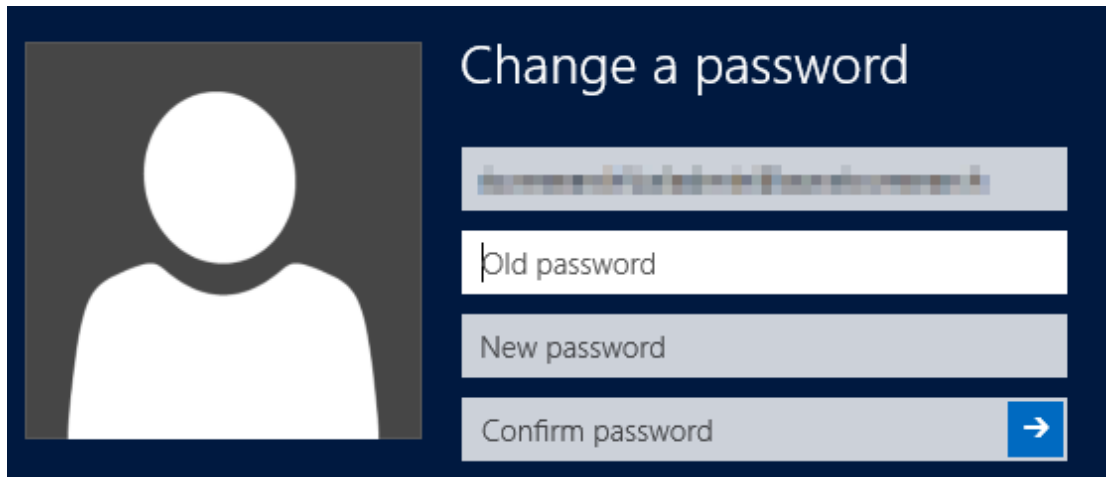


Figure 5 - Windows UI for the change of a user's password

The change of password is also implemented via the Kerberos protocol<sup>6</sup>:

1. Using its (old) domain credentials, the user authenticates against the KDC's KADMIN/CHANGE PW service, dedicated for the change of password task.
2. The user sends the new password in an encrypted message of the KPASSWD protocol. The password string is needed (and not a digest) so that the KDC will be able to enforce password complexity.

Source	Destination	Protocol	Info
192.168.0.17	192.168.0.2	KRB5	AS-REQ
192.168.0.2	192.168.0.17	KRB5	KRB Error: KRB5KDC_ERR_PREAUTH_REQUIRED
192.168.0.17	192.168.0.2	KRB5	AS-REQ
192.168.0.2	192.168.0.17	KRB5	KRB Error: KRB5KDC_ERR_KEY_EXP NT Status: STATUS_PASSWORD_MUST_CHANGE
192.168.0.17	192.168.0.2	KRB5	AS-REQ
192.168.0.2	192.168.0.17	KRB5	KRB Error: KRB5KDC_ERR_PREAUTH_REQUIRED
192.168.0.17	192.168.0.2	KRB5	AS-REQ
192.168.0.2	192.168.0.17	KRB5	AS-REP
192.168.0.17	192.168.0.2	KPASSWD	Reply
192.168.0.2	192.168.0.17	KPASSWD	Reply

Figure 6 Network Traffic flow of a successful change of password procedure

<sup>6</sup> <https://www.ietf.org/rfc/rfc3244.txt>



A successful change of password process, triggers the update of the user's Cached Credentials store on the machine of which the update was performed.

**Note, that this procedure only involves the users' credentials and does not validate Domain Trust (i.e. the machine's domain authentication or keys)**

## The “Evil Maid” Attack

### The attack scenario

An “Evil Maid” attack is one where the attacker has physical access to the victim’s unattended computer. The term was coined by Joanna Rutkowska on 2009<sup>7</sup>: “You leave your laptop (can be even fully powered down) in a hotel room and go down for a breakfast... Meanwhile an Evil Maid enters your room.”

The main challenge of this scenario is dealing with Hard Drive’s (HD) hardware based encryption. If the HD is not encrypted, the problem becomes trivial, as the attacker can just mount it to another computer.

Rutkowska’s attack was based on booting the victim computer from an evil USB. Other researchers (Halderman et al<sup>8</sup>) suggested the “cold-boot” attack. In this attack, the encryption keys are extracted from the RAM of a powered-down computer and used to decrypt the hard drive.

### The “Evil Maid” Attack via Cached Credentials Poisoning

At Black Hat Europe 2015, Ian Haken<sup>9</sup> presented<sup>10</sup> a novel “Evil Maid” attack that leverages the change password mechanism in order to bypass Windows authentication and gain access to a machine. The attack worked as follows:

1. The attackers set up a new rogue DC with the same domain name as of the victim’s computer. The name of the domain can be easily extracted from the lock screen UI.
2. On the rogue DC, the attackers create a user account with the same username as the user logged-on to the victim machine. As in before, the username can be easily extracted from the lock screen UI. The user’s password on the Rogue DC is controlled by the attackers and they set it to be expired and as a result, a password change will be requested by the DC on the user’s next logon.
3. The attackers physically connect the victim machine to the Rogue DC.
4. The attackers log on with the password they previously set on the rogue DC and is prompted to change it.

---

<sup>7</sup> <http://theinvisiblethings.blogspot.com/2009/01/why-do-i-miss-microsoft-bitlocker.html>

<sup>8</sup> <http://citpsite.s3-website-us-east-1.amazonaws.com/oldsite-hdocs/pub/coldboot.pdf>

<sup>9</sup> <https://twitter.com/ianhaken>

<sup>10</sup> <https://www.blackhat.com/eu-15/briefings.html#ian-haken>

5. The attackers change the password to another arbitrary password. This password is updated in the victim's computer's Cached Credentials
6. The attackers disconnect the victim machine from the rogue DC
7. Now that there is no DC, a Cached Credentials logon is performed, and the attackers logs on with their new password (created on step 5)

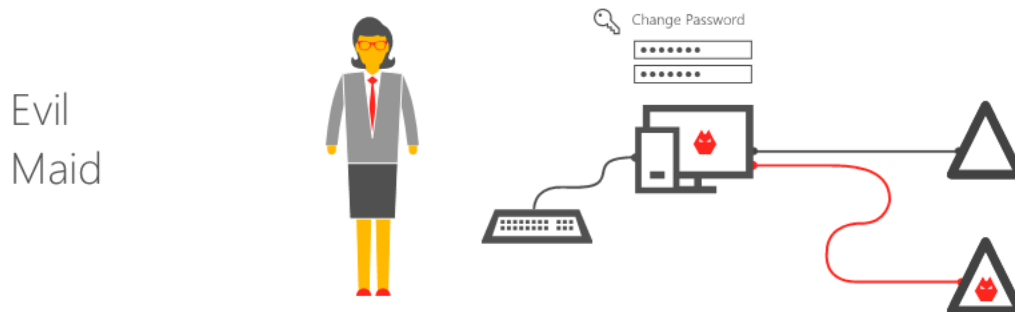
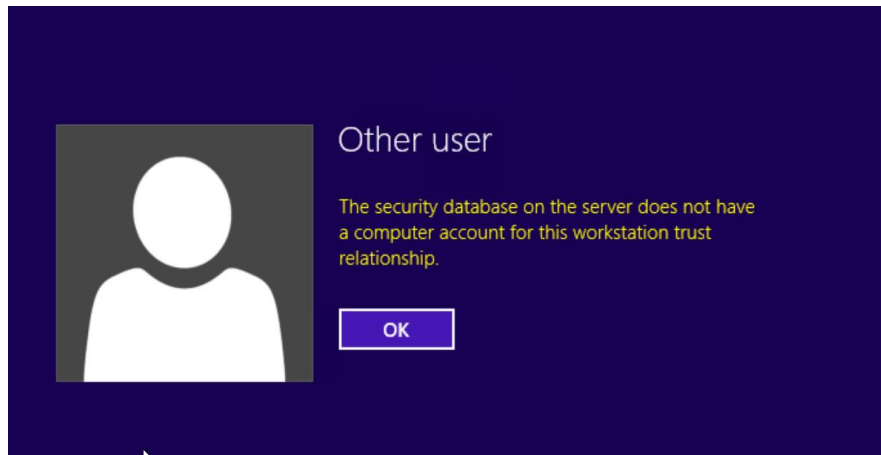


Figure 7- The "Evil Maid" attack, illustrated. Triangles represents DCs

This attack is successful, as the change password procedure does not validate the domain trust, which enables attackers to change the user's password even though they don't have the machine credentials. As a result of the successful password change, the Cached Credentials are updated ("poisoned") with the attackers' supplied password.

If the target machine is left connected to the rogue DC and the attackers try to logon, the attempt would fail, as the attackers' Rogue DC doesn't have the computer's key and thus the machine identifies that the Domain Trust Relationship is not verified. The user would be prompted with the following message:



*Figure 8- Broken trust prompt.*

However, when the machine is disconnected and the attacker logs on, the machine checks the provided credentials against its cached credential store. That cache is poisoned with an attacker-chosen password and thus the logon is successful. After logon, attackers gain access to sensitive information on the machine and is able to install a backdoor to enable them to return to the computer in future.

This elegant attack which does not involve executing a single line of attackers' code on the victim's computer, had been fully automated by the researcher, and released as the open source "bluebox" project<sup>11</sup>.

### Anti-"Evil Maid" Attack Patches

The root cause weakness which enables the attack is that the change password procedure does not validate the domain trust, which enables attackers to change the user's password even though they don't have the machine credentials. This vulnerability was assigned with the CVE-2015-6095<sup>12</sup>

The relevant Windows patch (MS15-122<sup>13</sup>) addressed that root cause by adding a check for the Domain Trust to the change password procedure, by thus preventing the Cached Credentials poisoning.

---

<sup>11</sup> <https://github.com/JackOfMostTrades/bluebox>

<sup>12</sup> <https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2015-6095>

<sup>13</sup> <https://technet.microsoft.com/en-us/library/security/ms15-122.aspx>

Later on researchers (Nabeel Ahmed and Tom Gilis<sup>14</sup>) had found out that this fix is incomplete as the Domain Trust is not properly validated (CVE-2016-0049<sup>15</sup>) and a subsequent patch was released to address it (ms16-014<sup>16</sup>).

The same researchers had identified additional Windows vulnerabilities (currently patched), related to Domain Trust validation that could lead to Privileges Escalation.

---

<sup>14</sup> <http://www.slideshare.net/NabeelAhmed7/from-zero-to-system-on-full-disk-encrypted-windows-system>

<sup>15</sup> <https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-0049>

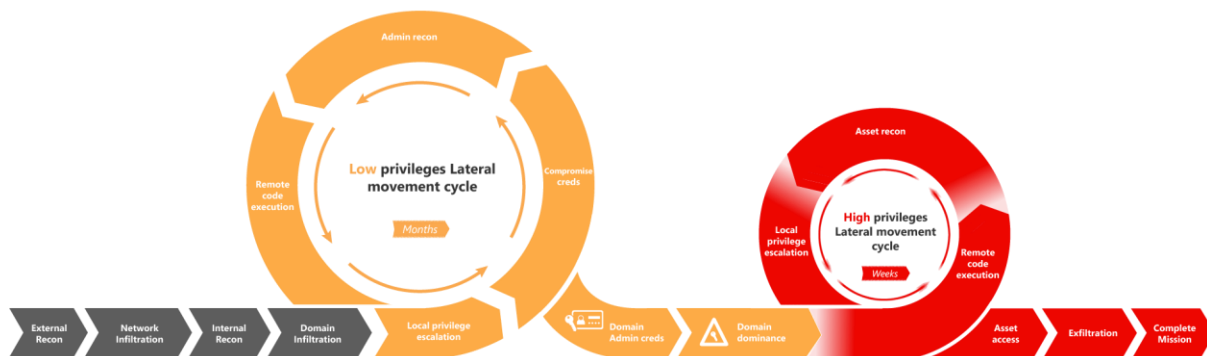
<sup>16</sup> <https://technet.microsoft.com/en-us/library/security/ms16-014.aspx>

## The “Remote Butler” Attack: The “Evil Maid” Meets the Cyber Kill-Chain

Advanced attackers (APTs) attack their victims’ networks, by abusing the authentication mechanisms, usually with the use of compromised credentials. Since the “Evil maid” attack implements an elegant (no credentials needs to be compromised) and efficient (can be fully automated) method of bypassing authentication controls, it theoretically presents an invaluable new tool for such attackers. However, the “Evil Maid” attack requires the attackers to have a physical access to the victim computers, which the attackers lack in the vast majority of their attacks. In this section we present the “Remote Butler” attack which enables attackers to perform the “Evil Maid” attack in a network environment, **with no physical access**.

### The Cyber Kill-Chain

The Cyber Kill-chain is an accepted model to describe Advanced attackers (APT) modus operandi (MOs). The model was first presented by Lockheed Martin<sup>17</sup>, and since then many parties had suggested a more detailed model. In this section we would use the one suggested by the Microsoft Advanced Threat Analytics (ATA) group.



The attack is divided into three main phases, denoted by different colors:

1. Initial phase: this phase starts with the initial attackers’ information gathering on their target, continues with their initial penetration into the network and ends when the attackers compromise a single set of Domain credentials
2. Intermediate phase: By abusing their single set of domain credentials, attackers connect to more machines, compromise credentials found on them and repeat until they find Domain

<sup>17</sup> <http://www.lockheedmartin.com/content/dam/lockheed/data/corporate/documents/LM-White-Paper-Intel-Driven-Defense.pdf>

Admin Credentials. With these credentials, attackers can compromise the DC and obtain all domain credentials.

3. Final phase: Now the attackers have all Domain credentials in their possession, they still need to find the data relevant to their attack and exfiltrate it to their servers.

As we can see, domain credentials are the fuel that propels the Lateral Movement engine. They are critical for the first phase (in fact they are its goals) and very helpful for the second phase. Thus, the “Evil Maid” attack would be very relevant for attackers, should it was applicable to the network access scenario and not only to the physical access one.

#### A Deeper Look into the Cyber Kill-Chain Initial Phase

As stated above, obtaining a single set of domain credentials is the goal of this attackers’ phase.



*Figure 9 The Cyber Kill-Chain Initial Phase in Details*

In many attacks, the first foothold obtained in the “network infiltration” step, is a non-domain joined machine, which can be a result of the hacking of some internet facing network asset, such as a web server (“web shell” hacking<sup>18</sup>), a Router, a Security device or any other IoT device. In that situation, the attacker faces the non-trivial challenge of moving into a domain-joined machine, in order to proceed to the next attack phase. If the “Evil Maid” attack would have been applicable to the network access scenario, it would have made a perfect attack for this phase.

A real-world example for this phase can be found in the attackers’ account<sup>19</sup> of their attack on the “HackingTeam” company. The attackers compromised a non-domain joined network device (“Network Infiltration” step), discovered a network storage device that did not require authentication (“Internal

<sup>18</sup> <https://www.us-cert.gov/ncas/alerts/TA15-314A>

<sup>19</sup> <https://ghostbin.com/paste/6kho7>

Recon" step) and finally extracted Domain credentials from a VM backup stored on the vulnerable network storage device ("Domain Infiltration" step)



## The “Remote Butler” Attack in Details

The “Remote Butler” attack is, in essence, an extension of the “Evil Maid” attack, originally designed for the physical access scenario, for the network access scenario, which is the relevant scenario in APT attacks.

The attack steps are as follows:

1. Attackers first compromise a machine in the network. The machine is not necessarily domain-joined, and may not include any domain credentials on it.
2. Attackers install the needed rogue DC functionality on the breached machine (can use the aforementioned “Bluebox” project)
3. Attackers use a reconnaissance tool (such as nmap<sup>20</sup>) in order to find machine with an open RDP (Remote Desktop Protocol) port (3389).
4. Attackers passively sniffs adjacent machines’ traffic to monitor user activity and logons, in order to identify vulnerable machines:
  - a. Their traffic is seen and hence hijack-able
  - b. Their logged-on user is away from the machine (late at night, for example).
5. Attackers manipulate the routing of such vulnerable machines’ network traffic to the DC, so that such traffic so that it would go through to their machine (e.g. ARP poisoning<sup>21</sup>).
6. Attackers now perform the “Evil Maid” attack using RDP access:
  - a. Configure the rogue DC according to the details in the UI
  - b. Create a password change procedure with Rogue DC to poison the Cached Credentials
7. Attackers stop answering from the Rogue DC, thus triggering a Cached Credentials logon, with the attacker’s controlled changed password
8. Attackers are now logged-on to the victim’s computer and extract the original domain user’s Domain credentials (e.g. NTLM hash) from memory by using some memory dumping tool (e.g. Mimikatz<sup>22</sup>), thus making this computer accessible even when it returns to its original DC, without installing any additional backdoor.
9. Attackers clean-up the traces of their attack by restoring the original Cached Credentials values using the extracted NTLM hash and return the computer’s traffic to the original DC

---

<sup>20</sup> <https://nmap.org/>

<sup>21</sup> [https://en.wikipedia.org/wiki/ARP\\_spoofing](https://en.wikipedia.org/wiki/ARP_spoofing)

<sup>22</sup> <https://github.com/gentilkiwi/mimikatz>



Figure 10- "Remote Butler" attack, illustrated. Triangles are DCs

"Evil Maid" has now been fully translated to "Remote Butler"; instead of a physical rogue DC, attackers have a breached machine, the physical cables are transformed into a routing manipulation attack to hijack the Kerberos traffic and physical access is transformed into RDP access.

In the subsections below, we will take a deeper look into some of the new, non-trivial attack steps.

#### The "Remote Butler": RDP Reconnaissance Method

We had seen that attackers' use RDP reconnaissance data as part of the "Remote Butler" attack, to configure their rogue DC with the relevant domain name and user. (Alternatively, attackers can obtain this information from the analysis of the machine's traffic )

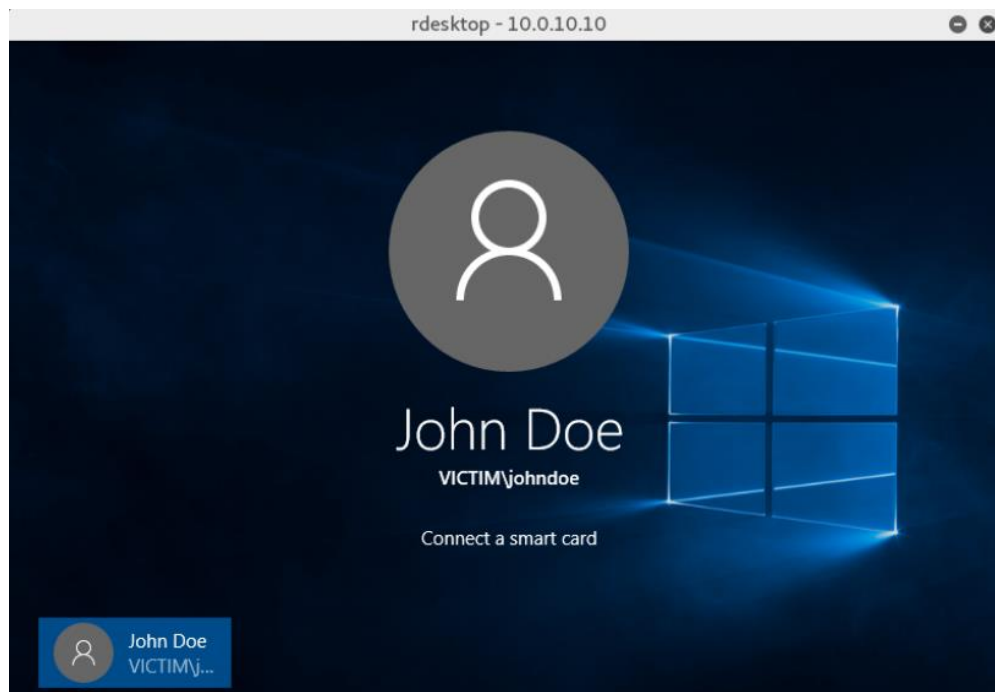


Figure 11 RDP lock-screen: Domain's and logged-on user's name are visible

However, attackers' might find such information to be relevant for other attack phases, too. In the aforementioned intermediate cyber kill-chain phase, we mentioned that attackers are looking for Domain Admins credentials, to enable them to get that phase goal: DC access.

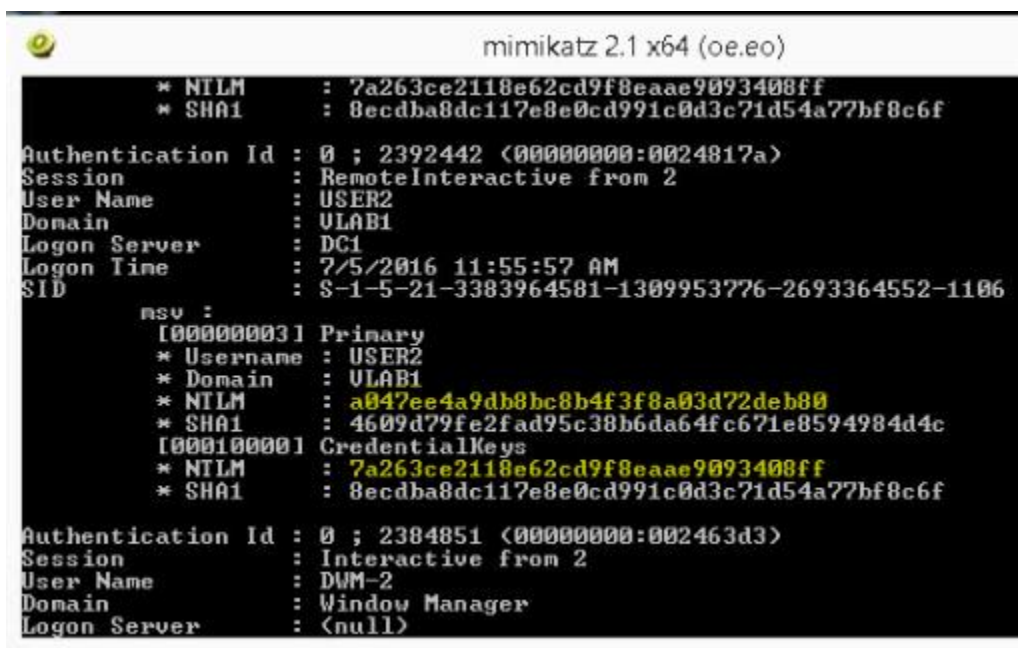
Attackers can achieve this goal by using RDP reconnaissance, which to the best of our knowledge, was not discussed before in that context. By connecting to all network accessible machines via RDP, attackers can find machines that domain admins are currently logged-on to. As a result, the attacker can target these machines, either via the "Remote Butler" attack or any other method, to obtain the Domain admin credentials.

### The "Remote Butler": Compromising the Original User's Domain Credentials

As explained above, compromising a Domain credentials set is the raison d'être of this attack phase.

Since the attackers change the original user password, we might expect that the victim user's original credentials are lost. (Of course, Domain credentials can still be obtained "by chance", if other Domain users has still live sessions on the computer, or the Domain credentials are saved in some text files, e-mails etc.)

However, it seems that in the case of a password change (any password change – not related to the specific of the "Evil Maid"/"Remote Butler" attack), the old password's keys still remain in memory.



```
mimikatz 2.1 x64 (oe.oe)
* NTLM      : 7a263ce2118e62cd9f8eaae9093408ff
* SHA1      : 8ecd8a8dc117e8e0cd991c0d3c71d54a77bf8c6f

Authentication Id : 0 ; 2392442 (00000000:0024817a)
Session           : RemoteInteractive from 2
User Name         : USER2
Domain            : ULAB1
Logon Server      : DC1
Logon Time        : 7/5/2016 11:55:57 AM
SID               : S-1-5-21-3383964581-1309953776-2693364552-1106

msv :
[00000003] Primary
* Username      : USER2
* Domain        : ULAB1
* NTLM          : a047ee4a9db8bc8b4f3f8a03d72deb80
* SHA1          : 4609d79fe2fad95c38b6da64fc671e8594984d4c
[00010000] CredentialKeys
* NTLM          : 7a263ce2118e62cd9f8eaae9093408ff
* SHA1          : 8ecd8a8dc117e8e0cd991c0d3c71d54a77bf8c6f

Authentication Id : 0 ; 2384851 (00000000:002463d3)
Session           : Interactive from 2
User Name         : DVM-2
Domain            : Window Manager
Logon Server      : <null>
```

Figure 12 A Mimikatz memory dump of a session in which the password was changed, both NTLM hashes (old and new) are highlighted

Therefore, attackers can be sure they will be able to extract Domain Credentials as a result of their “Remote Butler” attack.

#### The “Remote Butler”: Clean-up Step

In most cases, an explicit clean-up of the “Remote Butler” attack is not needed. Once the attackers reroute the victim machine to talk back with its original DC, the attack will naturally “evaporate” from the attacked computer: When the original users’ logon (when they arrive at work in the morning, for example), they use their original password to authenticate against the original DC, which in turn updates the Cached Credentials to erase the attackers’ password and restores the original password.

However, if the victims had disconnected their machine from the network (e.g. they arrive at work in the morning, and take their laptop into a meeting room which offers no LAN connectivity), they will be unable to logon to their computer, as the Cached Credentials remains poisoned with the attackers’ password. This scenario might draw unwanted suspicion to the attackers’ campaign, and therefore attackers would like to prevent it.

As mentioned above, MsCacheV2 is derived from the username and from an MD4 hash of the user’s password (which is his NTLM hash). In the change password process, the old NTLM hash is kept in memory as well, which means that if local administrator privileges are available – they can be extracted.


A Cached Credentials entry holds the username, domain name, last update time and other data. The last part of the entry is encrypted and it holds the MsCacheV2. The data can be decrypted using the NL\$KM registry key (SECURITY\Policy\Secrets\NL\$KM), which in turn decrypted using the LSA registry key (SECURITY\Policy\PolEKList)<sup>23</sup>. The LSA registry key is encrypted using the ‘boot’ key, which is comprised of four different registry keys (SYSTEM\CurrentControlSet\Control\Lsa\{JD,Skew1,GBG,Data})<sup>24</sup> and requires SYSTEM privileges to access.

Once attackers obtained the old NTLM hash, they can revert the Cached Credentials to the old ones (of the old password). Some open-source tools (e.g. Mimikatz) already implement that functionality.

---

<sup>23</sup> <https://moyix.blogspot.com/2008/02/decrypting-lsa-secrets.html>

<sup>24</sup> <https://moyix.blogspot.com/2008/02/syskey-and-sam.html>



```
mimikatz 2.1 x64 (oe.eo)
mimikatz # lsadump::cache /user:USER2 /ntlm:7a263ce2118e62cd9f8eaae9093408ff
> User cache replace mode !
* user      : USER2
* ntlm      : 7a263ce2118e62cd9f8eaae9093408ff

Domain : CLIENT2
SysKey : ef278a6303fb627e5536ebdac5573e7c

Local name : CLIENT2 < S-1-5-21-2855241813-3116034789-286929080 >
Domain name : ULAB1 < S-1-5-21-3383964581-1309953776-2693364552 >
Domain FQDN : ULAB1.com

Policy subsystem is : 1.12
LSA Key(s) : 1. default <aa99157a-0d0d-5260-ba69-b1f2398a63db>
             [00] <aa99157a-0d0d-5260-ba69-b1f2398a63db> c6d93cfe3ef141934d2f9bec0e20e3e3ad
             185b4fbaf00a43ab94b9a603c4c45a

* Iteration is set to default <10240>

[INL$1 - 7/11/2016 1:50:45 PM]
RID      : 00000452 <1106>
User     : ULAB1\USER2
MsCacheV2 : e0d618683ef8f28fed366d94edc01799
> User cache replace mode <2>!
MsCacheV2 : ad09564248b70d11be9a652baa17d464
Checksum : 36e23de6bc1a1f40cdaed6f4a0d07bb1
> OK!
```

Figure 13 – Using Mimikatz to replace the MsCacheV2 entry with the old NTLM hash

Once the Cached Credentials are reverted, the attackers' generated entry is erased and the old password can be used again to log on. As a result, the victim user is non-the-wiser about the attack taking place.

## Mitigation

We recommend on three major ways to mitigate this attack and make it infeasible: Patching, Hardening and Defense-in-Depth policy.

### Patching

Advanced attackers mainly use compromised credentials to move laterally within the network, but when they learn about some known and proved vulnerability they do not hesitate to use them, as shown by the JPCERT data below.

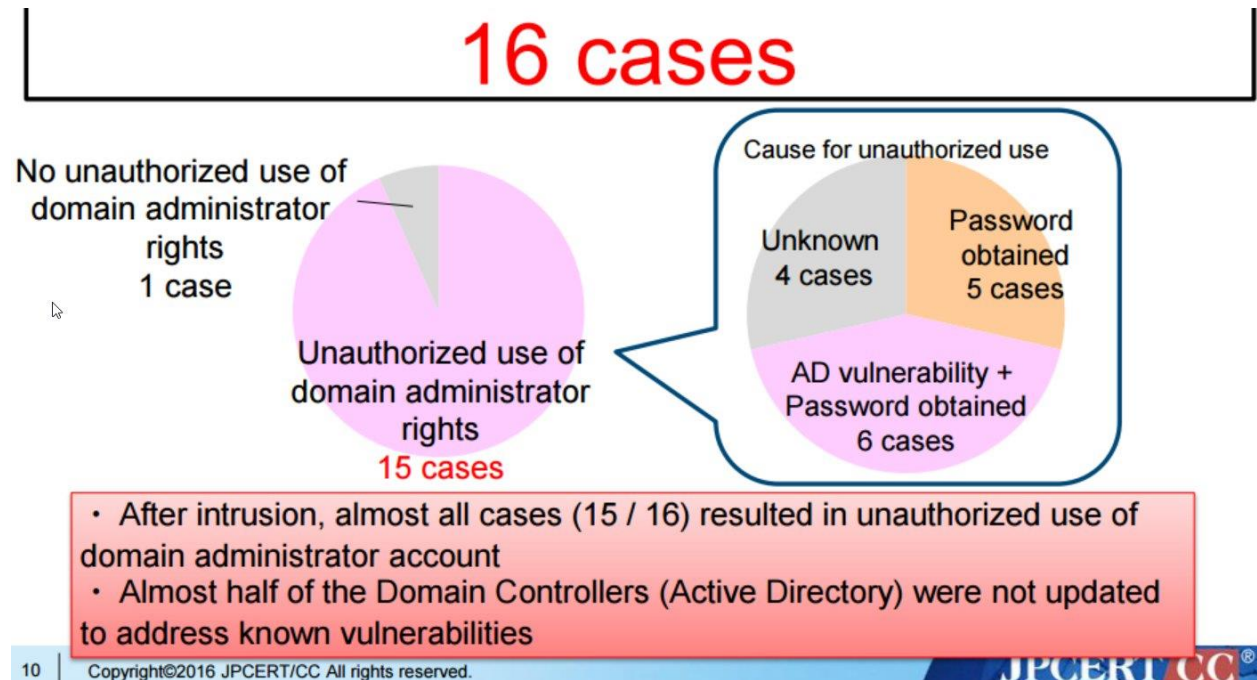


Figure 14 In many advanced attacks incidents, attackers use known vulnerabilities<sup>25</sup>

Patching the vulnerabilities related to the original “Evil Maid” attack, would prevent the “remote Butler” attack too.

Note that organizations which had deprioritized the patching of these “Evil Maid” related vulnerabilities, due to the fact that the “Evil Maid” attack required physical access, need to reassess their prioritization now, as “Remote Butler” attack makes these vulnerabilities exploitable remotely over the network too.

### Hardening

By securely configuring the protocols relevant to the “Remote Butler” attack (RDP) and the “Evil Maid” attack (Kerberos), defenders can avoid these attacks in the first place.

<sup>25</sup> <https://www.first.org/resources/papers/conf2016/FIRST-2016-105.pdf>

## RDP hardening with Network Level Authentication (NLA)

“Network Level Authentication is an authentication method that can be used to enhance RD Session Host server security by requiring that the user be authenticated to the RD Session Host server before a session is created.

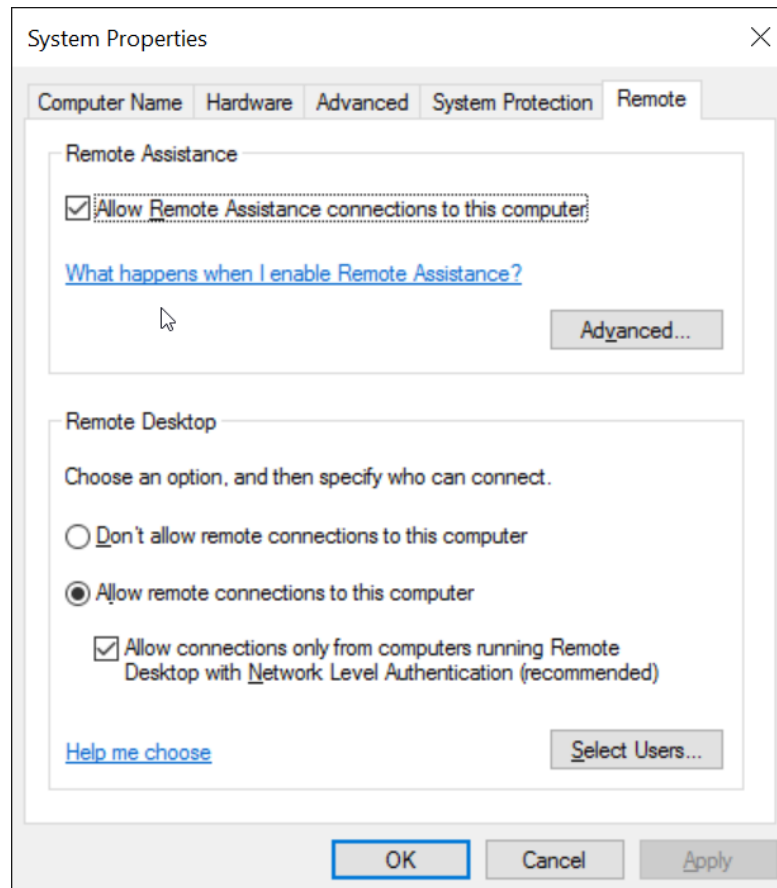


Figure 15 Configuring RDP NLA

Network Level Authentication completes user authentication before you establish a remote desktop connection and the logon screen appears. This is a more secure authentication method that can help protect the remote computer from malicious users and malicious software. The advantages of Network Level Authentication are:

- It requires fewer remote computer resources initially.

- The remote computer uses a limited number of resources before authenticating the user, rather than starting a full remote desktop connection as in previous versions.”<sup>26</sup>

Under this setting, each remote user must authenticate and get authorization **prior** to the establishment of the RDP session. This will prevent the “Remote Butler” attackers to maneuver from a non-domain joined machine to a domain joined machine as the RDP would require them to domain authenticate them beforehand.

### Kerberos Armoring

Kerberos Armoring is a standardized extension (RFC 6113<sup>27</sup>) of the Kerberos protocol. It adds security to the user and DC’s Kerberos protocol conversation, by building on the previously established trust between the machine and the DC (Domain Trust Relationship).

Specifically, Kerberos Armoring adds:

- Authenticated Kerberos Error Messages – Error message that previously couldn’t have been authenticated, are now encrypted with a key that depends on the previous machine to DC authentication
- Additional protection for the user’s authentication – the user’s encrypted timestamp is now encrypted not only by the user’s key which is derived from its password which might be weak, but also with a strong key that depends on the previous machine to DC authentication. As a by-product, every Kerberos authentication, including for the change password procedure, must be performed after a Domain Trust is established.

---

<sup>26</sup> [https://technet.microsoft.com/en-us/library/cc732713\(v=ws.11\).aspx](https://technet.microsoft.com/en-us/library/cc732713(v=ws.11).aspx)

<sup>27</sup> <https://tools.ietf.org/html/rfc6113>



Name	Value	Bit Offset	Bit Length	Type
Length	Length: 430	0	32	Kerbero...
Message	KRB_ERROR	32	3440	Kerbero...
Pvno	5 (0x0000000000000005)	96	40	Int64
MsgType	KRB_ERROR(30) (0x000000000000001E)	136	40	MsgType
Stime	2014-03-10T22:10:56.000000	176	152	DateTime
Susec	927784 (0x000000000000E2828)	328	56	Int64
ErrorCode	KDC_ERR_PREAUTH_REQUIRED(25) (0x0000000000000019)	384	40	ErrorCo...
Realm	aorato.research	424	152	String
Sname	krbtgt/aorato.research	576	304	Kerbero...
EData	MethodData{MethodData=[PA-FX-FAST (136)]}	880	2592	Kerbero...
MethodData	[PA-FX-FAST (136)]	0	2528	ArrayVa...
[0]	PA-FX-FAST (136)			Kerbero...
PADATAType	PA-FX-FAST (136) (0x0000000000000088)	64	48	Int64
PadataValue	PA-FX-FAST-REPLY	112	2416	Kerbero...
PADataValue	KrbFastArmoredRep{EncFastRep=EncryptedData{Etype=18,Kvno=not...	0	2352	Kerbero...

Figure 16 A Kerberos Error message with Kerberos Armoring: Encrypted part is highlighted

Each of these properties, breaks the “Evil Maid” attack’s change password via a rogue DC. The target computer will reject the rogue DC’s error message and change password authentication process, as the rogue DC does not know the relevant machine key, required for the Kerberos Armoring.

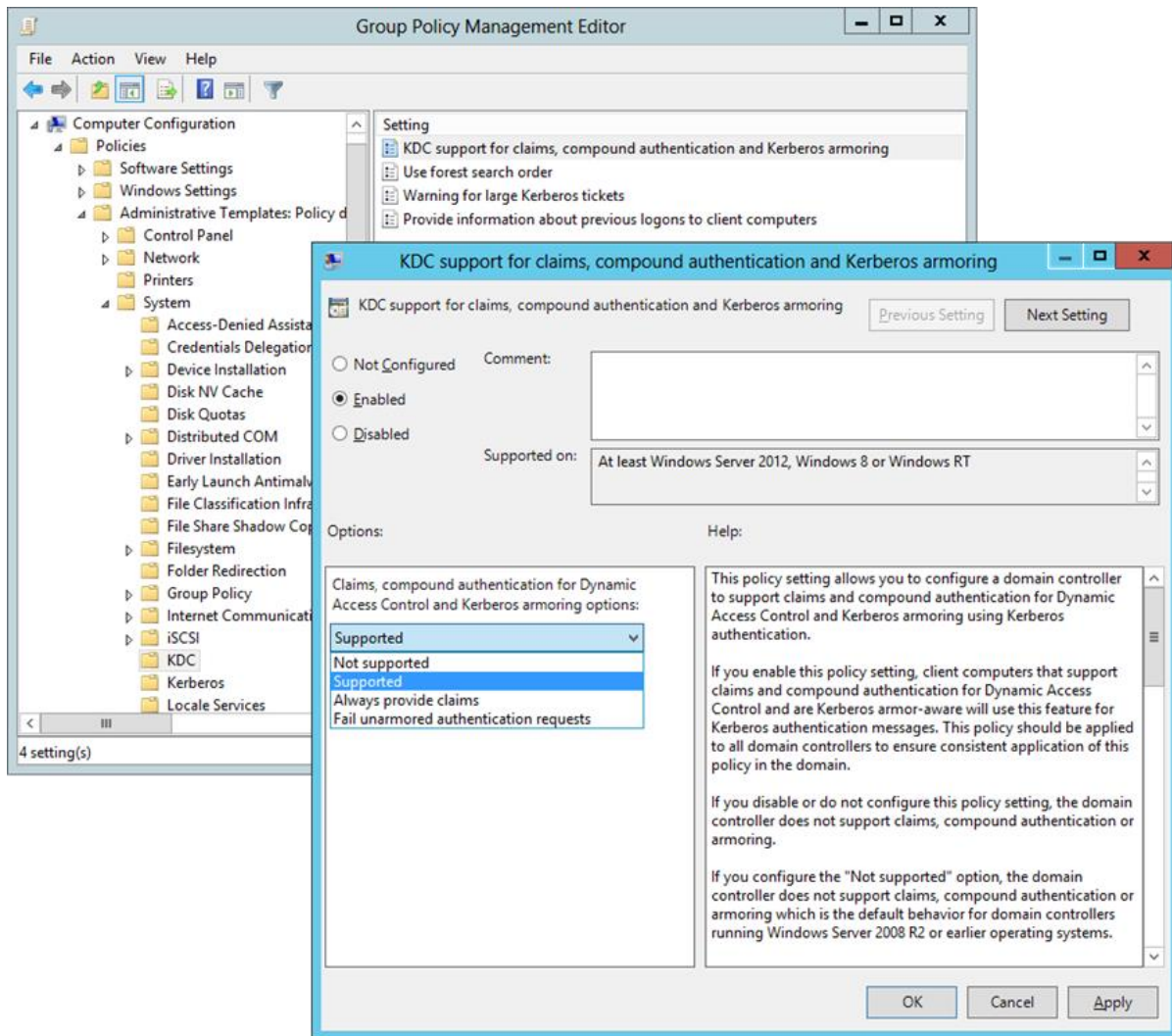


Figure 17 Configuring Kerberos Armoring

Kerberos armoring is supported by Windows since Windows 8 and Server 2012<sup>28</sup>

## Defense-in-Depth

The Cyber Kill-chain model which describes the Advanced Attackers Modus Operandi, carries an implicit promise for defenders. The Kill-chain, as any other chain, is only as strong as its weakest link. As a result, should the defenders discover the attackers' doings in **ANY** part of their attack, the attack can be defeated.

<sup>28</sup> [https://technet.microsoft.com/en-us/library/hh831747\(v=ws.11\).aspx](https://technet.microsoft.com/en-us/library/hh831747(v=ws.11).aspx)

Therefore, by placing some relevant detection capabilities in advance within the network, aiming to detect as many of the different Cyber Kill-chain steps as possible, defenders can remain safe. Even if defenders are unprepared for a certain attack (e.g. the “Remote Butler” attack) which impacts certain steps of the Cyber Kill-chain, they still have enough opportunities to catch the attack campaign in other steps, and thwart the attack altogether.

## Conclusions

As we've seen, the "Evil Maid" attack is an elegant attack, however it has a major drawback: it requires physical access to the target machine.

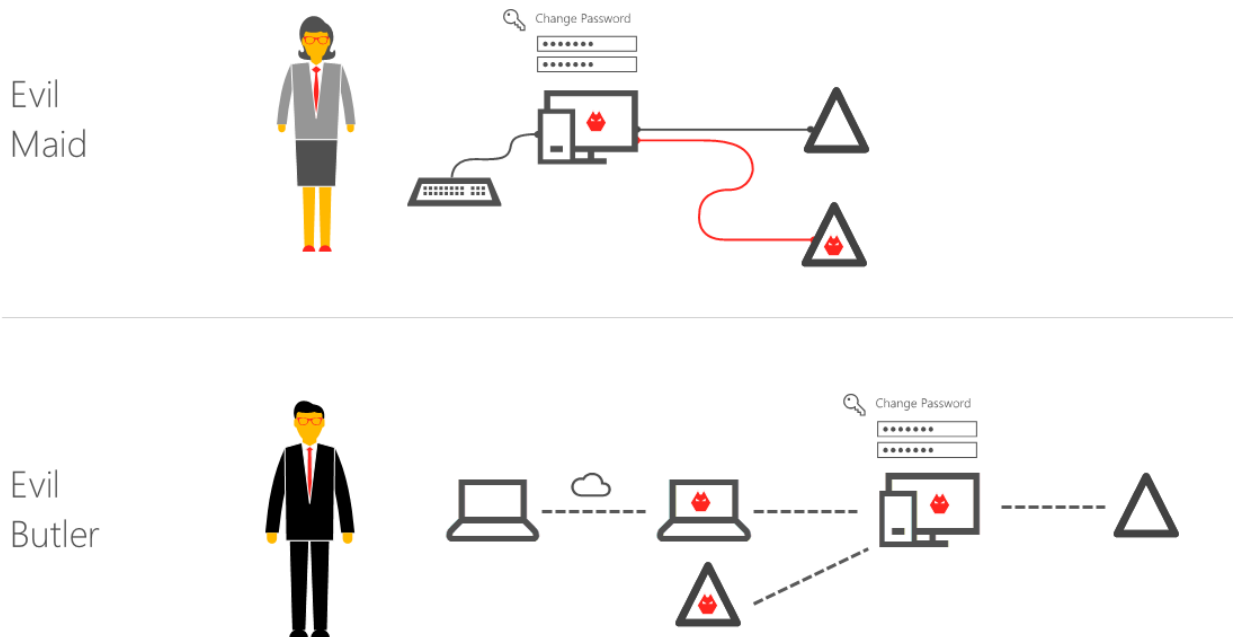


Figure 18 The "Evil maid" vs. the "Remote Butler" attack

This paper introduces the following new findings:

- A new attack, the "Remote Butler", which extends the original "Evil Maid" attack into the network-oriented scenario, which makes it relevant to APT scenarios, including the novel contributions of:
  - Network access via RDP
  - Rogue DC as a malware's payload
  - Network connectivity to the rogue DC with some network manipulations
  - Non-trivial Domain Credentials extraction from memory
  - Clean up method to clean the Cache Credentials store and erase attack traces from the victim machine
- A previously undiscussed internal network reconnaissance method via RDP, which may use either in conjunction with the "Remote Butler" attack or independently

- A previously undiscussed general mitigation for both “Evil Maid” and “Remote Butler” attacks, by using the standard “Kerberos Armoring” hardening feature.

We had shown some practical protection solution against such attacks, by applying patches, protocol hardening and most importantly having a Defense-in-Depth policy against the cyber Kill-chain.