WALLSEC

Network File System Security

Sergej Schmidt @ MRMCD2024
06.10.2023

My Background

- Cyber Cyber Sailor @ wallsec.de
- “Linux-Guy”
- Some blue team experience
- Mostly offensive security
- NFS:
  - Ops with Kerberos
  - Architecture consulting

Find me here: @disasmwinnie@social.linux.pizza

Motivation

- Many frustrating discussions about NFS shares
- No understanding among decision makers
- Network shares must die (or implement Kerberos)

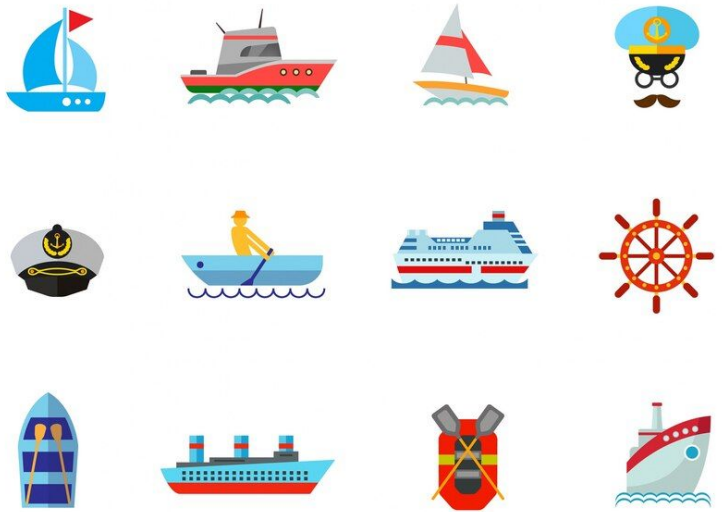
Reality (2023 film)
https://en.wikipedia.org/wiki/File:Reality_poster.jpg

Better Title:

Network File System Security In a Corporate Reality

Todays Boat Trip

- NFS Intro
- Security in a nutshell
- Corporate network reality
- Mitigating measures & reality
  - root_squash
  - NFSv4!?
  - Kerberos



Overview

- Well, it's a Network File System
- NFSv2 (1989)
- NFSv3 (1995)
  - 64bit file size :)
  - asynchronous writes
- NFS v4 (2000)
  - No need for portmap, port 2049 only
  - Better write-access
  - Kerberos-support
  - v4.1 (2010) and v4.2 (2016)

User Cases

- UNIX home folders
- Software delivery in semi-modern world
  DEV → QA →PROD
- Arbitrary file exchange

NFS-Server

```
[root@ziegelstein ~]# tail -2 /etc/exports
/exports/homes 192.168.66.150(rw,no_root_squash)
/exports/homes 192.168.66.152(rw,root_squash)
[root@ziegelstein ~]# systemctl restart nfs-server
[root@ziegelstein ~]# exportfs -v
/exports/homes  192.168.66.150(sync,wdelay,hide,no_subtree_check,sec=sys,rw,secure,
no_root_squash,no_all_squash)
/exports/homes  192.168.66.152(sync,wdelay,hide,no_subtree_check,sec=sys,rw,secure,
root_squash,no_all_squash)
[root@ziegelstein ~]#
```


NFS-Client

```
root@mrmcd-vm1:~# showmount -e 192.168.66.1
Export list for 192.168.66.1:
/exports/homes 192.168.66.152,192.168.66.150
root@mrmcd-vm1:~# grep "nfs-homes" /etc/fstab
192.168.66.1:/exports/homes /nfs-homes nfs
root@mrmcd-vm1:~# id myuser1
uid=2000(myuser1) gid=2000(myuser1) groups=2000(myuser1)
root@mrmcd-vm1:~# id myuser2
uid=3000(myuser2) gid=3000(myuser2) groups=3000(myuser2)
root@mrmcd-vm1:~# ls -la /nfs-homes
total 16
drwxrwxr-x  4 root    root    4096 Oct  5 18:37 .
drwxr-xr-x 21 root    root    4096 Oct  5 18:15 ..
drwxr-x---  2 myuser1 myuser1 4096 Oct  5 18:36 myuser1
drwxr-x---  2 myuser2 myuser2 4096 Oct  5 18:37 myuser2
root@mrmcd-vm1:~# mount | grep nfs-homes
192.168.66.1:/exports/homes on /nfs-homes type nfs4 (rw,relatime,vers=4.
2,rsize=1048576,wsize=1048576,namlen=255,hard,proto=tcp,timeo=600,retran
s=2,sec=sys,clientaddr=192.168.66.150,local_lock=none,addr=192.168.66.1)
root@mrmcd-vm1:~#
```

NFS-Client

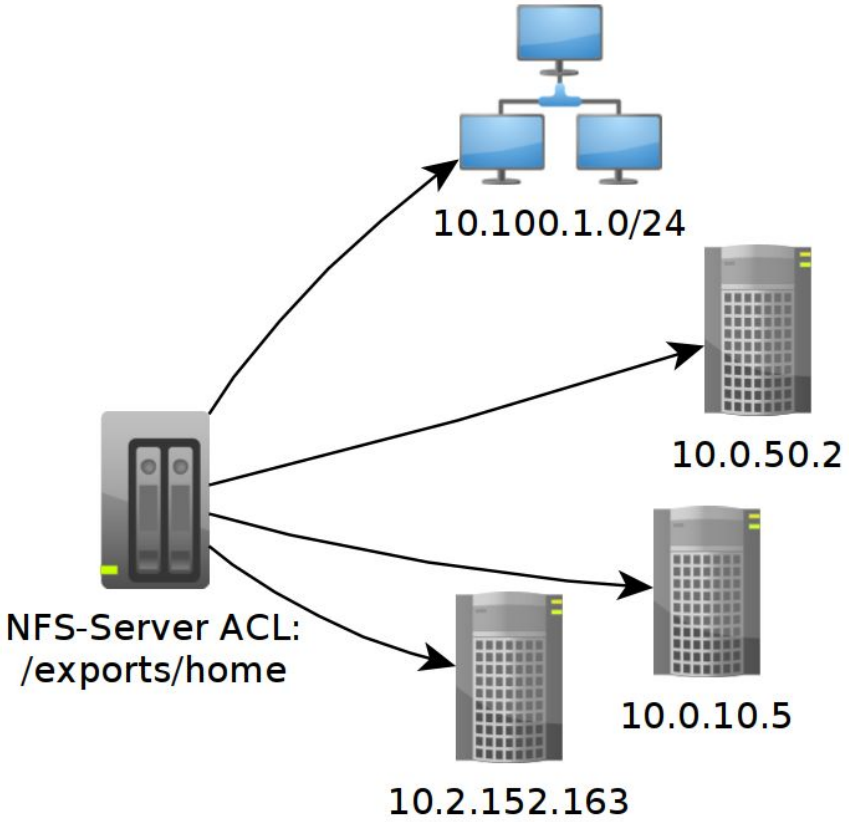
```
victim@mrmcd-vm2:~$ ls -la /nfs-homes/myuser1
ls: cannot open directory '/nfs-homes/myuser1': Permission denied
victim@mrmcd-vm2:~$ sudo su -
root@mrmcd-vm2:~# ls -la /nfs-homes/myuser1
total 20
drwxr-x--- 2 myuser1 myuser1 4096 Oct  5 22:52 .
drwxrwxr-x 4 root    root    4096 Oct  5 18:37 ..
-rw------- 1 myuser1 myuser1    0 Oct  5 18:36 .bash_history
-rw-r--r-- 1 myuser1 myuser1  220 Jan  6  2022 .bash_logout
-rw-r--r-- 1 myuser1 myuser1 3784 Oct  5 22:52 .bashrc
-rw-r--r-- 1 myuser1 myuser1  807 Jan  6  2022 .profile
root@mrmcd-vm2:~#
```

Security in a nutshell

- IP/subnet based ACLs
- UID/GID are passed by the NFS-Client
- **FULL trust in NFS-Client**

- Risk management:
  - 1000 NFS-clients
  - 1 compromised NFS-client
    → compromised NFS-Share/Export
    → creds/secrets/code exec in .bashrc

Security in a nutshell



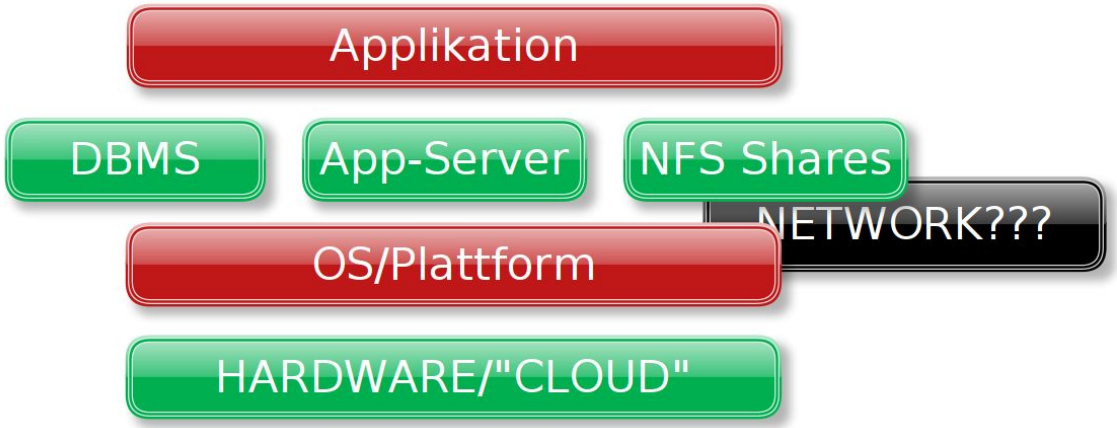
Corporate network reality

Corporate network reality

Corporate network reality - Networking Know-How

Service Layer Cake (™)

- Commodity services
- Made “consumable”
- Ops by different teams
- Moderate knowledge of each other’s stack



Corporate network reality - Networking Know-How

Mitigating measures & reality

But we use $MITIGATION... we secure...

Mitigating measures & reality

root_squash from man 5 exports

Map requests from uid/gid 0 to the anonymous uid/gid. [...]

Mitigating measures & reality

Simple bypass of ***root_squash***

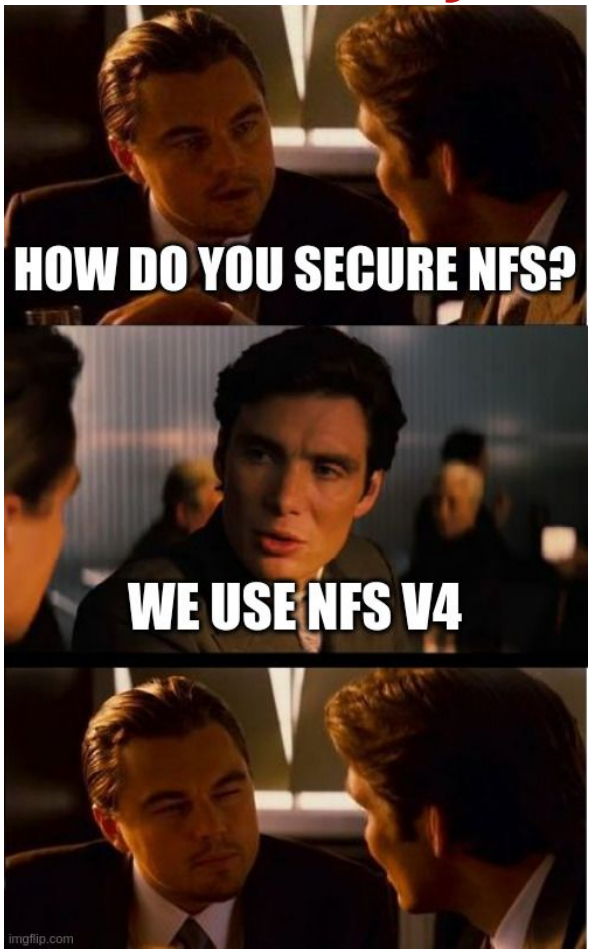
- Create desired UID locally
- Remember: full trust towards NFS-client

Mitigating measures & reality

Simple bypass of ***root_squash***

```
root@mrmcd-vm2:~# ls -la /nfs-homes/myuser2/
ls: cannot open directory '/nfs-homes/myuser2/': Permission denied
root@mrmcd-vm2:~# ls -la /nfs-homes/
total 16
drwxrwxr-x  4 root    root    4096 Oct  5 18:37 .
drwxr-xr-x 21 root    root    4096 Oct  5 18:14 ..
drwxr-x---  2 myuser1 myuser1 4096 Oct  5 18:36 myuser1
drwxr-x---  2    3000    3000 4096 Oct  5 18:37 myuser2
root@mrmcd-vm2:~# useradd -m --uid 3000 someuser
root@mrmcd-vm2:~# su - someuser
$ ls -la /nfs-homes/myuser2/
total 20
drwxr-x--- 2 someuser someuser 4096 Oct  5 18:37 .
drwxrwxr-x 4 root     root     4096 Oct  5 18:37 ..
-rw-r--r-- 1 someuser someuser  220 Jan  6  2022 .bash_logout
-rw-r--r-- 1 someuser someuser 3771 Jan  6  2022 .bashrc
-rw-r--r-- 1 someuser someuser  807 Jan  6  2022 .profile
$
```

Mitigating measures & reality



NFSv4

[...], mandates strong security, [...]

https://en.wikipedia.org/wiki/Network_File_System#NFSv4

NFSv4

From Abstract:

[...], NFSv4 provides strong security through the use of either Kerberos V5, SPKM-3, or LIPKEY. [...]

2005 USENIX Annual Technical Conference (USENIX ATC 05)
By Spencer Shepler
https://www.usenix.org/conference/2005-usenix-annual-technical-conference/nfsv4

NFSv4

[...], NFSv4 is inherently more secure than NFSv3. For example, NFSv4 security is normally based on usernames, not user ID's. The result is it's more difficult for an intruder to spoof credentials to gain access to data on an NFSv4 server. [...]

NFSv4 also includes options to make it even more secure. [...]

https://community.netapp.com/t5/Tech-ONTAP-Blogs/NFSv3-and-NFSv4-What-s-the-difference/ba-p/441316#toc-hId-1020336704

NFSv4

One area of great confusion is that many believe that NFSv4 ***requires*** the use of strong security.

The NFSv4 specification simply states that ***implementation*** of strong RPC security by servers and clients is ***mandatory***, not the ***use*** of strong RPC security.

USENIX ;login: February 2012, Volume 37, Number 1
by Alex McDonald
https://www.usenix.org/system/files/login/articles/mcdonald_0.pdf

NFSv4

- NFSv4 GSS-API / Kerberos* **implementation** is mandatory
- But usage is **optional!1!!!!!!!!!!**
- Username mapping works only with Kerberos
  - irrelevant as security feature
  - Without Kerberos still UID-/GID-based security (none)

* Btw, never seen SPKM-3 or LIPKEY in the wild

WALLSEC

Kerberos - SIMPLIFIED (a lot)

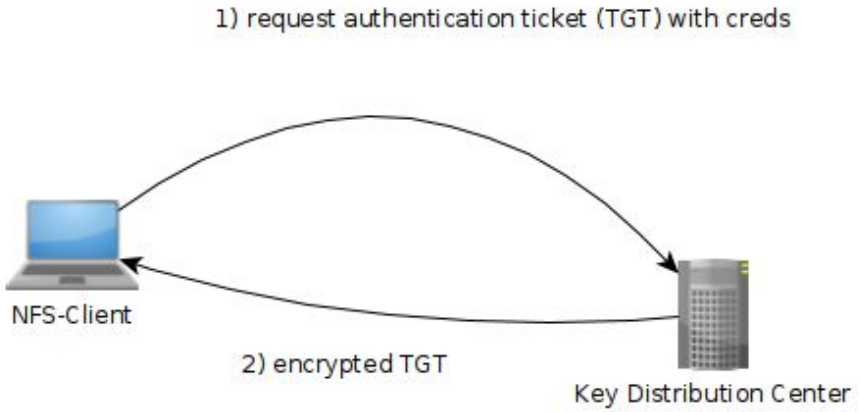
Kerberos

- Three headed dog (3 systems involved)
  - NFS-Client
  - NFS-Server
  - KDC (Key Distribution Center)
- KDC is actually two components
  - Authentication Server
  - Ticket Granting Server (TGS)

Kerberos

- Step 1: Send creds

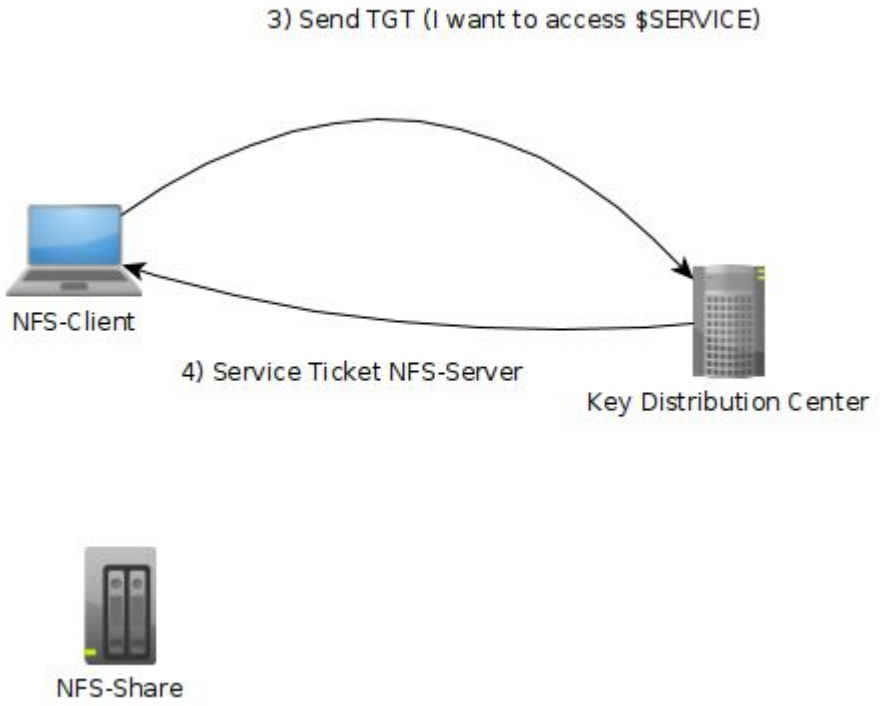
- Step 2:
  - KDC validates creds
  - Sends back
    Ticket Granting Ticket (TGT)



Kerberos

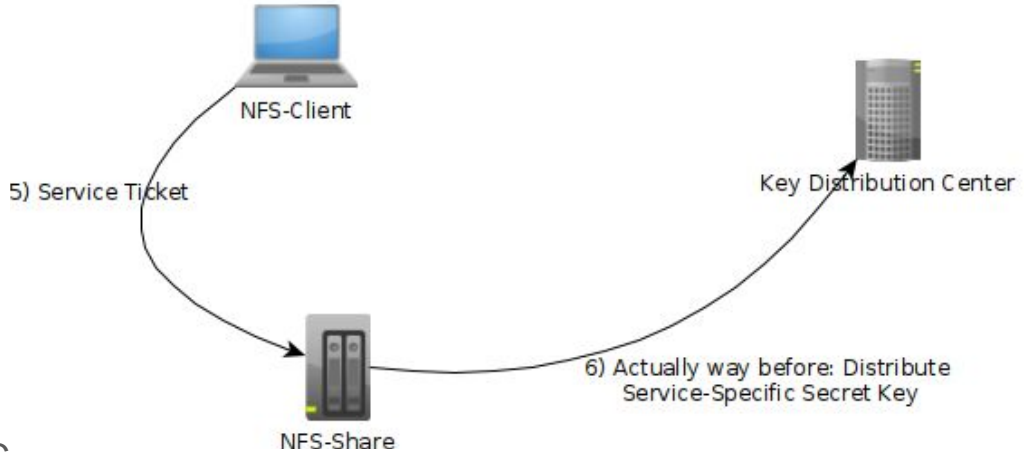
- Step 3: Send TGT and ask for Service Ticket (NFS-Server)

- Step 4: KDC sends Service Ticket for NFS-Server



Kerberos

- Step 5: Send Service Ticket



- Step 6:
  - Can decrypt Service Ticket
  - NFS-Server has Secret Key from KDC
  - KDC encrypted the Service Ticket with that secret before
  - Secret Key for $service (keytab)
    must be

- Step 7: Profit

Kerberos

“I just explained Kerberos to a colleague.
And, immediately forgot how it works myself.”

Somebody on Twitter

Kerberos

- Complexity kills
  - Lack of operational experience
  - Reverse DNS entry
  - Strict time sync
- Kills SSH Public-Key-Auth
- Not really common in Unix/Linux environment

Kerberos

But even Microsoft can do it!

→ First class citizen since Windows 2000

→ Out-of-the-box experience

Conclusion

- NFS Share (without Kerberos) must die!
- No secure network shares without Kerberos
- Go into the world and fuck up NFS Shares (and tell people about it, so they can fix it)

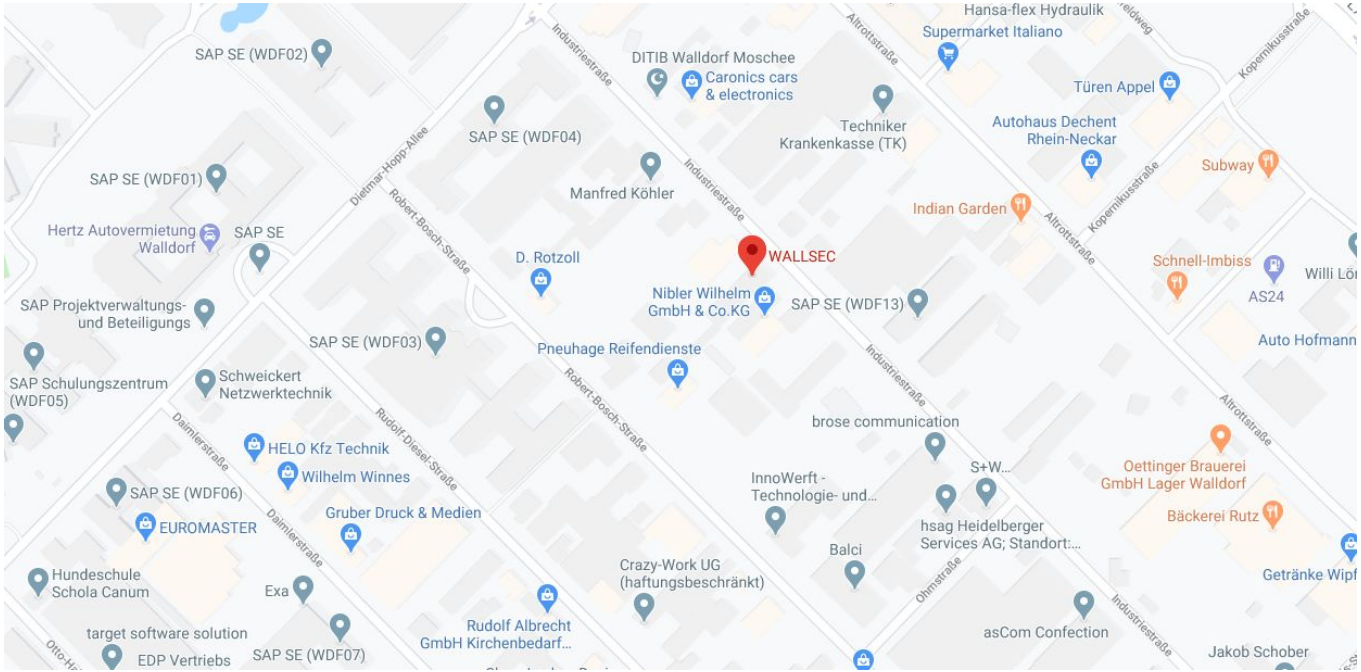
‘\0’

Slides: https://github.com/WALLSEC/blog-attachments

Find me here: @disasmwinnie@social.linux.pizza

Contact

WALLSEC GmbH
Industriestrasse 44
69190 - Walldorf
Germany
Email: contact@wallsec.de



WALLSEC GmbH. All Rights Reserved.