

Separating key management from file system security

D. Mazieres, M. Kaminsky, M. Kaashoek, E. Witchel

CPSC 508 - Operating Systems

November 29, 2006

Brought to you by...

Presented by

- ▶ Billy Cheung
- ▶ Alfred Pang

About the paper

- ▶ Proceedings of the seventeenth ACM symposium on Operating systems principles, 1999, 124-139.

Project webpage

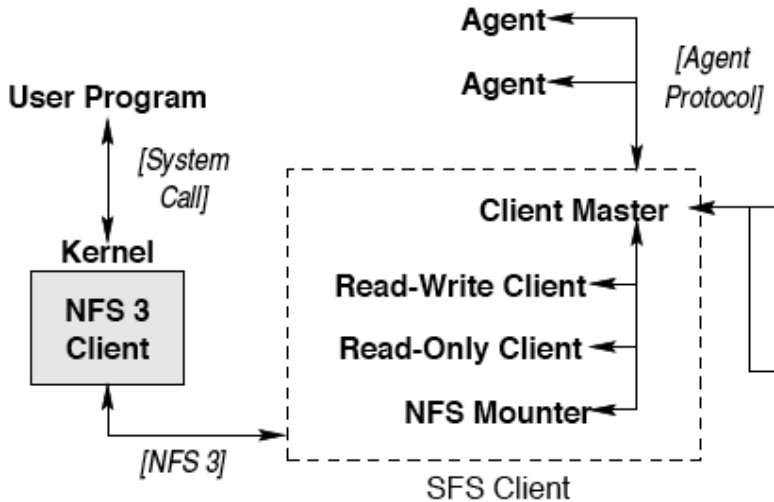
- ▶ <http://www.fs.net/sfswww/>

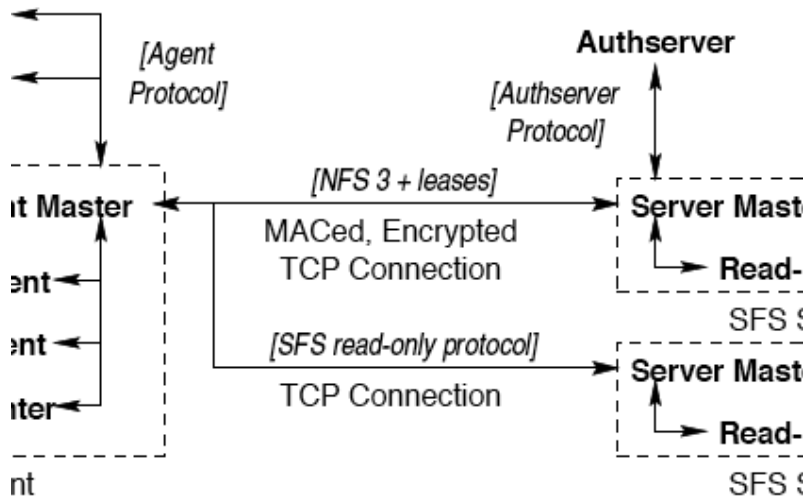
Motivation

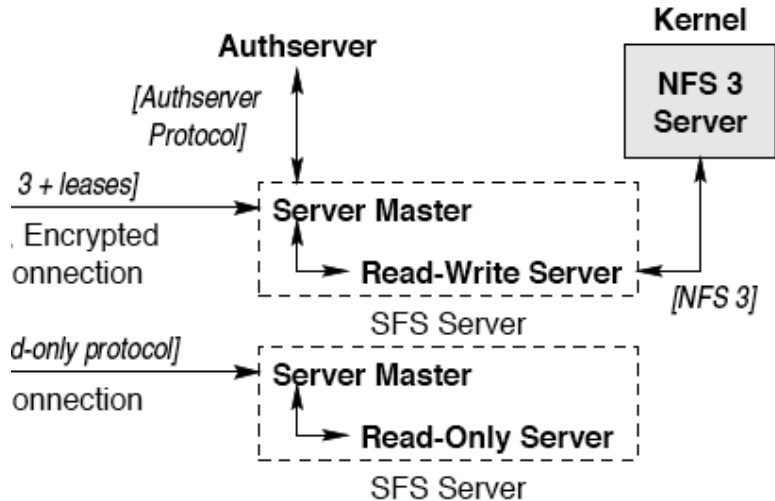
- ▶ How many logins do I have to keep in my head!?!?!?!?
- ▶ New server automatically = new user management
- ▶ Untrustworthy networks

Introducing SFS

- ▶ Self-certifying File System
- ▶ User authentication is decoupled from file system
- ▶ Portable implementation (using NFS 3 for access)







How to authenticate users?

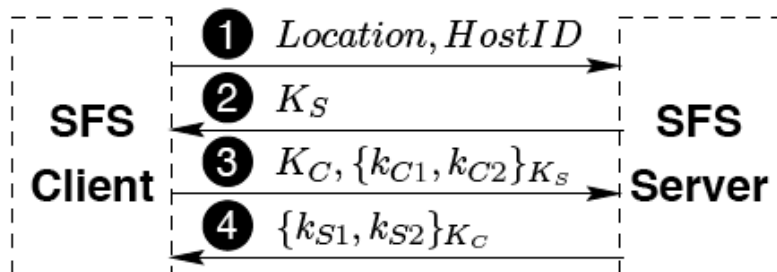
- ▶ sfsagent - keeps user's private keys (signs authentications etc.)
- ▶ authserv - maps public keys to user credentials (server side)

Syntax

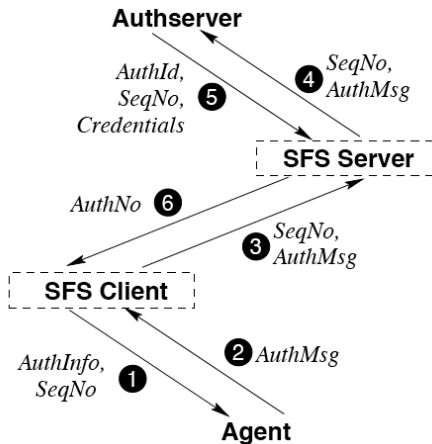
```
redlab:/sfs/new-york.lcs.mit.edu
[redlab 1] sfsagent dm@new-york
Passphrase for dm@new-york.lcs.mit.edu:
[redlab 2] cd /sfs/new-york.lcs.mit.edu
sfsagent: sfsrwc1: new-york.lcs.mit.edu:85xq6pznt4mgfvj4mb23x6b8adak55ue (1)
[redlab 3] pwd
/sfs/new-york.lcs.mit.edu:85xq6pznt4mgfvj4mb23x6b8adak55ue
[redlab 4] █
```

- ▶ Negotiate session keys
- ▶ Server constructs authentication request for client to sign
- ▶ Client signs, and is good to go

SFS key negotiation



SFS user authentication



Cryptography used

- ▶ Rabin for public key cryptosystem
- ▶ SHA-1 message authentication
- ▶ Blowfish to encrypt file handles
- ▶ ARC4 (or RC4) - stream cipher

Performance and Evaluation

File System	Latency (μ sec)	Throughput (Mbyte/sec)
NFS 3 (UDP)	200	9.3
NFS 3 (TCP)	220	7.6
SFS	790	4.1
SFS w/o encryption	770	7.1

System	Time (seconds)
Local	140
NFS 3 (UDP)	178
NFS 3 (TCP)	207
SFS	197

Figure 7: Compiling the GENERIC FreeBSD 3.3 kernel.

More benchmarks

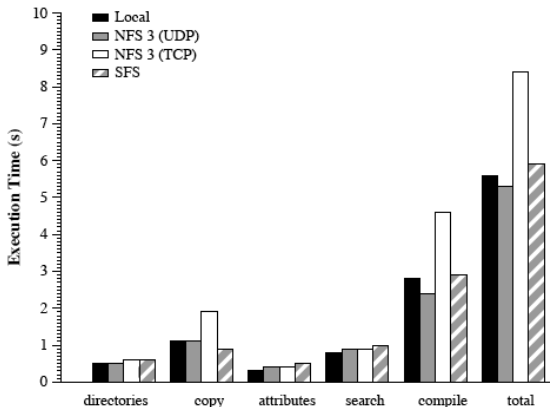


Figure 6: Wall clock execution time (in seconds) for the different phases of the modified Andrew benchmark, run on different file systems. Local is FreeBSD's local FFS file system on the server.

More benchmarks

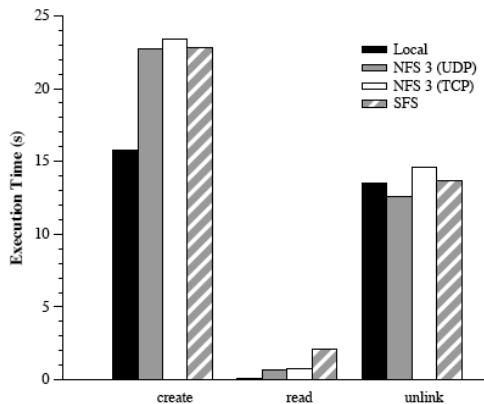


Figure 8: Wall clock execution time for the different phases of the Sprite LFS small file benchmark, run over different file systems. The benchmark creates, reads, and unlinks 1,000 1 Kbyte files. Local is FreeBSD's local FFS file system on the

More benchmarks

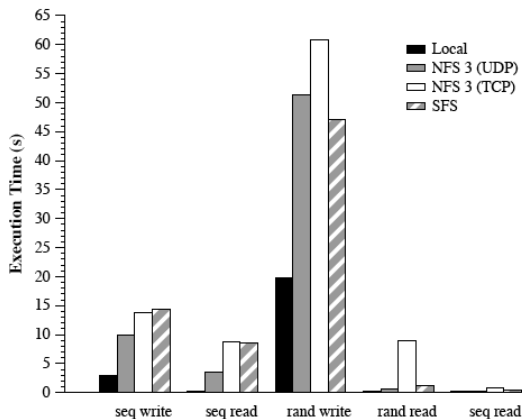


Figure 9: Wall clock execution time for the different phases of the Sprite LFS large file benchmarks, run over different file systems. The benchmark creates a 40,000 Kbyte file and reads and writes 8 Kbyte chunks. Local is FreeBSD's local

Vulnerabilities

- ▶ NFS server - file handles, open ports
- ▶ Other usual vulnerabilities - implementation bugs, insecure client, etc.

My questions for you

- ▶ What's the most expensive part of the end-to-end sequence?
- ▶ Can I also use this for logins and other user authentication?
- ▶ What do we use around UBC? (CWL?)