



# Network Storage Security

---

Brett Willman

Sukwoo Kang

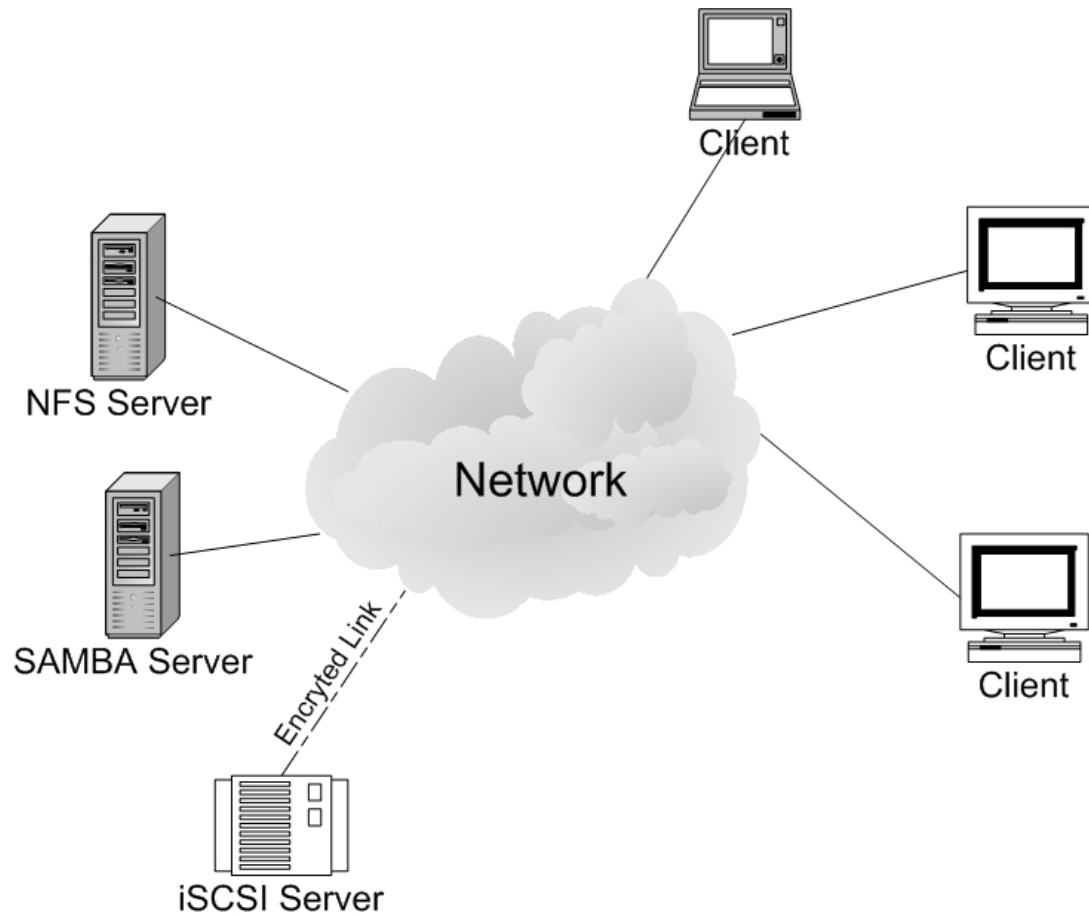


# Presentation Overview

---

- Define Network Storage
- Introduce a framework for evaluating storage system security
- Explain innovative new technology
  - Snapdragon – Block-level security
- Apply framework to Snapdragon
- Conclusion and Questions

# General Concept





# Storage Security Framework

---

- Classifies current storage techniques
- Provides ability to compare security vs. performance
- Components include:
  - Players
  - Attacks
  - Security Primitives
  - Granularity of Protection
  - User Inconvenience



# Two Encryption Schemes

---

- Pre-computed encryption
  - Data stored in encrypted form
  - Server is not burdened with encryption and decryption
  - Requires longer lived keys
- On the wire encryption
  - Data is encrypted and decrypted before and after it is sent on the wire
  - Requires session keys for transfer
  - Server and client bears CPU burden



# Players

---

- Owners have full control of specific data
- Readers – read data delegated by owners
- Writers – modify data delegated by owners
- Wire – transfer between players
- Storage servers – file server or disks
- Group servers – authenticate and authorize players
- Namespace servers – directory lookups and traversal of namespaces
- Adversary – performs unauthorized actions



# Attacks

---

- Adversary on the wire
- Adversary on the servers
- Revoked user on the server
- Adversary colluding with storage server
  - Deletion of file by UNIX file system
- Adversary colluding with group server
  - Corrupting metadata server
- Adversary colluding with readers/writers



# Core Security Primitives

---

- Authentication
  - Distributed authentication
  - Centralized authentication
- Authorization
  - Server-mediated
  - Owner-mediated



# Security Primitives (cont...)

---

- Securing data on the wire
  - Encryption of data
  - MACs to ensure data integrity
- Securing data on the disk
  - May not trust the server
  - Typically uses symmetric cipher



# Security Primitives (cont...)

---

- Key Distribution
  - Group-server
  - Owner-handled
- Revocation
  - Player's access is revoked
  - On encrypted disks
    - Aggressive re-encryption
    - Lazy re-encryption
    - Periodic re-encryption



# Granularity of Protection

---

- Group membership
  - Distributed group membership – owners explicitly determine authorization and key distribution
  - Centralized group membership – owners delegate key distribution (NFS w/ NIS)
- Granularity of Keys
  - Short-term keys decrease window of vulnerability
  - Long-lived keys – typically last across sessions and may be same across multiple players (CFS)
    - Impacts number of keys



# User Inconvenience

---

- Convenient
  - single login password
- Inconvenient
  - multiple passwords for different services
  - passwords re-entered frequently
- Very Inconvenient
  - resources protected at very low level
  - per-file password, etc



# Snapdragon

---

Marcos K. Aguilera et al.

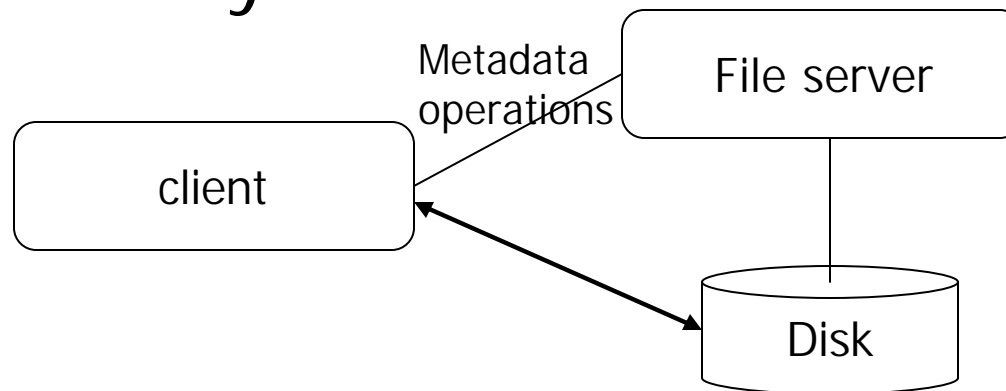
Block-Level Security for Network-Attached Disks

*The 2nd USENIX Conference on File and Storage Technologies  
(FAST '03)*

# Introduction

---

- Network-attached disks (NADs)
  - storage devices that accept block read/write requests over the network
- NAD file system





# Motivation

---

- Problems
  - Commercial NADs do not provide security
  - Must use in separate LAN for storage (Storage Area Network – SAN)
- Existing approaches
  - Requires major changes to file system
- Provide *simple* method for adding block-level security to NADs



# Block-based security

---

- Assumption
  - Server and disks can be trusted
  - Clients can be compromised
    - NADs must verify requests from client
  - Network is not secure
    - Traffic must be encrypted

# Basic capability scheme

- Capability

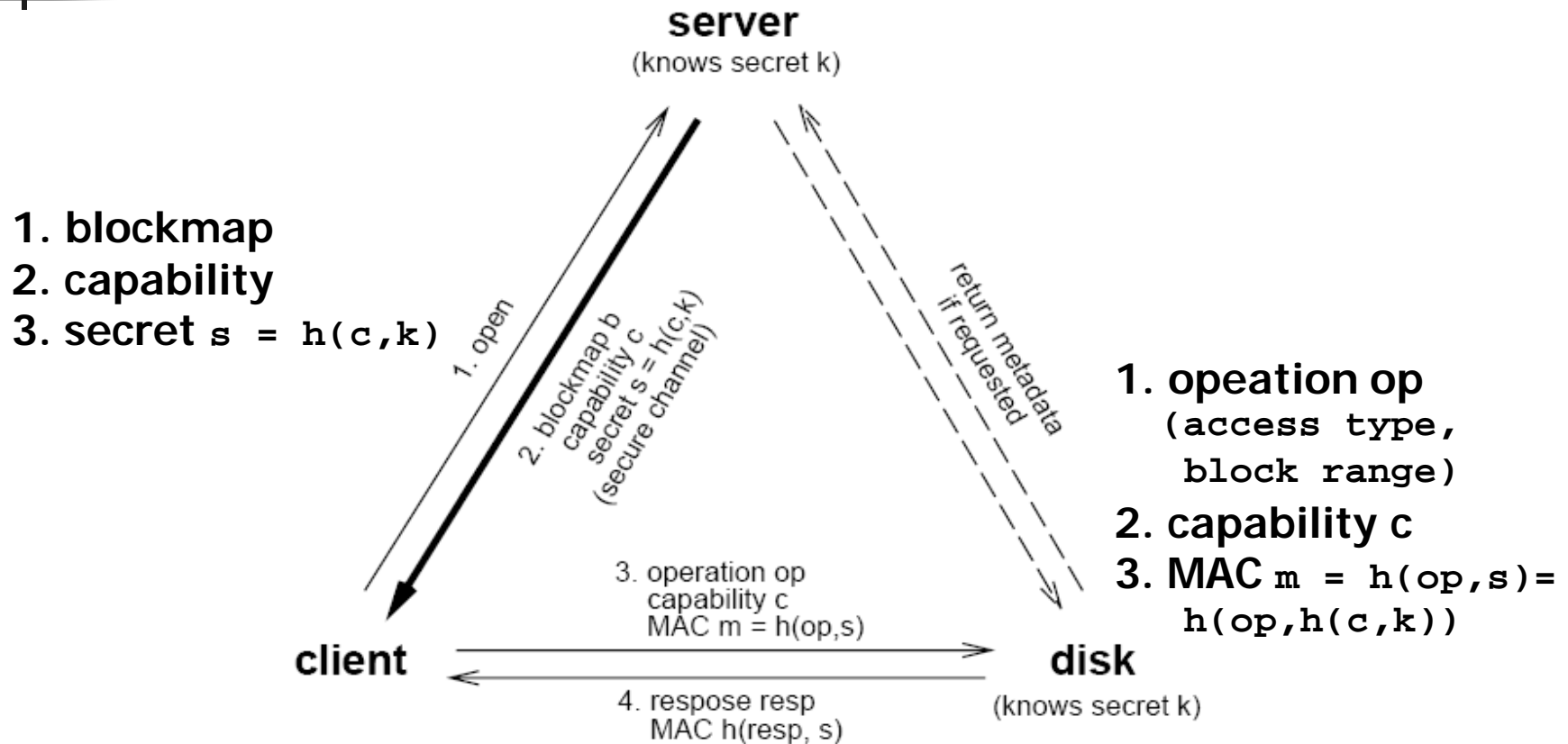
- self-descriptive certificate that grants a specified type of access to parts of disk when used with an associated secret

group ID	capability ID	disk ID	extents	mode
----------	---------------	---------	---------	------

- Secret

- Generated using message authentication code (MAC)  $\rightarrow \text{mac} = h(\text{capability}, \text{key})$

# Basic capability scheme (cont.)



This picture is from Marcos K. Aguilera et.al., Block-Level Security for Network-Attached Disks, The 2nd USENIX Conference on File and Storage Technologies (FAST '03)



# Revoking capabilities

---

- Revocation
  - Required when a client should no longer have the access granted by a previously issued capability
- Constraints
  - It must be *memory efficient* to be implemented in existing NADs



# Revoking capabilities (cont.)

---

- Earlier approach
  - Use object version numbers for revocations
  - It is problematic for blocks
- New approach
  - Assign capability ID's to each capability
    - Using a bit vector

# Revoking capabilities (cont.)

- Burstiness problems
  - Run out of capability IDs
  - Change  $k$
  - All clients must request new capabilities from server

- Capability groups

- Place capabilities into groups, and invalidate groups

Group ID		Revoked capability ID's (bitmap)
Index	Counter	
0	10	100010001000...
1	5	001111011010...
2	14	111011111000...
⋮	⋮	⋮
63	3	000011010111...



# Preventing replay attacks

---

- Replay attack
  - Replay requests that have already been sent to a disk
- Earlier approach
  - Keep per-client state
- New approach
  - To remember recent requests, employs a *Bloom filter*
  - Low cost in memory and computation



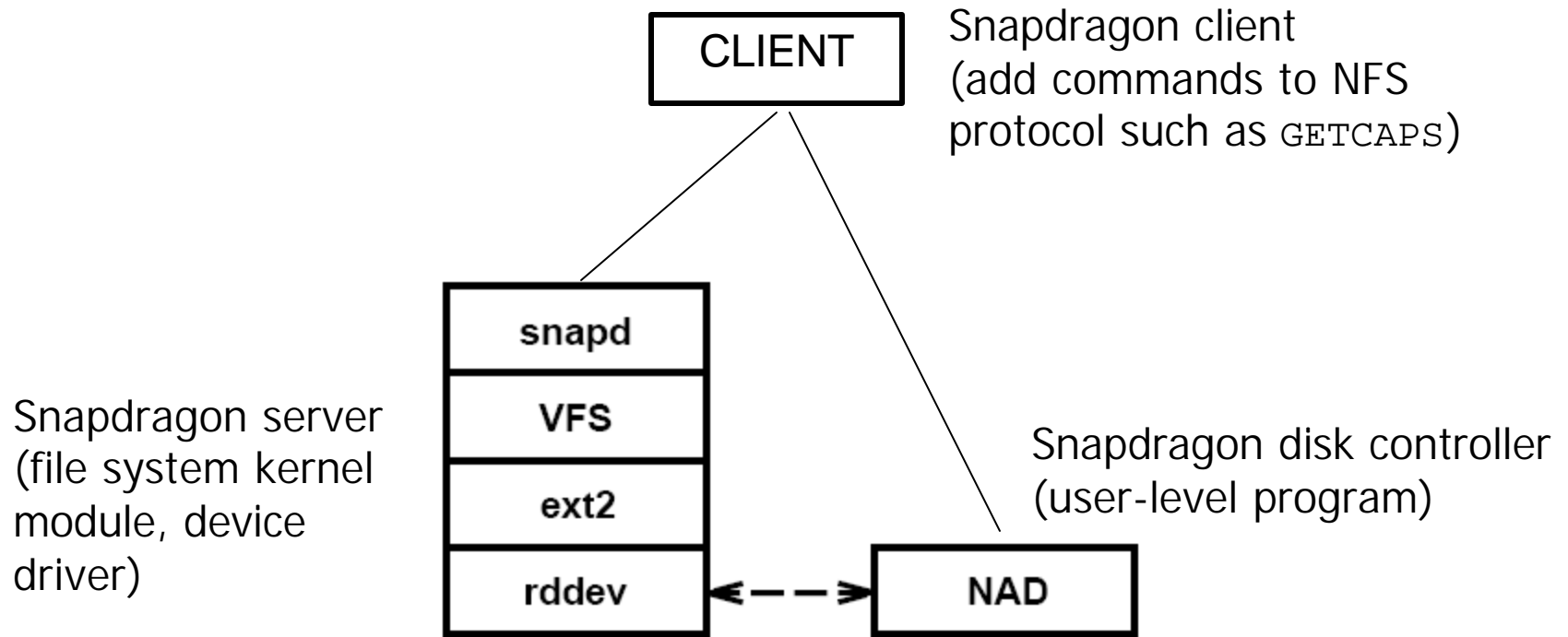
# Preventing replay attacks (cont.)

---

- Bloom filter
  - Simple space-efficient data structure
  - Performing approximate set-membership queries
    - “is request  $r$  in the filter?”
    - “probably” or “definitely not”

# Implementation

Modified Linux's existing NFS version 2



This picture is from Marcos K. Aguilera et.al., Block-Level Security for Network-Attached Disks, The 2nd USENIX Conference on File and Storage Technologies (FAST '03)



# Summary

---

- New block-based security scheme
  - Requires no changes to the data layout
  - Requires small memory
    - Revocation scheme based on capability groups
    - Replay-detection with Bloom filters
- Reasonable performance
  - Read/write latency increase < 5%
  - Bandwidth decrease < 16%
- Can be used in incorporating security into existing NADs with minimal change



# Applying Framework to Snapdragon

---



# Players

---

- Owners, readers, and writers are differentiated
- Metadata server (snapd)
  - Integrates group server and namespace server
- Storage disks



# Trust Assumptions

---

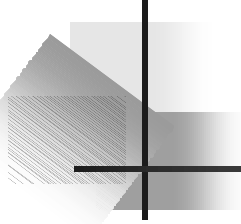
- All messages encrypted on the wire
- Vulnerable to attacks in collusion with storage disks
  - Data is stored in clear
- Vulnerable to attacks in collusion with metadata server
  - Includes all authentication and authorization data
- Owners delegate capability distribution to metadata server
- Storage & Metadata servers assumed trusted



# Security Primitives

---

- Metadata server authenticates and authorizes clients
  - Provides capabilities and secret hash
  - Later verified by storage disks
- Data is encrypted on the wire
- Integrity is guaranteed using a “double MAC” on the operation and client secret
- Revocation is fast through a bitmap vector kept on the storage disks



# Granularity & Convenience

---

- Granularity
  - Session keys generated at log-in
  - Keyed MAC to ensure request integrity
  - “Double MAC” to ensure operation integrity
  - Centralized group management
- User Convenient
  - Single log-in per session
  - Similar to NFS



# References

---

- Erik Riedel, Mahesh Kallhalla, and Ram Swaminathan: *A framework for evaluating storage system security*, Proc. of 1<sup>st</sup> Conference on File and Storage Technologies (FAST), January 2002.
- Marcos Aguilera, Minwen Ji, et al.: *Block-Level Security for Network-Attached Disks*, Proc. Of 2<sup>nd</sup> Conference on FAST, March 2003.