NFSv4 Extension for Integrity Measurement

Chuck Lever
<chuck.lever@oracle.com>

Today's Approach

- This presentation covers `draft-ietf-nfsv4-integrity-measurement-05`
  - This document's `Introduction` is architectural and high level. Today I will complement that with a use case, an interoperability analysis, and operational examples.
- Then we will discuss remaining controversies

Purpose of Integrity Measurement

- Protect file content from creation to use
  - In particular: the content of executables
  - Protects data at-rest and data in-transit; the protection envelope is continuous
  - Thus data is protected during distribution, installation, execution, and archiving

Purpose of This Extension

- Enable transport and storage of IMA metadata for files stored on NFS servers. IMA metadata is transparent to the NFS protocol and the client and server implementations

- Enable installation of IMA-protected executables from NFS clients

- Extend protection from NFS server to end users on NFS clients

- Enable appraisal policy on an NFS client to be different than the server's or the policies on other clients

Global Pre-requisites

- A software vendor *V* generates a key pair K_{public} and $K_{private}$. *V* publishes K_{public} to its customers via a trust authority.
- *V* finalizes a Golden Master of its application *A*.
- *V* generates a checksum, C_A, of the contents of *A*'s executable file, then signs it with $K_{private}$. Call this C_{signed}.
- *V* publishes *A* and C_{signed}.

Local Pre-requisites

- On systems where integrity measurement is used to protect users from corrupted file content, the following is required:
    - A trusted mechanism for storing multiple K_{public}
    - A privileged security module which measures and appraises files
    - A policy for handling appraisal failures

Operation on a Local FS

- A customer installs A in a file on a local filesystem. It stores C_{signed} as an extended attribute of that file.

- A privileged local security module M_{local} computes the checksum of A. Call this C'_A.

- Before A can be executed, M_{local} verifies C_{signed} with K_{public} and confirms that C_A matches C'_A. If either test fails, M_{local} may report the failure in an audit log or prohibit user access, depending on local policy.

Operation on a Remote FS

Current Scenario

- A customer installs *A* in a file on a file server. It installs C_{signed} as an extended attribute of that file. The file access **does not** expose the extended attribute.

- A security module on the file server M_{server} computes the checksum of *A*. Call this C'_A.

- Before *A* can be accessed remotely, M_{server} verifies C_{signed} with K_{public} and confirms that C_A matches C'_A. If either test fails, M_{server} may report the failure in an audit log or prohibit remote access, depending on policy on the server.

Operation on a Remote FS

With NFS extension

- A customer installs *A* in a file on a file server. It installs C_{signed} as an extended attribute of that file. The file access protocol **does** expose the extended attribute.

- A security module on the client, M_{client}, computes the checksum of *A*. Call this C'_A.

- Before *A* can be executed, M_{client} verifies C_{signed} with K_{public} and confirms that C_A matches C'_A. If either test fails, M_{client} may report the failure in an audit log or prohibit user access, depending on policy on the client.

Metadata Interoperability

- Interoperability is defined as the ability for NFS client A to recognize IMA metadata generated on NFS client B or on an NFS server
    - Local IMA appraisers have to continue to recognize metadata generated long ago (backwards compatibility)
    - Local IMA appraisers have to recognize metadata generated from different sources using different checksum and certificate formats (source compatibility)

Issues for Consensus

- Does the document Introduction focus on the right Linux IMA operational details and use cases?
- Are the IMA metadata interoperability concerns adequately covered?
- What is the proper level of permission needed for modifying the extended attribute via NFS?
- Is an error code needed for communicating integrity failure to NFS clients?

Next Steps

- Add a charter milestone including a delivery date target
- More working group review, especially assessing how well the document explains integrity measurement
- More prototype experience. Does the extension provide useful and effective security?
- WGLC

Possible Future Work

- Similar cryptographic protection for file attributes (EVM) would require:
  - NFS protocol support for SMACK access control and file capabilities, which are non-standard
  - Determining how to handle NFSv4 ACLs
  - Exposing FS UUID and list of protected attributes
- Good performance for mutable files