Toward a Threat Model for Storage Systems

Ragib Hasan
rhasan@ncsa.uiuc.edu

Suvda Myagmar
myagmar@ncsa.uiuc.edu

Adam J. Lee
adamlee@ncsa.uiuc.edu

William Yurcik
byurcik@ncsa.uiuc.edu

National Center for Supercomputing Applications (NCSA)
University of Illinois at Urbana-Champaign (UIUC)

ABSTRACT

The growing number of storage security breaches as well as the need to adhere to government regulations is driving the need for greater storage protection. However, there is the lack of a comprehensive process to designing storage protection solutions. Designing protection for storage systems is best done by utilizing proactive system engineering rather than reacting with ad hoc countermeasures to the latest attack du jour. The purpose of threat modeling is to organize system threats and vulnerabilities into general classes to be addressed with known storage protection techniques. Although there has been prior work on threat modeling primarily for software applications, to our knowledge this is the first attempt at domain-specific threat modeling for storage systems. We discuss protection challenges unique to storage systems and propose two different processes to creating a threat model for storage systems: one based on classical security principles (*C*onfidentiality, *I*ntegrity, *A*vailability, *A*uthentication, or CIAA) and another based on the Data Lifecycle Model. It is our hope that this initial work will start a discussion on how to better design and implement storage protection solutions against storage threats.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks**]: General—*Security and Protection*; H.3.4 [**Information Systems**]: Information Storage and Retrieval—*Systems and Software*; D.4.2 [**Software**]: Operating Systems—*Storage Management*

Keywords

storage system, security, threat model

1. INTRODUCTION

Data security breaches, like recent publicized storage losses of private data by many companies, are rightfully calling into question how storage is protected. The risks even increase as enterprises migrate storage from direct attached storage systems to networked storage environments for the logical consolidation of data. New security threats are emerging as storage is increasingly geographically centralized on a few storage arrays or distributed across wide area networks.

Government regulations such as Health Insurance Portability and Accountability Act (HIPAA) [5], which addresses the security and privacy of health data, are also spurring interest in storage security. Financial institutions are constrained by laws like the Sarbanes-Oxley Act [40], Gramm-Leach-Bliley Act [13], and SEC regulation 17a, which mandate the safe archival of financial communication like emails and instant messages. There is also the California Database Breach Act (SB 1386) [4], which states that California residents must be notified if there is a reason to believe that the security of their personal information has been breached.

Some experts suggest that storage can be secured following the security models set forth in other domains of computing, such as application or network security. These domains rely on the use of strong authentication mechanisms, ensuring the right authorization systems are in place, replication for availability, integrity detection mechanisms, and the use of encryption for confidentiality. Unfortunately, none of these methods alone—especially encryption—is a comprehensive solution for protecting storage systems.

Storage protection has classical tradeoffs as detailed in [39]. For instance, the use of encryption may provide storage confidentiality but may also hamper performance, usability, and introduce denial-of-service vulnerabilities. The use of space replication may provide storage availability (with performance and cost tradeoffs) but may also increase storage exposure to confidentiality and integrity attacks. Time replication may provide storage protection (with performance and cost tradeoffs) but only if detected and restored within a backup or versioning window. When designing a storage protection solution, the security engineer cannot just utilize every protection technique (since their results are often in conflict) but must rather weigh the value of each security countermeasure versus the threats and vulnerabilities present in the specific environment.

Thus, it is important to understand all the threats and vulnerabilities present in a storage system before designing or implementing any storage protection solution because the threats determine the security countermeasures. To do otherwise invites disaster in that the chosen storage protec-

StorageSS'05, November 11, 2005, Fairfax, Virginia, USA.

tion solution may not match the threats and vulnerabilities in the actual system resulting in wasted investment, performance degradation, data compromise, service denial, or worse. This mismatch between threats and protection is often found in storage systems where cryptography is used when confidentiality or integrity is not a threat.

Despite this well-known challenge to match protection to threats, engineers typically design and implement protection solutions based on attacks enumerated by brainstorming or responding to exploits that have recently occurred. This approach is not systematic and is likely to leave large portions of the attack space unprotected [25]. The manifestation of the unsystematic protection approach can be found in retrofitted protection solutions patched into existing systems against threats unforeseen during design. To our knowledge this paper is the first work to address this fundamental issue of threat modeling in the storage domain.

Threat modeling is a proactive systematic engineering approach to identifying all possible threats and vulnerabilities in a complex system, regardless of the probability of occurrence. In Figure 1, we conceptualize the appropriate place for threat modeling as the basis upon which to build other security engineering processes. After threat modeling, threats may be analyzed using risk management techniques based on criticality and likelihood, and a decision made whether to mitigate or accept related risks. Security requirements specify what the system will do to mitigate the critical threats identified in risk management stage. The development of security countermeasures follows the general software engineering cycle of design, implementation, testing, and maintenance. Each stage feeds back to the preceding stage, and through that stage to all earlier stages. Feedback allows designers to catch mistakes made in earlier stages without allowing mistakes to escalate.

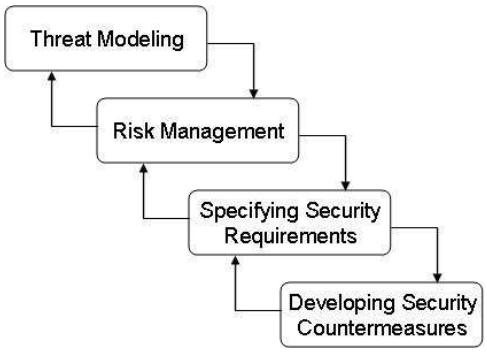

Figure 1: Threat Modeling as a Basis for Protection Engineering Processes

Threat modeling provides the foundation upon which the rest of the security system is built. Identifying threats supports developing realistic and meaningful security requirements. This is particularly important, for if the security requirements are faulty, the definition of security for that system is faulty, and thus the system cannot be secure.

As a step toward defining an integrated security solution, we propose two processes to creating a threat model for storage systems: (1) a threat model process based on the CIAA principles of confidentiality, integrity, availability, and authentication and (2) a threat model process based on the Data Lifecycle model. The remainder of this paper is organized as follows. In Section 2, we present the unique challenges of protecting storage systems. We then introduce two threat modeling processes for storage systems in Section 3. We briefly review related work in Section 4, and conclude with a summary in Section 5.

2. THE CHALLENGE OF PROTECTING STORAGE SYSTEMS

Protecting storage has its own set of unique challenges compared to other types of systems. For example, the most important asset of a storage system, data, needs to be properly labeled and protected, whether it is at rest within storage systems or in use. In most organizations, data volume continues to rise toward infinity despite enlightened views recommending the use of a finite retention policy. Any new storage protection techniques to be introduced must be backwards compatible to legacy systems and legacy data. For example, introducing a new cryptographic technique must not only encrypt or decrypt current data but re-encrypt or decrypt legacy data.

There are three types of storage today, with a fourth emerging [20]. Direct attached storage (DAS) is connected directly to a single system, much as the disk within a PC. Network attached storage (NAS) is accessed by way of the Ethernet LAN network, and accesses files. Storage area network (SAN) is accessed over a storage network, which today is typically Fibre Channel, providing what looks like disk drives to systems. Internet SCSI (iSCSI) offers storage networking over Ethernet LAN, but is not yet in widespread use. Object storage is an emerging technology combining aspects of SAN and NAS. The threat modeling we propose is independent of any of these storage types, some parts of the processes we propose may not be applicable to particular storage types.

Advances in storage technology has actually exacerbated the security problem with the shift to network storage. Suddenly, there are many point-to-point connections within the storage network, as many servers connect to storage arrays and NAS appliances supporting a number of different operating environments and applications. Storage also is increasingly distributed in architecture, meaning it is spread out across multiple data centers. As more customers use disaster recovery tools such as replication, remote disk and remote tape, the vulnerability of data-at-rest on remote sites will increase. Managed storage services also need to be evaluated for organizational security practices.

For storage over standard networking, including both NAS and emerging iSCSI block storage, security depends on both how well the network is protected and on security of the storage system itself. This is particularly true when storage is accessed over an organizational backbone network rather than through an isolated storage network or subnet.

Storage management interfaces also present security challenges for storage administrators. Many times, multiple storage arrays are managed as islands that need to be hand-touched to make configuration changes. This requires multiple passwords, access points, and ultimately new vulnerability points that need to be considered when evaluating storage security [17].

We summarize the unique challenges of storage protection to include:

- ever-growing data volume

- legacy systems storing legacy data
- innovative systems require backward interoperability
- more centralization on localized arrays
- more decentralization with accesses over networks
- increasing compliance requirements

3. THREAT MODELING PROCESSES FOR STORAGE

Creating a satisfactory threat model requires systematic and repeatable processes. It cannot be accomplished by simply brainstorming an attacker's possible intentions. An attacker only has to find one security flaw to compromise an entire storage system. Thus, it is important to be systematic during the threat modeling process to ensure that all known and unknown threats and vulnerabilities can be addressed.

Attacker Capabilities. Maybe the first question to ask, but a very difficult one to answer, is "who are the attackers?". While attackers can be categorized into broad categories such as individual hackers, hacker cells, insiders with privileged access, individual criminals, organized crime, espionage, terrorists, and nation-states – real attackers rarely fit neatly into one of these categories. Good data documenting attackers is rare, the best data so far documents only several of these attacker types via undercover operations monitoring Internet Relay Chat channels where attackers discuss their exploits. Understanding the attacker type is important to understand the resources and capabilities they have at their disposal.

Asset Goals. An attacker always has a specific goal in mind, targeting a particular asset. Assets are system resources which can be tangible (*e.g.*, data) or abstract (*e.g.*, data consistency) [38]. It is impossible to have a threat without a corresponding asset because assets are the threat goals. In a generic storage system, we identify the following incomplete list of assets that may be targeted by attackers:

Data blocks
Metadata
Log files
Buffer cache
File handles
Communication channel
Storage media
Device drivers
Data management software
Data availability
Data secrecy
Data integrity
Data consistency

Access Entry Points. Access entry points are what the attacker uses to gain access to the assets of the system. Examples of access points are open sockets, RPC interfaces, configuration files, hardware ports, and file system read/write. Now that storage networks can be directly connected to public networks, often a "back door" is created to an organization's information [12]. In a generic storage system, we identify the following incomplete list of access entry points that may be exploited by attackers:

- Access data from outside through network connection
- Access data from inside via trusted access or system compromise
- Physical access to SAN fabric
- Management interface from remote location to SAN fabric
- Compromised server accessing data and SAN fabric

Based on attacker capability, asset goals, and access entry points, we propose two processes to creating a threat model for storage systems. The first process evaluates storage threats and vulnerabilities organized in terms of classical security properties: confidentiality, integrity, availability, and authentication. We call this the CIAA process and discuss it in detail in Section 3.1. A second process involves identifying the value of the data being protected and mapping the data paths within the environment to ensure that they are fully protected *at-rest* and *in-flight*. We call this the Data Lifecycle process and discuss in Section 3.2. While components within each of these processes may or may not be applicable for a particular storage system, the overall processes are valid for any storage system.

3.1 The CIAA Threat Model Process

Security of computer-related systems must address four aspects: confidentiality, integrity, availability, and authentication (CIAA). We call this the *CIAA process* for threat modeling. In this section, we organize different types of storage system attacks into groups so they can be addressed by established CIAA protection techniques. We first present a description of the application of each CIAA aspect to storage systems followed by a listing of specific attack instances. While we attempt to be as comprehensive as possible, page length limitations do not allow us to list all known storage attacks and even if a complete list were possible, it would be outdated quickly as new attacks emerge. This process is dynamically extensible and it is left as an exercise to the reader to place attacks not listed into the appropriate CIAA group – new attacks will emerge but the CIAA aspects will remain constant. While the focus of this paper is on threat modeling, we do briefly provide references to the appropriate protection techniques for each aspect.

For completeness, we treat physical attacks as a separate group from CIAA. Physical attacks are outside the scope of computer security; they are best dealt with by organizational policies for physical security. We also treat them as a separate group since each physical attack instance typically violates one or more of CIAA aspects.

Confidentiality Attacks. Confidentiality attacks attempt to *read* information from a storage system without proper authorization. If an attacker gains administrator-level access to a system, they can typically explore storage with few safeguards. However, an attacker may also read information without illegitimate privilege escalation (e.g. insider attackers). Storage leaks via covert channels also fall into this group. For a survey of protection techniques against confidentiality attacks through the combined use of cryptography and access control see [35]. We identify the following confidentiality attacks on storage systems:

- *Sniffing Storage Traffic*: Storage traffic on dedicated storage networks or shared networks can be sniffed revealing data, metadata, and storage protocol signaling.

- *Snooping on Buffer Cache*: Most file systems utilize buffer caches to read and write storage blocks from and into the storage media. This is the norm regardless of the file system technology used. The buffer caches are allocated on demand. If an attacker can snoop into the buffer caches in memory she can access storage blocks and hence stored information she is not authorized to access.

- *Snooping on Deleted Storage Blocks*: In most file systems, storage blocks are allocated to files on demand. When a file is deleted, the storage block contents are not necessarily erased. Rather, most of the storage systems implement file deletion by erasing the file name and links from metadata and deleting the file i-node. Thus, data contents can be left un-erased in deleted and now free storage blocks. By accessing these storage blocks, it is possible for an attacker to gain access to sensitive data.

- *Snooping on Deallocated Memory*: Although most modern software deallocate data in memory after its last usage, it is possible for attackers to snoop on deallocated memory because the content of freed memory stays intact until it gets overwritten. Chow *et al.* point out in [9] that after deallocation, sensitive data such as passwords, social security numbers, and credit card numbers, often remain in memory indefinitely, possibly for days. This increases the risk of exposing sensitive data when a system is compromised, or of data being accidentally leaked due to unexpected feature interactions such as core dumps, logging, etc. One solution to this problem is to reduce data lifetime by zeroing at time of deallocation [9].

- *File System Profiling*: File system profiling [29, 27] attacks attempt to use access type, timestamps of last modification, file names, and other file system metadata to gain insight about storage system operation. For example, if a set of files are accessed in regular patterns, the attacker may infer the importance, function, and possibly even the content of these files.

Integrity Attacks. Integrity attacks attempt to *modify* information in a storage system without proper authorization. Modification may include creating, changing, appending, writing, and deleting both data and metadata. For a survey of integrity protection techniques focused on making storage immutable see [18]. For a survey of techniques focused on making any storage modifications clearly detectable so systems do not use illegitimately modified data or metadata see [19]. Here, we briefly discuss the following integrity attacks:

- *Storage Jamming*: *Storage Jamming* refers to the modification of data or metadata, done for the purpose of subversion, degradation, or disruption of operations [23]. An attacker may alter parts of legitimate data replacing it with semantically valid, but incorrect, data. Attackers can jam storage slowly over a long period of time or quickly over a short period of time or even coordinate changes with event triggers. There are numerous variants of this attack: in *barrage jamming* almost every original value is changed; in *spot jamming*, attackers only change a very small, but critical part of storage to cause disruption; in *repeat-back jamming* attackers replace original values with incorrect new values, wait until their purpose is served, and then replace the original values to avoid detection; in *freshness or roll-back jamming* attackers replace the current values with older values, [1] perhaps values more advantageous to attackers (e.g. a fired person becomes rehired with previous salary) [15]. Jamming is commonly used by attackers to hide attack traces by altering system and network log files.

- *Modifying Metadata*: Modifying metadata will disrupt a storage system. In any file system, if the i-node or file table are corrupted, the storage linked to the metadata cannot be accessed.

- *Subversion Attacks*: Attacks which modify operating system (OS) commands, kernel system calls, and/or storage system drivers to cause the wrong files, metadata or blocks to be modified or deleted.

Availability Attacks. Availability attacks attempt to make data or storage services *unavailable* for a period of time. Data (or metadata) and storage services must be available on-demand to legitimate parties when requested. Denial-of-service (DoS) attacks try to make data and storage services unavailable by exhausting resources through legitimate storage mechanisms which is what makes this attack the hardest to prevent. A storage system should have reasonable capacity (in terms of storage blocks for data and metadata) to meet the peak demands, however, this capacity is finite and can be exhausted. For surveys of availability protection techniques focused on replication in time using backups or versioning see [28] and [39] respectively. For a survey of techniques focused on backups in space (i.e., RAID) see [6].

The following list shows different types of availability attacks on storage systems:

- *Exhausting Log Space*: Storage systems use different types of logging. In log-structured file systems, the whole file system is a series of logs. An attacker can create a large number of small modifications to fill up the log space and lock up the system.

- *Exhausting Data Blocks*: An attacker can create a large number of files with random content to use up the available disk space.

- *Exhausting Metadata Space*: An attacker may create many empty/small/hidden files. While each file uses only a small amount of metadata space, a large number of metadata entries will degrade storage system performance.

- *Creating Redundant Versions*: Some versioning file systems, like S4 [37] and Elephant [31] create multiple versions of objects. Taking advantage of this, an attacker may launch a DoS attack by creating multiple versions of objects with minimal changes that will eventually exhaust storage space.

[1] With some cryptographic schemes this is possible if the keys are unknown and sometimes it is even possible to replace parts of a file with older versions.

- *Exhausting File Handles*: In most storage systems, file handles are used to access files, and these are locked until the file is closed. Also, file systems usually have a fixed number of file handles. An attacker may create a DoS by opening up multiple files but not closing them, thereby holding the file handle and degrading storage system performance.

- *Flash Memory Attacks*: Attacks on flash memory are designed to force inordinate numbers of erase cycles to exhaust that capability.

- *Attacks on Storage-Related OS Structures*: There are DoS attacks on the structure and management of buffer cache and other storage-related OS structures.

- *Fragmentation Attack*: If an attacker is able to place specific blocks in particular locations of a disk, they can create a file with associated blocks scattered across a disk platter. To access this particular file, the disk head has to move rapidly from one track to another in opposite directions degrading storage system performance and possibly causing damage. This attack may require system-level access to storage and can be difficult to detect.

- *Deletion of Data*: Deleting data or metadata is an extreme DoS attack but also one that is easily detectable and possibly recoverable given versioning or backups in time or space. If the deleted data is unrecoverable, the cost may range from insignificant to incalculable. Deleting system and network logs is commonly used by attackers to cover their attack traces.

- *Network Disruption*: Regardless of the underlying network technology, any software or congestion disruption to the network between the user and the storage system can degrade or disable storage.

Authentication Attacks. Authentication attacks occur when an attacker masquerades as a legitimate user identity (using a purloined password or credential) or an attack storage device masquerades as a legitimate storage device. For instance, a masquerader can launch insider attacks to access data/metadata (confidentiality), modify data/metadata (integrity), and/or deny others data/metadata (availability) based on the legitimate user identity authorization capabilities they have taken over. Man-in-the-middle attacks are another attack vector on authentication. We discuss the following two types of authentication attacks on storage systems:

- *Storage User Masquerading*: An attacker authenticates to a storage system as a legitimate user identity in order to access/modify/deny data or metadata. This may not necessarily involve the system OS since storage systems can have independent authentication and/or authorization controls.

- *Storage Device Masquerading*: An attack storage device authenticates as a legitimate storage device to the OS in order to access/modify/deny data or metadata.

Physical Attacks. We treat physical attacks as a separate group from CIAA since they are best dealt with by organizational policies for physical security (outside the scope of computer security). In CIAA attacks, it was assumed that the storage hardware is physically secure and the only way an attacker can launch attacks is through system vulnerabilities. In practical terms, physical attacks on storage hardware are common and may be the most likely and dangerous type of attack. We also treat them as a separate group since each physical attack instance typically violates one or more CIAA aspects.

The following list shows examples of physical attacks against storage systems:

- *Power Disruption*: If the power supply to a storage devices is disrupted, storage systems can become unavailable and data/metadata lost. Many storage systems have backup power sources for this reason, however, even in these cases long term power disruption is possible.

- *Network Disruption*: Regardless of the underlying network technology, any hardware component or cable disruption to the network between the user and the storage system can degrade or disable storage.

- *Storage Theft*: Thefts of storage media, storage devices, and computers containing storage systems occur. Recently, there has been an epidemic of thefts of unencrypted storage tapes containing confidential customer information. The decreasing size of portable storage combined with increasing capacity—*e.g.*, a USB memory stick with 4GB capacity—makes it easier to steal storage media. Although such thefts require low levels of sophistication on the attacker's part, they may result in large economic and security damages unless the stolen data/metadata is encrypted and replicated.

- *Data Recovery from Discarded Storage Media*: Neglecting to properly sanitize storage media before disposing of them allows attackers (or other third parties) access to data and metadata [14]. Proper sanitation techniques include:, *e.g.* overwriting, degaussing, and encryption (along with destruction of corresponding decryption key).

- *Physical Destruction of Storage Media*: Storage media can be physically destroyed by attackers using disintegration, incineration, pulverization, shredding, or melting. If storage media is intentionally destroyed by the owner with a purpose of retiring it, the data may still be recoverable. Hughes [21] demonstrates that even after shooting at a hard disk with a bullet, it is still possible to read data using special instruments such as a Magnetic Force Microscope.

- *Hardware Trojan*: A USB driver can be exploited to load malicious software. In [3], Barrall *et al.* describe how a custom-built USB device can fool an operating system into believing the device is any form of USB peripheral. Attackers can load malicious software, such as a keystroke logger, onto a target system simply by physically plugging the device into a USB port, bypassing the built-in OS security. A file containing harvested passwords can be retrieved through the USB port after a few days or a week.

3.2 The Data Lifecycle Threat Model Process

As long as data lives in a computer system, it is susceptible to exposure. An alternate and equally valid storage threat model can be based on the *Data Lifecycle Model* by examining the types of threats that can occur at different stages of data state from creation to extinction. We organize storage attacks into six groups according to the storage Data Lifecycle as shown in Figure 2. It should be noted that many of the attacks we list in this section appeared previously in the CIAA threat modeling process and thus each attack will not be described again in as much detail. The Data Lifecycle threat model process is a complementary view from a different perspective so the representative attacks listed in this section are meant to contrast it from the CIAA threat model process. Physical attacks are not separated in the Data Lifecycle threat modeling process since each stage does not distinguish whether the data is in electronic or physical form.

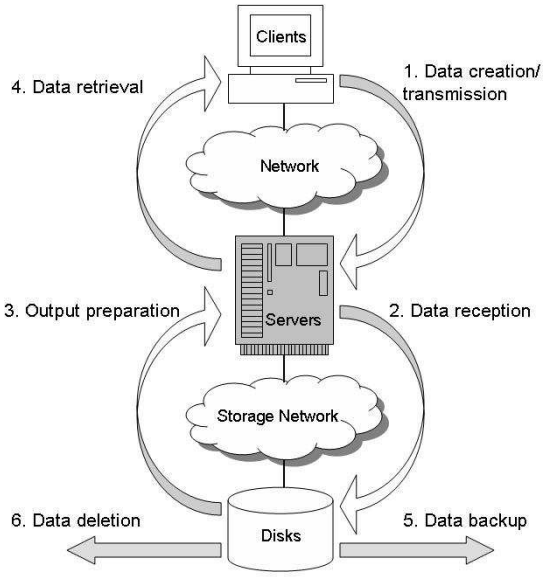

Figure 2: Storage Data Lifecycle.

Figure 3 expands the storage data lifecycle to show attacks with their target assets. Many organizations follow *Information Lifecycle Management* (ILM) practice to manage data from the moment it is created to the time it is no longer needed. One implementation of ILM is a hierarchical storage management where newer data, and data that must be accessed more frequently, are stored on faster, more expensive storage media, while less critical data is stored on cheaper, slower media. In [8], Chow *et al.* analyze sensitive data handling in several applications such as Mozilla, Apache, and Perl revealing that these applications take virtually no measures to limit the lifetime of sensitive data – leaving passwords and other sensitive data scattered throughout user and kernel memory.

Next, we briefly discuss the different stages of the Data Lifecycle Model:

1. Data Creation/Transmission. Newly created data is transferred from clients to the storage system. This may happen either through shared networks or dedicated system buses. During this stage, the following attacks are possible:

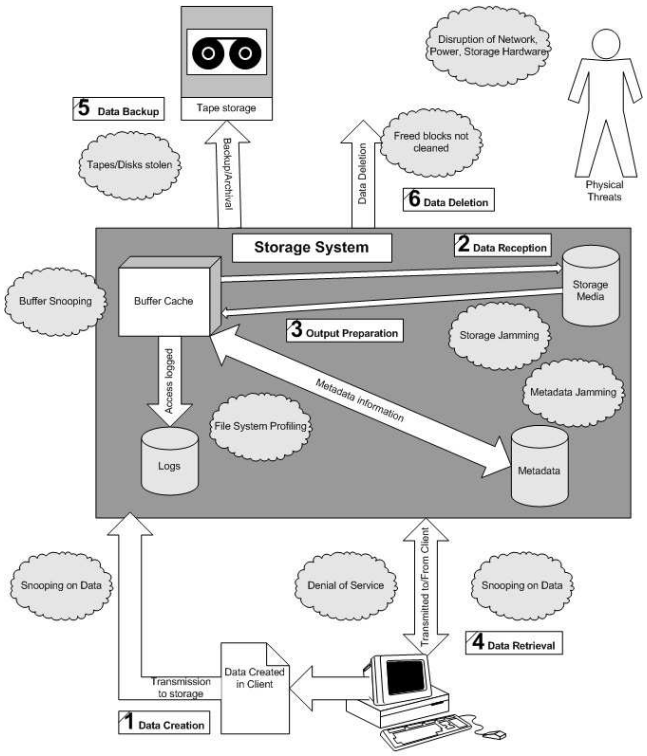

Figure 3: Storage Attacks Based on Data Lifecycle.

- *Confidentiality*: Attackers may sniff data on the communication channel.
- *Integrity*: Attackers may modify data by performing a man-in-the-middle attack.
- *Availability*: Attackers may disrupt the communication channel with DoS attacks.
- *Authentication*: Attackers may create real or fake data under a stolen identity.

2. Data Reception. Data arrives at the storage system, buffer caches are allocated, data is written to storage media, and activity logs are created. During this stage, the following attacks are possible:

- *Confidentiality*: Attackers may sniff data on buffer caches.
- *Integrity*: Attackers may change contents of buffer caches.
- *Availability*: Attackers may exhaust available file handles blocking the creation of new files.
- *Authentication*: An attacker storage device may masquerade as a legitimate storage device to receive data.

3. Output Preparation. Storage is accessed by servers and data is written into buffer caches in preparation for client retrieval. During this stage, the following attacks are possible:

- *Confidentiality*: Attackers may perform file system profiling to detect usage patterns.
- *Integrity*: Attackers may jam storage in one of many variants.
- *Availability*: Attackers may exhaust data blocks or metadata to prevent output access.
- *Authentication*: Attackers may masquerade as a stolen identity to gain access to output data.

4. Data Retrieval. In this stage, data is retrieved from storage servers by clients. The following attacks are possible during this stage:

- *Confidentiality*: Data can be sniffed during transmission from storage system to clients. Storage media or hardware can be stolen.
- *Integrity*: By launching a man-in-the-middle attack, storage contents can be modified in transit. Storage media can be vandalized.
- *Availability*: By attacking the power supply, storage media, or network, availability can be denied.
- *Authentication*: Attackers may masquerade as a stolen identity to retrieve selected data.

5. Data Backup. Data is replicated in time to tape or disk for archival purposes. During this stage, the following attacks may occur:

- *Confidentiality*: Backup media may be stolen.
- *Integrity*: As some operating systems have become harder to compromise, attackers have focused increasingly on relatively less-protected storage application (backup) software. For example, a buffer-overflow vulnerability in backup applications has allowed attackers to take control of systems, execute malicious code and launch DoS attacks [41].
- *Availability*: By attacking backup timing synchronization, power supply, storage media, or network, backup availability can be denied.
- *Authentication*: Attacker devices may masquerade as trusted storage system components to receive replicated data.

6. Data Deletion. When storage systems are upgraded, retired due to proactive maintenance, or reach the end of their lifetime, data is evaluated for deletion and discarding. The following attacks can be executed during this stage:

- *Confidentiality*: An attacker may snoop on deleted storage blocks. Regulations such as Sarbanes-Oxley, and HIPAA require proof of secure erasure.
- *Integrity*: Metadata can be modified to subvert accurate evaluation for deletion and discarding.
- *Availability*: Data can be spread intermingled with other data within storage systems such that its deletion is difficult, if not impossible.
- *Authentication*: Attackers may masquerade under a legitimate identity to delete data before the end of its lifetime or to extend data beyond the end of its lifetime.

3.3 Threats and Standards

When deciding which security countermeasures should be provided for a storage system based on its threat model, engineers should consult storage security standards since they may provide useful policies that directly address CIAA or Data Lifecycle threats. These standards improve the quality and interoperability of various storage systems, however, security standards only exist for a subset of storage system implementations and merely adhering to them does not guarantee that a system is protected.

For example, consider the Fibre Channel Security Protocol (FC-SP), and the Storage Network Industry Association's Storage Management Initiative Specification (SMI-S) storage security standards. Both standards facilitate storage protection by building consensus between vendors to allow interoperability and broader heterogeneous approaches. The FC-SP includes protocols to authenticate and establish secrets for Fibre Channel entities, protocols for frame-by-frame integrity and confidentiality, and protocols to define and distribute security policies within the fabric. The SMI specification provides a common approach to managing devices in a storage network, using the common information model (CIM) as a foundation and SSL for secure management.

A counter argument on the use of standards for storage security is that a widely implemented standard must be solid since any vulnerability in a standard implementation would be very attractive to attackers to use as part of a class-break exploit (standard becomes a threat). For this reason some protectors advocate non-standard storage implementations based on security-by-obscurity [7].

4. RELATED WORK

While this paper is the first comprehensive treatment of different storage attacks, specific attacks on storage systems have been discussed by many researchers. We have already referenced many different storage attack instances to illustrate specific aspects of the two threat modeling processes. In this section we highlight significant related work we have not yet referenced as well as attempting to put our contribution in context. McDermott [23] was the first to identify a unique storage attack when he defined storage jamming and variants of this attack in 1996. Different types of defense mechanisms against storage jamming are further discussed by McDermott *et al.* [22]. Ammann *et al.* [1] explore information warfare attacks in which they discuss attacks on storage confidentiality, integrity and availability. McDermott extends his previous work to discuss the possible attacks and corresponding replication based defenses in [24]. Systems like self-securing storage [37], and the PASIS architecture [42], developed at Carnegie Mellon University, focus on insider threats. Pennington *et al.* [27], also from Carnegie Mellon University, explore threat models for networked storage with intrusion detection systems specifically designed for storage systems. Dagon *et al.* [11] define the secure storage problem and present techniques for enhancing security of storage. Nguyen *et al.* [26] explore methods of file system monitoring via interception of system calls, in order to detect insider threats. More detailed research on file profiling techniques is presented by Reiher in [29].

Threat modeling provides a method of assessing and formalizing the security risks associated with a system. Schneier

[33, 34] first introduced the attack-tree method for threat modeling. Steffan and Schumacher [36] explore collaborative attack modeling based on shared knowledge about vulnerabilities. Roscoe *et al.* [30] discuss threat models for ubiquitous systems. Threat models for web applications are analyzed by Cock *et al.* [10]. The only book on the subject is one by Swiderski and Snyder which focuses on software engineering [38]. Threat models for computer networked systems are discussed from an economic perspective by Schechter and Smith [32]. The most recent work in threat modeling is a comprehensive Internet browser threat model by Griggs [16] and a threat model for attacks against the Domain Name System by Atkins and Austein [2].

5. SUMMARY

Storage systems present unique security challenges. In this paper, we present two systematic threat modeling processes upon which to base protection for storage systems: (1) the CIAA process and (2) the Data Lifecycle Model process. To the best of our knowledge, this is the first paper to present systematic processes toward threat modeling for storage systems. The first CIAA process organizes threats and vulnerabilities into classes of attacks to match existing protection techniques for Confidentiality, Integrity, Availability, and Authentication. The second Data Lifecycle process focuses on the most important asset of a storage system—data—and traces the data lifecycle within an environment to ensure it is fully protected at each stage. Within each process, we illustrate each class of threats and vulnerabilities being examined with specific attack instances. The purpose is not to enumerate every possible attack instance but rather to show how all attacks can be classified and thus addressed by protection techniques facilitated by either of the threat modeling processes we have introduced. Of course, zero-day storage attacks will also occur and they also can be classified using either threat process. We invite feedback on this work, especially attack instances to illustrate different aspects of the threat model processes we have presented. It is our hope that this initial work will start a discussion on how to better design and implement storage protection solutions.

Acknowledgments

The authors would first like to thank their colleagues in the StorageSS Research Group at NCSA who contributed directly and indirectly to this work. We next acknowledge insights on specific storage threats from (in alphabetical order): Roy Campbell (UIUC), Ben Pfaff (Stanford University), Klara Nahrstedt (UIUC), Marianne Winslett (UIUC), and Yuanyuan Zhou (UIUC). Last, we acknowledge the constructive criticism from the anonymous StorageSS reviewers which we have incorporated to improve this paper.

6. REFERENCES

[1] P. Ammann, S. Jajodia, C. D. McCollum, and B. T. Blaustein. Surviving Information Warfare Attacks on Databases. In *Proc. of the IEEE Symposium on Security and Privacy*, 1997.

[2] D. Atkins and R. Austein. Threat Analysis of the Domain Name System (DNS). *RFC 3833*, August 2004.

[3] D. Barrall and D. Dewey. Plug and Root, the USB Key to the Kingdom. *Presentation at Black Hat Briefings*, 2005.

[4] California Senate. California Database Breach Act (SB 1386). http://info.sen.ca.gov/pub/01-02/bill/sen/sb_1351-1400/sb_1386_bill_20020926_chaptered.html, 2002.

[5] Centers for Medicare & Medicaid Services. The Health Insurance Portability and Accountability Act of 1996 (HIPAA). http://www.cms.hhs.gov/hipaa/, 1996.

[6] P. M. Chen, E. K. Lee, G. A. Gibson, R. H. Katz, and D. A. Patterson. RAID: High-Performance, Reliable Secondary Storage. In *ACM Computing Surveys 26(2)*, pages 145–185, 1994.

[7] J. Chirillo and S. Blaul. *Storage Security: Protecting, SANs, NAS and DAS*. Wiley, 2002.

[8] J. Chow, B. Pfaff, T. Garfinkel, K. Christopher, and M. Rosenblum. Understanding Data Lifetime via Whole System Simulation. In *Proc. of 13th Usenix Security Symposium*, 2004.

[9] J. Chow, B. Pfaff, T. Garfinkel, and M. Rosenblum. Shredding Your Garbage: Reducing Data Lifetime Through Secure Deallocation. In *Proc. of 14th Usenix Security Symposium*, 2005.

[10] D. D. Cock, K. Wouters, D. Schellekens, D. Singele, and B. Preneel. Threat Modelling for Security Tokens in Web Applications. In *Proc. of the IFIP TC6/TC11 International Conference on Communications and Multimedia Security (CMS)*, pages 183–193, 2004.

[11] D. Dagon, W. Lee, and R. Lipton. Protecting Secret Data from Insider Attacks. In *Proc. of Ninth International Conference on Financial Cryptography and Data Security*, 2005.

[12] A. Edmonds. Towards Securing Information End-to-End: Networked Storage Security Update and Best Practices. *White Paper*, February 2003.

[13] Federal Trade Commission. Gramm-Leach-Bliley Act of 1999.

[14] S. Garfinkel and A. Shelat. Remembrance of Data Passed: A Study of Disk Sanitization Practices. *IEEE Security & Privacy*, pages 17–27, January/February 2003.

[15] E. Goh, H. Shacham, N. Modadugu, and D. Boneh. SiRiUS: Securing Remote Untrusted Storage. In *10th Annual Network and Distributed System Security Symposium (NDSS)*, 2003.

[16] I. Griggs. Browser Threat Model. http://iang.org/ssl/browser_threat_model.html, 2004.

[17] J. Gruener and M. Kovar. The Emerging Storage Security Challenge. *Yankee Group Report*, September 2003.

[18] R. Hasan, J. Tucek, P. Stanton, W. Yurcik, L. Brumbaugh, J. Rosendale, and R. Boonstra. The Techniques and Challenges of Immutable Storage for Applications in Multimedia. In *IS&T/SPIE International Symposium Electronic Imaging / Storage and Retrieval Methods and Applications for Multimedia (EI121)*, 2005.

[19] E. Haubert, J. Tucek, L. Brumbaugh, and W. Yurcik. Tamper-Resistant Storage Techniques for Multimedia

Systems. In *IS&T/SPIE International Symposium Electronic Imaging / Storage and Retrieval Methods and Applications for Multimedia (EI121)*, 2005.

[20] HP. Understanding Storage Security. *RFC 3833*, February 2005.

[21] J. Hughes. Encrypted Storage–Challenges and Methods. In *Tutorial, IEEE/NASA Goddard Conference on Mass Storage Systems & Technologies (MSST)*, 2005.

[22] J. McDermott, R. Gelinas, and S. Ornstein. Doc, Wyatt, and Virgil: Prototyping Storage Jamming Defenses. In *13th Annual Computer Security Applications Conference (ACSAC)*, 1997.

[23] J. McDermott and D. Goldschlag. Storage Jamming. In *Proc. of the Ninth Annual IFIP TC11 WG11.3 Working Conference on Database Security IX : Status and Prospects*, pages 365–381, 1996.

[24] J. P. McDermott. Replication Does Survive Information Warfare Attacks. In *IFIP Workshop on Database Security*, pages 219–228, 1997.

[25] S. Myagmar, A. J. Lee, and W. Yurcik. Threat Modeling as a Basis for Security Requirements (SREIS). In *Symposium on Requirements Engineering for Information Security*, 2005.

[26] N. Nguyen, P. Reiher, and G. Kuenning. Detecting Insider Threats by Monitoring System Call Activity. In *Proc. of IEEE Workshop on Information Assurance*, 2001.

[27] A. Pennington, J. Strunk, J. Griffin, C. Soules, G. Goodson, and G. Ganger. Storage-Based Intrusion Detection: Watching Storage Activity for Suspicious Behavior. In *Proc. of Usenix Security Symposium*, 2003.

[28] G. A. Pluta, L. Brumbaugh, W. Yurcik, and J. Tucek. Who Moved My Data? A Backup Tracking System for Dynamic Workstation Environments. In *18th Usenix Large Installation System Administration Conference (LISA)*, 2004.

[29] P. Reiher. File Profiling for Insider Threats. *Technical Report*, February 2002.

[30] A. Roscoe, M. Goldsmith, S. Creese, and I. Zakiuddin. The Attacker in Ubiquitous Computing Environments: Formalising the Threat Model. In *Proc. of First International Workshop on Formal Aspects in Security and Trust*, 2003.

[31] D. S. Santry, M. J. Feeley, N. C. Hutchinson, A. C. Veitch, R. W. Carton, and J. Ofir. Deciding When to Forget in the Elephant File System. In *Proc. of the Seventeenth ACM Symposium on Operating Systems Principles (SOSP)*, pages 110–123, 1999.

[32] S. Schechter and M. D. Smith. How Much Security Is Enough to Stop a Thief?: The Economics of Outsider Theft via Computer Systems and Networks. In *Financial Cryptography*, pages 122–137, 2003.

[33] B. Schneier. Attack Trees: Modeling Security Threats. *Dr. Dobb's Journal*, December 1999.

[34] B. Schneier. *Secrets and Lies: Digital Security in a Networked World*. John Wiley and Sons, 2000.

[35] P. Stanton, W. Yurcik, and L. Brumbaugh. Protecting Multimedia Data in Storage: A Survey of Techniques Emphasizing Encryption. In *IS&T/SPIE International Symposium Electronic Imaging / Storage and Retrieval Methods and Applications for Multimedia (EI121)*, 2005.

[36] J. Steffan and M. Schumacher. Collaborative Attack Modeling. In *Proc. of the 2002 ACM symposium on Applied computing (SAC)*, pages 253–259, 2002.

[37] J. D. Strunk, G. R. Goodson, M. L. Scheinholtz, C. A. Soules, and G. R. Ganger. Self-Securing Storage: Protecting Data in Compromised Systems. In *Proc. of the 4th Symposium on Operating Design and Implementation (OSDI)*, 2000.

[38] F. Swiderski and W. Snyder. *Threat Modeling*. Microsoft Press, 2004.

[39] J. Tucek, P. Stanton, E. Haubert, R. Hasan, L. Brumbaugh, and W. Yurcik. Trade-offs in Protecting Storage: A Meta-Data Comparison of Cryptographic, Backup/Versioning, Immutable/Tamper-Proof, and Redundant Storage Solutions. In *2nd IEEE - 13th NASA Goddard Conference on Mass Storage Systems and Technologies (MSST)*, 2005.

[40] U.S. Securities and Exchange Commission. Sarbanes-Oxley Act of 2002. http://www.sarbanes-oxley-forum.com/.

[41] J. Vijayan. CA Security Hole Points to Data Backup Threats. *Computerworld*, August 2005.

[42] J. J. Wylie, M. W. Bigrigg, J. D. Strunk, G. R. Ganger, H. Kilite, and P. K. Khosla. Survivable Information Storage Systems. *IEEE Computer*, 33(8):61–68, 2000.