ISSN:0975-9646

A.Rajesh et al, / (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 2 (4) , 2011, 1386-1389

The Effectiveness Of The Locationguard Scheme Against DoS Attacks

A.Rajesh
(CS)
Vivekananda Institute Of Technology & Science
Karimnagar, India

P.Pradeep kumar
CSE-Dept
Vivekananda Institute Of Technology & Science
Karimnagar, India

T.Bhaskar
IT-Dept
Sree Chaitanya College Of Engineering
Karimnagar, India

Abstract- **Server less distributed computing has received significant attention from both the business and the investigate society. Among the most popular applications are the wide-area network file systems, exemplified by CFS, Farsite, and Ocean Store. These file systems store files on a large collection of untrusted nodes that form an overlay network. They use cryptographic techniques to maintain file confidentiality and integrity from malicious nodes. Unfortunately, cryptographic techniques cannot protect a file holder from a denial-of-service (DoS) attack or a host compromise attack. Hence, most of these distributed file systems are vulnerable to targeted file attacks, wherein an adversary attempts to attack a small (chosen) set of files by attacking the nodes that host them. This paper presents Location Guard—a location hiding technique for securing overlay file storage systems from targeted file attacks. Location Guard has three essential components: 1) location key, consisting of a random bit string (e.g., 128 bits) that serves as the key to the location of a file, 2) routing guard, a secure algorithm that protects accesses to a file in the overlay network given its location key such that neither its key nor its location is revealed to an adversary, and 3) a set of location inference guards, which refer to an extensible component of the Location Guard. Our experimental results quantify the overhead of employing Location Guard and demonstrate its effectiveness against DoS attacks, host compromise attacks, and various location inference attacks.**

Keywords- File systems, overlay networks, denial-of-service attacks, performance and scalability, location hiding.

I.INTRODUCTTION

SEVERAL serverless file storage services, like CFS [2], Farsite [1], OceanStore [5], and SiRiUS [4], have recently emerged. In contrast to traditional file systems, they harness the resources available at desktop workstations that are distributed over a wide-area network. The collective resources available at these desktop workstations amount to several petaflops of computing power and several hundred petabytes of storage space [1]. These emerging trends have motivated serverless file storage as one of the most popular applications over decentralized overlay networks. An overlay network is a virtual network formed by nodes (desktop workstations) on top of an existing TCP/IP-network. Overlay networks typically support a lookup protocol. A lookup operation identifies the location of a file given its filename. Location of a file denotes the IP-address of the node that currently hosts the file. There are four important issues that need to be addressed to enable wide deployment of serverless file systems for mission critical applications. Efficiency of the lookup protocol. There are two kinds of lookup protocol that have been commonly deployed: the Gnutella-like broadcast-based lookup protocols [3] and the distributed hash table (DHT)-based lookup protocols [8], [6], [7]. File systems like CFS, Farsite, and OceanStore use DHT-based lookup protocols because of their ability to locate any file in a small and bounded number of hops. Malicious and unreliable nodes. Serverless file storage services are faced with the challenge of having to harness the collective resources of loosely coupled, insecure, and unreliable machines to

provide a secure and reliable file storageservice. To complicate matters further, some of the nodes in the overlay network could be malicious. CFS employs cryptographic techniques to maintain file data confidentiality and integrity. Farsite permits file write and update operations by using a Byzantine fault-tolerant (BFT) group of metadata servers (directory service). Both CFS and Farsite use replication as a technique to provide higher fault tolerance and availability.

Targeted file attacks. A major drawback with serverless file systems is that they are vulnerable to targeted attacks on files. In a targeted attack, an adversary is interested in compromising a small set of target files through a denial-ofservice (DoS) attack or a host compromise attack. A DoS attack would render the target file unavailable; a host compromise attack could corrupt all the replicas of a file thereby effectively wiping out the target file from the file system. The fundamental problem with these systems is that: 1) the number of replicas (R) maintained by the system is usually much smaller than the number of malicious nodes (B) and 2) the replicas of a file are stored at publicly known locations, that is, given the file name f, an adversary (including users who may not have access to file f) can determine the IP-addresses of nodes that host f's replicas.Hence, malicious nodes can easily launch DoS or host compromise attacks on the set of R replica holders of a target file (R<< B).

Efficient access control. A read-only file system like CFS can exercise access control by simply encrypting the contents of each file, and distributing the keys only to the legal users of that file. Farsite, a read/write file system, exercises access control using access control lists (ACL) that are maintained using a BFT protocol. However, access control is not truly distributed in Farsite because all users are authenticated by a small collection of directory group servers. Further, public-key infrastructure (PKI)-based authentication and Byzantine fault tolerance-based authorization are known to be more expensive than a simple and fast capability-based access control mechanism [5].

Serverless file system like CFS, Farsite, and OceanStore are layered on top of DHT-based protocols. These file systems typically provide the following properties:

1. A file lookup is guaranteed to succeed if and only if the file is present in the system.
2. A file lookup terminates in a small and bounded number of hops.
3. The files are uniformly distributed among all active nodes.
4. The system handles dynamic node joins and leaves.

In the rest of this paper, we assume that Chord [29] is used as the overlay network's lookup protocol. However, the results presented in this paper are applicable to most DHT-based lookup protocols.

II. LOCATION GUARD

A. Concepts and Definitions

In this section, we define the concept of location keys and its location hiding properties. We discuss the concrete design of location key implementation and how location keys and location guards protect a file system from targeted file attacks in the subsequent sections. Consider an overlay network of size N with a Chord-like lookup protocol _. Let f1; f2; . . . ; fR denote the R replicas of a file f. Location of a replica fi refers to the IP-address of the node (replica holder) that stores replica fi. A file lookup algorithm is defined as a function that accepts fi and outputs its location on the overlay network. Formally, we have _ : fi ! loc maps a replica fi to its location loc on the overlay network P.

Definition 1. location key. A location key lk of a file f is a relatively small amount (m-bit binary string, typically m ¼ 128) of information that is used by a lookup algorithm $\Psi : (f, lk) \longrightarrow loc$ customize the transformation of a file into its location such that the following three properties are satisfied:

1. Given the location key of a file f, it is easy to locate the R replicas of file f.
2. Without knowing the location key of a file f, it is hard for an adversary to locate any of its replicas.
3. The location key lk of a file f should not be exposed to an adversary when it is used to access the file f.

Informally, location keys are keys with location hiding property. Each file in the system is associated with a location key that is kept secret by the users of that file. A location key for the file f determines the locations of its replicas in the overlay network. Note that the lookup algorithm _ is publicly known; only a file's location key is kept secret.

Property 1 ensures that valid users of a file f can easily access it provided they know its location key *lk*.

Property 2 guarantees that illegal users who do not have the correct location key will not be able to locate the file on the overlay network, making it harder for an adversary to launch a targeted file attack. Property 3 warrants that no information about the location key *lk* of a file f is revealed to an adversary when executing the lookup algorithm Ψ.

Having defined the concept of location key, we present a reference model for a file system that operates on LocationGuard. We use this reference model to present a concrete design of LocationGuard's three core components: the location key, the routing guard, and the location inference guards.

III. LOCATION INFERENCE GUARDS

Location inference attacks refer to those attacks where in an adversary attempts to infer the location of a file using indirect techniques that exploit file metadata information such as file access frequency, file size, and so forth. LocationGuard includes a suite of four fundamental and inexpensive inference guards: lookup frequency inference guard, user IP-address inference guard, file replica inference guard, and file size inference guard. LocationGuard also includes a capability revocation-based location rekeying

mechanism as a general guard against any inference attack. In this section, we present the four fundamental inference guards and the location rekeying technique in detail.

A.Passive Inference Guards

Passive inference attacks refer to those attacks wherein an adversary attempts to infer the location of a target file by passively observing the overlay network. We present two inference guards: lookup frequency inference guard and user IP-address inference guard to guard the file system against two common passive inference attacks. The lookup frequency inference attack is based on the ability of malicious nodes to observe the frequency of lookup queries on the overlay network. Assuming that the adversary knows the relative file popularity, it can use the target file's lookup frequency to infer its location. The user IP-address inference attack is based on assumption that the identity of the user can be inferred from its IP-address by an overlay network node r, when the user requests node r to perform a lookup on its behalf. The malicious node r could log and report this information to the adversary.

IV.HOST COMPROMISE-BASED INFERENCE GUARDS

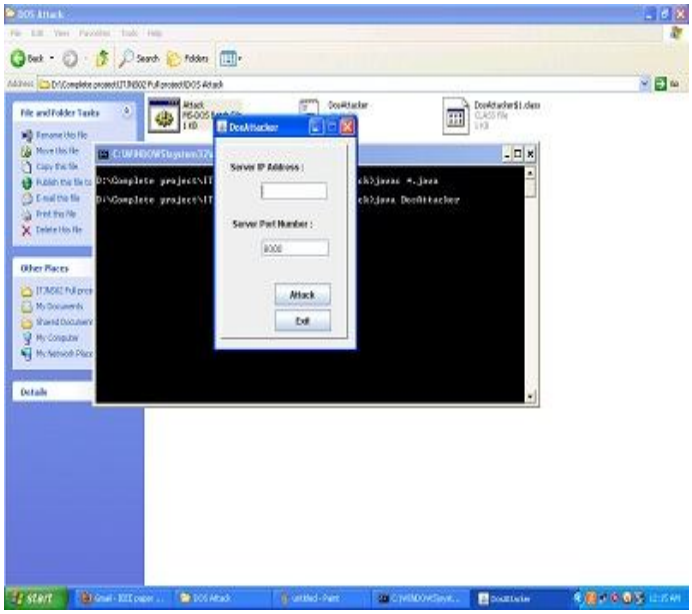
Host compromise-based inference attacks require the adversary to perform an active host compromise attack before it can infer the location of a target file. We present two inference guards: file replica inference guard and file size inference guard to guard the file system against two common host compromise-based inference attacks. The file replica inference attack attempts to infer the identity of a file from its contents. Note that an adversary can reach the contents of a file only after it compromises the replica holder (unless the replica holder is malicious). The file size inference attack attempts to infer the identity of a file from its size. If the sizes of files stored on the overlay network are sufficiently skewed, the file size could by itself be sufficient to identify a target file.

V. LOCATION REKEYING

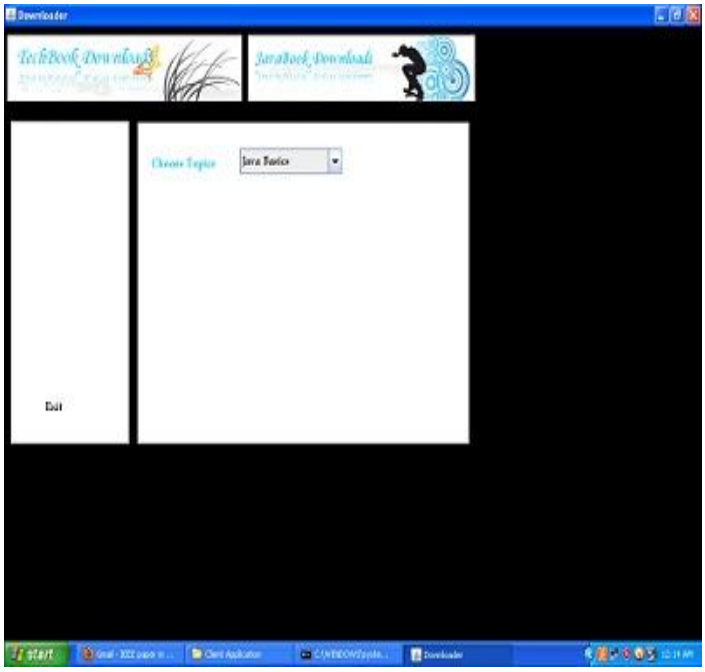
In addition to the inference attacks listed above, there could be other possible inference attacks on a LocationGuardbased file system. In due course of time, the adversary might be able to gather enough information to infer the location of a target file. Location rekeying is a general defense against both known and unknown inference attacks. Users can periodically choose new location keys so as to render all past inferences made by an adversary useless. This is analogous to periodic rekeying of cryptographic keys. Unfortunately, rekeying is an expensive operation: rekeying cryptographic keys requires data to be reencrypted; rekeying location keys requires files to be relocated on the overlay network. Hence, it is important to keep the rekeying frequency small enough to reduce performance overheads and large enough to secure files on the overlay network. In our experiment section, we estimate the periodicity with which location keys have to be changed in order to reduce the probability of an attack on a target file.

VI.EXPERIMENTAL EVALUATION

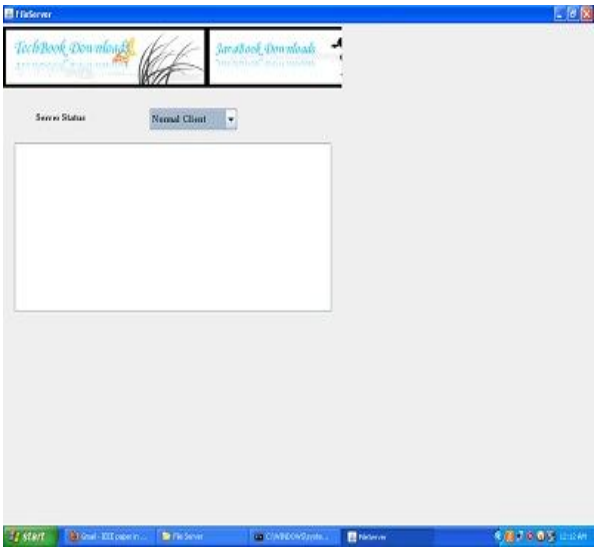
In this section, we briefly sketch our implementation of LocationGuard and quantify the overhead added by LocationGuard to the file system.



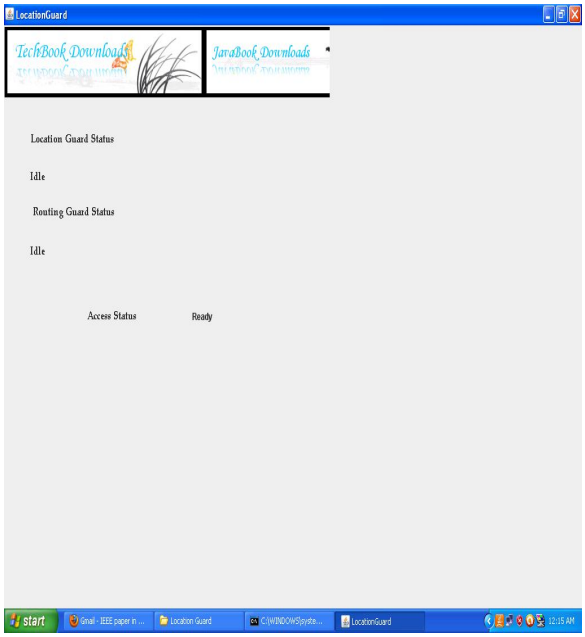
Dos attacker



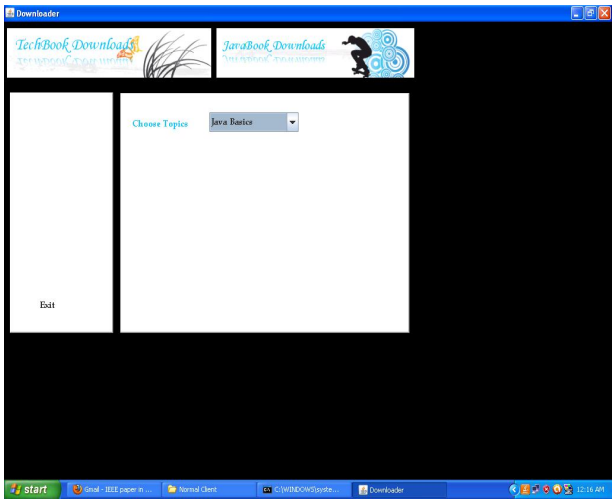
Downloader



File server



Location guard



Normal client

VII. CONCLUSION

We have described LocationGuard—a technique for securing wide-area serverless file sharing systems from targeted file attacks. Analogous to traditional cryptographic keys that hide the contents of a file, LocationGuard hides the location of a file on an overlay network. LocationGuard protects a target file from DoS attacks, host compromise attacks, and file location inference attacks by providing a simple and efficient access control mechanism with minimal performance and storage overhead. The unique characteristics of LocationGuard approach is the careful combination of location key, routing guard, and an extensible package of location inference guards, which makes it very hard for an adversary to infer the location of a target file by either actively or passively observing the overlay network. Our experimental results quantify the overhead of employing location guards and demonstrate the effectiveness of the LocationGuard scheme against DoS attacks, host compromise attacks, and various location inference attacks.

REFFERENCES

1.A. Adya, W. Bolosky, M. Castro, G. Cermak, R. Chaiken, J.R. Douceur, J. Howell, J.R. Lorch, M. Theimer, and R.P. Wattenhofer, “Farsite: Federated, Available and Reliable Storage for an Incompletely Trusted Environment,” Proc. Fifth Symp. Operating Systems Design and Implementation (OSDI), 2002.

2. F. Dabek, M.F. Kaashoek, D. Karger, R. Morris, and I. Stoica, “Wide-Area Cooperative Storage with CFS,” Proc. 18th ACM Symp. Operating System Principles (SOSP ’01), Oct. 2001.

3. The Gnutella Home Page, Gnutella, http://gnutella.wego.com/, 2008.

4. E.J. Goh, H. Shacham, N. Modadugu, and D. Boneh, “SiRiUS: Securing Remote Untrusted Storage,” Proc. 10th Ann. Network and Distributed System Security Symp. (NDSS), 2003.

5 J. Kubiatowics, D. Bindel, Y. Chen, S. Czerwinski, P. Eaton, D. Geels, R. Gummadi, S. Rhea, H. Weatherspoon, W. Weimer, C. Wells, and B. Zhao, “OceanStore: An Architecture for Global-Scale Persistent Storage,” Proc. Ninth Int’l Conf. Architectural Support for Programming Languages and Operating Systems (ASPLOS ’00), Nov. 2000.

6. S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, “A Scalable Content-Addressable Network,” Proc. ACM SIGCOMM ’01, Aug. 2001.

7. A. Rowstron and P. Druschel, “Pastry: Scalable, Distributed Object Location and Routing for Large-Scale Peer-to-Peer Systems,” Proc. 18th IFIP/ACM Int’l Conf. Distributed Systems Platforms (Middleware ’01), Nov. 2001.

8 I.Stoica, R. Morris, D. Karger, M. Kaashoek, and H. Balakrishnan, “Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications,” Proc. ACM SIGCOMM ’01, Aug. 2001.