ISSN No. 0976-5697

Volume 2, No. 4, July-August 2011

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

A LocationGuard Approach: An Efficacious Scheme to Alleviate DoS Attacks

B. Raj Kumar
Assoc. Professor in CSE Dept
S.R. Engineering College
Ananthasagar, Warangal, India
raj_kumar_b@srecwarangal.ac.in

G. Aruna Kranthi
Sr. Asst. Professor in CSE Dept
S.R. Engineering College
Ananthasagar, Warangal, India
raj.kranthi@gmail.com

K. Srikanth Reddy*
Computer Science and Engineering
S.R. Engineering College
Ananthasagar, Warangal, India
konni.srikanthreddy@gmail.com

Abstract: CFS, Farsite, and OceanStore 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 a n 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 ha s three essential components: 1) location key , consisting of a random bit string (eg .,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 LocationGuard. Our experimental results quantify the overhead of employing LocationGuard and demonstrate its effectiveness against DoS attacks, host compromise attacks, and various location inference attacks.

Keywords: *Denial-of-service attacks; overlay networks; file system; location hiding; performance and scalability.*

I. INTRODUCTION

The resources are available at desktop workstations that are distributed over a wide-area network by the ability of several serverless file storage services. 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 to enable wide deployment of serverless file systems for mission critical applications.

A. *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 [4]. File systems like CFS [2], Farsite [1], and OceanStore [6] use DHT-based lookup protocols because of their ability to locate any file in a small and bounded number of hops.

B. *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-storage service. 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.

C. *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-of- service (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 [8], 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).

D. *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.

Bearing these issues in mind, we present LocationGuard as an effective technique for countering targeted file attacks. The fundamental idea behind LocationGuard is to hide the very location of a file and its replicas such that a legal user who possesses a file's location key can easily and securely locate the file on the overlay network; but without knowing the file's location key, an adversary would not be able to even locate the file, let alone access it or attempt to attack it.

Location Guard implements an efficient capability-based file access control mechanism through three essential components. The first component of LocationGuard is a location key, which is a random bit string (128 bits) used as a key to the location of a file in the overlay network, and addresses the capability revocation problem by periodic or conditional rekeying mechanisms. A file's location key is used to generate legal capabilities (tokens) that can be used to access its replicas. The second component is the routing guard, a secure algorithm to locate a file in the overlay network given its location key such that neither the key nor the location is revealed to an adversary. The third component is an the file is present in the system. extensible collection of location inference guards, which protect the system from traffic analysis-based inference attacks, such as lookup frequency inference attacks, user IP-address inference attacks, file replica inference attacks, and file size inference attacks.

II. LOCATION GUARD

Location Guard scheme guards the location of each file and its access with two objectives: 1) to hide the actual location of a file and its replicas such that only legal users who hold the file's location key can easily locate the file on the overlay network and 2) to guard lookups on the overlay network from being eavesdropped by an adversary. Location Guard consists of three core components. The first component is location key, which controls the transformation of a filename into its location on the overlay network, analogous to a traditional cryptographic key that controls the transformation of plain text into ciphertext. The second component is the routing guard, which makes the location of a file unintelligible. The routing guard is, to some extent, analogous to a traditional cryptographic algorithm which makes a file's contents unintelligible. The third component of LocationGuard includes an extensible package of location inference guards that protect the file system from indirect attacks. Indirect attacks are those attacks that exploit a file's metadata information such as file access frequency, user IP-address, equivalence of file replica contents, and file size to infer the location of a target file on the overlay network.

A. Concepts and Definations

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 [4]-like lookup protocol T. Let $f^1, f^2 \ldots . f^R$ 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 f^i . A file lookup algorithm is defined as a function that accepts fi and outputs its location on the overlay network. Formally, we have T: $f^i \rightarrow$ *loc* maps a replica f^i to its location *loc* on the overlay network P.

B. Definition Location Key:

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

a. Given the location key of a file f, it is easy to locate the R replicas of file f.
b. Without knowing the location key of a file f, it is hard for an adversary to locate any of its replicas.
c. 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 ψ 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.

III. LOCATION KEYS

The first and most simplistic component of Location Guard is the concept of location keys. The design of location key needs to address the following two questions: 1) how to choose a location key and 2) how to use a location key to generate a replica location tokens —the capability to access a file replica. The first step in designing location keys is to determine the type of string used as the identifier of a location key. Let user u be the owner of a file f. User u should choose a long random bit string (128 bits) lk as the location key for file f. The second step is to find a pseudorandom function to derive the replica location tokens rlt^i (1<=i<=R) from the filename f and its location key lk. The pseudofilename rlti is used as a file replica identifier to locate the *i*th replica of file f on the overlay network.

IV. ROUTING GUARD

The second component of LocationGuard is the routing guard. The design of routing guard aims at securing the lookup of file f such that it will be very hard for an adversary to obtain the replica location tokens by eavesdropping on the overlay network. There are two possible approaches to implement a secure lookup algorithm: 1) centralized approach and 2) decentralized approach. In the centralized approach, one could use a trusted location server [5] to return the location of any file on the overlay network. However, such a location server would become a viable target for DoS and host compromise attacks.

A. Strength of Routing Guard

The strength of a routing guard refers to its ability to counter lookup sniffing-based attacks. A typical lookup sniffing attack is called the range sieving attack. Informally, in a range sieving attack, an adversary sniffs lookup queries on the overlay network and attempts to deduce the actual identifier rlt^i from its multiple obfuscated identifiers. We show that an adversary would have to expend $2^{\square\square}$years to discover a replica location token rlt^i even if it has observed 2^{25} obfuscated identifiers of rlt^i. Note that 2^{25} obfuscated identifiers would be available to an adversary if the file replica f^i was accessed once a second for one full year by some legal user of the file f. One can show that, given multiple obfuscated identifiers, it is nontrivial for an adversary to categorize them into groups such that all obfuscated identifiers in a group are actually obfuscations of one identifier. To simplify the description of a range sieving attack, we consider the worst case scenario where an adversary is capable of categorizing obfuscated identifiers (say, based on their numerical proximity).

V. LOCATTION INFERENCE GUARDS

Location inference attacks refer to those attacks wherein 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.

a. ***Lookup Frequency Inference Guard*:**

In this section, we present lookup frequency inference attack that would help a strategic adversary to infer the location of a target file on the overlay network. It has been observed that the general popularity of the web pages accessed over the Internet follows a Zipf-like distribution. An adversary may study the frequency of file accesses by sniffing lookup queries and match the observed file access frequency profile with a actual (predetermined) frequency profile to infer the location of a target file. Note that if the frequency profile of the files stored in the file system is flat (all files are accessed with the same frequency), then an adversary will not be able to infer any information.

b. User IP-Address Inference Guard:

In this section, we describe a user IP-address inference attack that assumes that the identity of a user can be inferred from his/her IP-address. Note that this is a worst case assumption; in most cases it may not possible to associate a user with one or a small number IP-addresses. This is particularly true if the user obtains IP-address dynamically(DHCP [7]) from a large Internet service provider (ISP).

B. 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.

a. File Replica Inference Guard:

Despite making the file capabilities and file access frequen cies appear random to an adversary, the contents of a file could by itself reveal the identity of the file f. The file f could be encrypted to rule out the possibility of identifying a file from its contents. Even when the replicas are encrypted, an adversary can exploit the fact that all the replicas of file f are identical. When an adversary compromises a good node, it can extract a list of identifier and file content pairs (or a hash of the file contents) stored at that node. Note that an adversary could perform a frequency inference attack on the replicas stored at malicious nodes and infer their filenames. Hence, if an adversary were to obtain the encrypted contents of one of the replicas of a target file f, it could examine the extracted list of identifiers and file contents to obtain the identities of other replicas. Once, the adversary has the locations of cr copies of a file f, the f could be attacked easily. This attack is especially more plausible on read-only files since their contents do not change over a long period of time. On the other hand, the update frequency on read/write files might guard them from the file replica inference attack.

b. File Size Inference Guard:

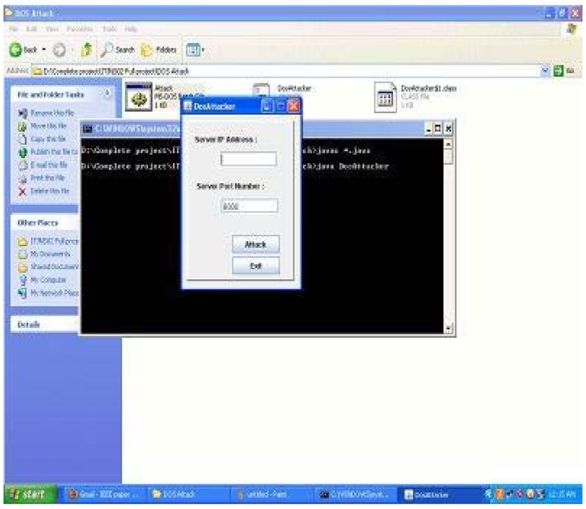
File size inference attack is based on the assumption that an adversary might be aware of the target file's size. Malicious nodes (and compromised nodes) report the size of the files stored at them to an adversary. If the size of files stored on the overlay network follows a skewed distribution, the adversary would be able to identify the target file (much like the lookup frequency inference attack). We guard the file system from this attack by fragmenting files into multiple file blocks of equal size. For instance, CFS divides files into blocks of 8 Kbytes each and stores each file block separately.

C. *Location Rekeying:*

In addition to the inference attacks listed above, there could be other possible inference attacks on a LocationGuard based 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.

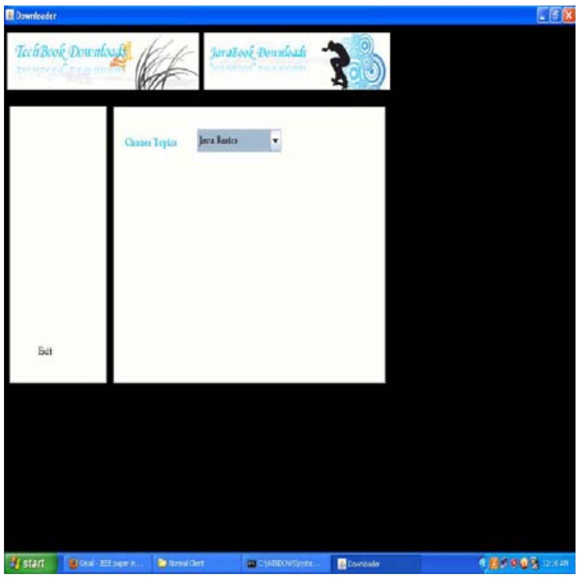
VI. EXPERIMENTAL EVALUATION

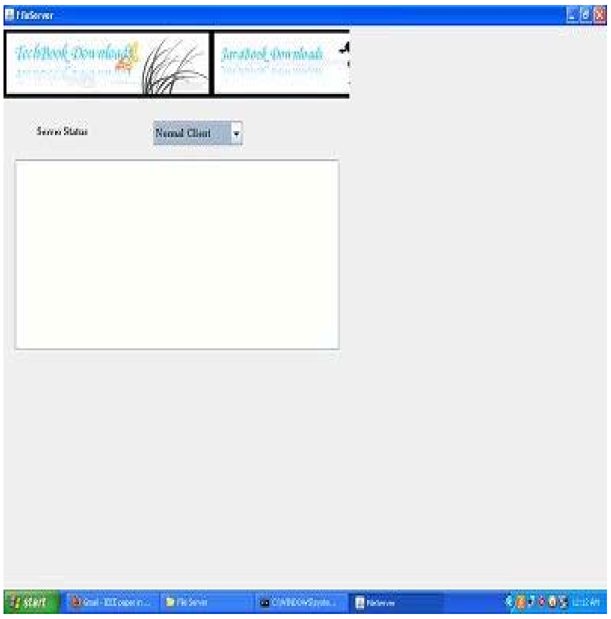
In this section, we briefly sketch our implementation of LocationGuard and quantify the overhead added by LocationGuard to the file system.



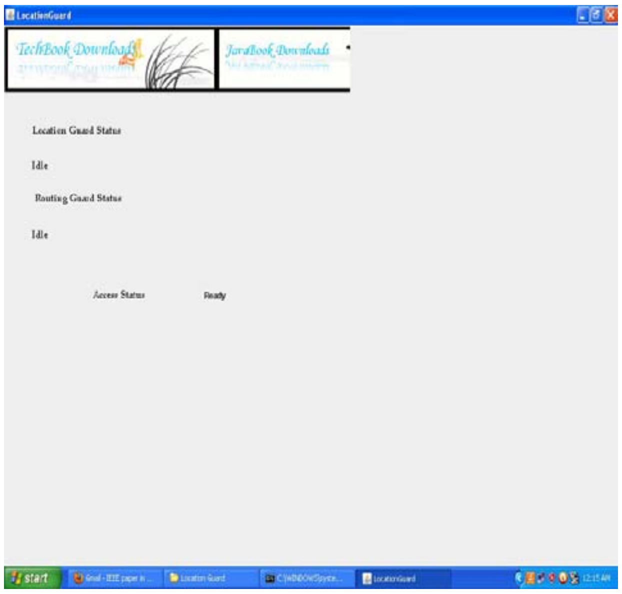
```
D:\Complete project\IT>javac *.java
D:\Complete project\IT>java DosAttacker
```

DOS Attacker





File Server



Location Guard Status

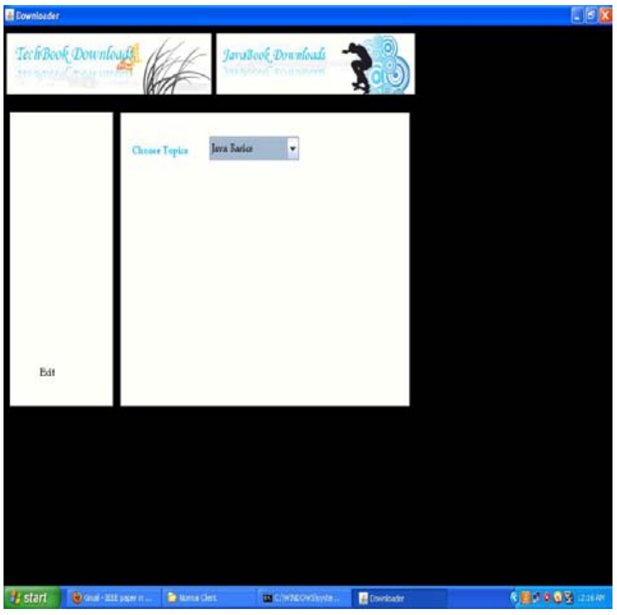
Idle

Routing Guard Status

Idle

Access Status Ready

Location Guard



Normal Client

VII. CONCLUSION

We have described that the Location Guard is a technique for securing wide-area server less file sharing systems from targeted file attacks. Analogous to traditional cryptographic keys that hide the contents of a file, Location Guard hides the location of a file on an overlay network. Location Guard 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 Location Guard 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 Location Guard scheme against DoS attacks, host compromise attacks, and various location inference attacks.

VIII. REFERENCES

[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 '0 1), Oct. 2001.

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

[4] I. Stoica, R. Morris, D. Karger, M. Kaashoek, and H. Balakrishnan, "Chord: A Scalable Peer-to-Peer Lookup Service for InternetApplications," Proc. ACM SIGCOMM '01, Aug. 2001.

[5] T. Jaeger and A.D. Rubin, "Preserving Integrity in Remote File Location and Retrieval," Proc. Ann. Network and Distributed System Security Symp. (NDSS), 1996.

[6] E.J. Go h, H. Shacham, N. Modadugu, and D. Bioneh, "SiRiUS: Securing Remote Untrusted Storage," Proc. 10th Ann. Network and Distributied System Security Symp. (NDSS), 2003.

[7] J. Kubiatowics, D. Bindel, Y. Chen, S. Czerwinski, P. Eaton, D. Geelis, R. Gum madi, S. Rhea, H. Weatherspo on, 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.

[8] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, "A Scalable Content-Addres sable Network," Proc. ACM SIGCOMM '01, Aug. 2001.

[9] A. Keromytis, V. Misra, and D. Rubenstein, "SOS: Secure Overlay Services," Proc. ACM SIGCOMM, 2002.