# Security Architectures and Algorithms for Overlay Network Services

**Ph.D. Proposal Document**

**Mudhakar Srivatsa**
College of Computing
Georgia Institute of Technology
mudhakar@cc.gatech.edu
http://www.cc.gatech.edu/∼mudhakar

## 1 Introduction

Overlay network computing systems and applications have continued to evolve over the past five years, ranging from SETI@Home and music sharing systems (Gnutella [18], KaZaa [27], and Limewire [32]) to more sophisticated applications, including file storage systems (cooperative file system [13], Farsite [2], and OceanStore [29]), publish-subscribe systems (Siena [9], Scribe [14], and Gryphon [4]), and Skype-like voice over IP (VoIP) systems. The overlay network computing model provides many opportunities for information dissemination across different organizational boundaries, heterogeneous platforms, and a large, dynamic population of users and has shown the potential to become a prominent network computing paradigm for massively distributed applications.

However, decentralized overlay network computing models also confront with various vulnerabilities and risks that impede such a new computing paradigm from being widely deployed in mission critical business systems and applications. There are three important security challenges for the overlay network service model: (i) confidentiality and integrity, (ii) authenticity, and (iii) availability. A curious node on the overlay network may threaten the confidentiality of application data. Using untrusted overlay nodes to deliver services threatens the authenticity of the service. Spamming, flooding and message dropping based denial of service (DoS) attacks may compromise the availability of the service. This dissertation research focuses on architectures and algorithms for building secure and scaleable information dissemination services on wide area overlay networks. Our research is conducted in three stages. First, we study application independent vulnerabilities and risks in overlay network computing mod-

els, focusing on targeted attacks on topology and peer to peer communication protocols. Second, we investigate on application specific vulnerabilities and security requirements for two important types of distributed information dissemination applications: event-driven publish-subscribe systems (push model) and file storage systems (pull model). In the third stage, we focus on developing security architecture and mechanisms for securing overlay network services against targeted attacks at the overlay network system layer, including failure and attack resilient overlay network topology and secure overlay routing protocols.

The ultimate goal of this dissertation research is to develop a secure architecture and a set of efficient algorithms to secure wide-area overlay network services. We refer to our architecture development as SGuard. SGuard comprises of a suite of security guards that can be seamlessly plugged into an overlay network service. Two categories of security guards are provided: the application independent security guards are used to guarantee availability [51, 53, 45, 46] and authenticity [54, 33, 55]. In addition, SGuard includes an application independent key dissemination mechanism for secure and scaleable distribution of keys to a large and dynamic population of users [47]. Given that the application independent guards for maintaining data confidentiality essentially reduce to secure multi-party computation (SMPC) protocols, which unfortunately are known to be inefficient and hard to scale. In SGuard, we resort to application specific techniques to guarantee data confidentiality and integrity. SGuard uses application specific guards to guarantee confidentiality and integrity for publish-subscribe overlay service [52, 50] and file storage service [49, 48]. We are building prototype implementations of several security guards to show that SGuard is easily stackable on an overlay network service. Our experimental results so far indicate that one can secure an overlay network service while preserving its performance and scalability metrics.

The autonomous nature of the overlay network service model is very similar to that of the Internet itself, allowing multiple content providers to efficiently publish data and deliver services to a large population of geographically scattered users. We believe that developing secure, efficient, and scalable techniques to guard overlay network services plays a very crucial role in making these services widely deployable.

# 2 Preliminaries

In this section, we describe the overlay network service model in detail. We also discuss several security threats on the overlay network service model.

## 2.1 Reference Overlay Network Service Model

This section presents a reference overlay network service model. An overlay network service model comprises of three entities: the content providers, the overlay network service provider, and the users. In this section we provide a reference model that describes the roles played by these three entities.

**Service Provider Models.** Advances in the networking technologies have triggered several networking services such as: 'software as a service' also referred to as the application service provider (ASP) model [24], 'database as a service' (DAS) [21, 20] that permits organizations to outsource their DBMS requirements, and 'storage as service' (SAS) model. These models allow organizations to leverage hardware and software solutions provided by the service providers, without having to develop them on their own, thereby freeing them to concentrate on their core businesses.

**Overlay Network Service Provider.** An overlay network is a virtual network that comprises of tens of thousands of nodes on top of an existing TCP/IP network. The topology of the nodes in the overlay network depends on the services hosted by the network. The performance and scalability of an application largely depends on the topology of the overlay network. For example, file sharing systems like Gnutella, KaZaa and Limewire use a power-law topology to support arbitrary keyword based search queries over a large collection of nodes; distributed file systems like cooperative file system, Farsite and OceanStore use distributed hash tables (such as Chord [58], CAN [36], Pastry [38], Tapestry [3], etc) to support efficient file lookups on the overlay network; a publish-subscribe system like Siena uses a tree-like topology to scaleably and efficiently disseminate events from a publisher to a group of subscribers; a VoIP system like Skype uses a geographical proximity based topology [37, 36] to obtain a low latency overlay path between the caller node and the receiver node.

The overlay network service model exports a set of interfaces to the content providers and the users. For example, in file sharing systems like Gnutella, KaZaa and Limewire the overlay network supports the `search` interface; in distributed file systems like cooperative file system, Farsite and OceanStore, the overlay network supports file operations `read`, `write`, `open` and `create`; in publish-subscribe systems like Siena, Scribe and Gryphon, the overlay network supports publish-subscribe operations `subscribe`, `publish`, `advertise`, `unsubscribe` and `unadvertise`; in a VoIP system like Skype, the overlay network supports operations like `session init` and `session end`.

**Content Provider.** The content provider owns the data either stored or processed by the overlay network. The overlay network nodes host the application data and the services on behalf of the content provider. The content provider may not trust the overlay network service provider with the confidentiality and integrity of application data. For example, in a file system scenario, the content provider may trust the overlay network nodes to perform read/write operations on a file; but the content provider may want to keep the file contents a secret (confidential) from the overlay network nodes; in a publish-subscribe system, the content provider may trust the overlay network nodes to scaleably disseminate events from a publisher to a group of subscribers; but the content provider may keep the event contents a secret (confidential) from the overlay network nodes. This problem of maintaining

data confidentiality from the service provider for arbitrary applications has been researched in the field of secure multi-party computation [66, 42]. However, these protocols are not deployed in practice because they incur heavy computation and communication overhead. In this thesis, we exploit application specific semantics to build secure, scalable and efficient algorithms to achieve acceptable levels of data confidentiality and integrity from the service provider. In particular, this thesis has focused on three important overlay network services: file sharing systems, publish-subscribe systems, and file storage systems.

The content provider may specify access control rules on the application data stored by the overlay network. These access control rules restrict the set of operations that can be performed by a given user on the application data hosted by the overlay network. Arbitrary access control rules represented as access control matrices [30] do not often result in efficient access control algorithms. Imposing a structure on the access control rules enables one to implement them efficiently and scaleably. In particular, we exploit application specific access structures represented as: directed acyclic graphs [62] & monotone Boolean expressions [39] for file storage services and constraints using binary operators [9] for publish-subscribe services.

**Users.** The users consume the application data either stored or processed by the overlay network. The users interact with the content providers to obtain authorization to perform certain operations on the application data and the services owned by the content provider. The users obtain their services from the overlay network service provider using well defined interfaces exported by the service provider. However, their interactions with the overlay network service provider are restricted by the access control rules specified by the content providers.

## 2.2 Security Threats and Challenges

In this section we discuss several possible attacks and security issues in an overlay network service model.

**Denial of Service (DoS) Attacks.** Recently we have seen increasing numbers of denial of service (DoS) attacks against online services and web applications either for extortion reasons, or for impairing and even disabling the competition [12, 31, 34]. These DoS attacks are increasingly mounted by professional attackers using huge zombie nets consisting of thousands of compromised machines on the Internet [23, 59, 57]. An FBI affidavit [12] describes a DoS attack on an e-Commerce website using a 5,000 node zombie net that caused a loss of several millions of dollars in revenue. Hence, countering DoS attacks on online services has become a very challenging problem.

The overlay network service provider has to protect the applications data hosted by the overlay nodes from DoS and host compromise attacks. Protecting the overlay network nodes from DoS and host compromise attacks improves service availability. In an overlay network service model, DoS attacks can target three different layers: (i) TCP/IP layer [40, 16, 6, 61], (ii) overlay network layer [10, 28], and the application layer [26, 63]. The over-

lay network service provider has to develop solutions to mitigate *insider* DoS attacks, wherein a set of malicious overlay nodes attempt to launch a DoS attack on the applications hosted by the overlay network.

**Authenticity Attacks.** The overlay network service provider has to protect the applications data hosted by the overlay nodes from incorrect or fake (spoofed) application data. Protecting the overlay network nodes from incorrect or fake application data guarantees the authenticity of application data hosted by the nodes. In an overlay network service model, authenticity attacks can be of two types: (i) an adversary may attempt to spoof the identity of a legitimate content provider and send incorrect or fake application data to the overlay network nodes [60], and (ii) an authentic content provider may flood the overlay network nodes with incorrect or inaccurate application data [65, 25]. The latter problem is prevalent in today's Internet wherein, we have multiple competitive web servers (with possibly conflicting interests) publish doctored information [19].

**Confidentiality and Integrity Attacks.** The overlay network service model has to protect the confidentiality and integrity from: (i) the overlay network nodes, and (ii) unauthorized users. As discussed in Section 2.1, for some applications, the content provider may not trust the overlay network service provider with the confidentiality and integrity of the application data. The malicious overlay network nodes may be able to eavesdrop or corrupt the application data hosted by them. In addition, malicious overlay nodes may collude with one another in their attempts to compromise the confidentiality and integrity of application data.

The overlay network service model allows the content provider to specify access control rules on application data. These access control rules restrict the set users that can access a given piece of application data hosted by the overlay network. However, malicious users may be curious to access application data and services that they are not authorized to access. In addition, malicious users may collude with one another and with the malicious nodes in the overlay network to compromise the confidentiality and integrity of application data.

**Key Distribution and Management.** An overlay network service model is faced with the challenge of having to meet the above security threats while preserving the performance and scalability of the application. Using cryptographic primitives to mitigate these security threats opens up new performance and scalability problems. Most cryptographic primitives assume an out-of-band distribution and management of cryptographic keys. In the overlay network service model, key distribution and management becomes a critical problem especially since the service provider typically employs tens of thousands of overlay nodes. Further, nodes can fail and leave the overlay network at a non-trivial rate; similarly, failed nodes can recover and join the overlay network at a non-trivial rate. Hence, the overlay network service model needs secure, efficient, and scalable key dissemination algorithms to handle a dynamic population of the overlay nodes, the content providers, and the users.

# 3 Contributions

SGuard aims at developing a suite of security guards to: (i) protect the interfaces exported by the overlay network service provider from denial of service (DoS) and host compromise attacks, (ii) protect the authenticity, confidentiality and integrity of application data as desired by the content provider, (iii) provide a secure key distribution & management algorithm for managing up to tens of thousands of overlay network nodes, and (iv) preserve the performance and scalability of the application while meeting requirements (i), (ii) and (iii).

**Denial of Service Attacks.** In an overlay network service model, DoS attacks can target three different layers: TCP/IP layer, overlay network layer, and the application layer. At the TCP/IP layer we protect the overlay nodes from SYN flooding and SYN+ACK flooding attacks using the port hiding guard [45]. At the overlay network layer, we protect the overlay nodes from routing attacks using DHTGuard [51]. At the overlay network layer, we have developed a novel location hiding algorithm that can hide the location of application data or a service on an overlay network [53]. Location of data on an overlay network refers to the IP address of the overlay network node that hosts the data. The key intuition here is that without knowing the location of a service, it is very hard for an adversary to launch a DoS attack on the service. At the application layer, we protect the overlay nodes from application layer attacks using the trust token guard [46].

**Authenticity, Confidentiality and Integrity Attacks.** In the overlay network service model, an authentic content provider may flood the overlay network nodes with incorrect or inaccurate application data. We have developed reputation management guards that enables users to filter out incorrect and inaccurate application data from malicious content providers [54, 33, 55].

We have developed application specific techniques to maintain confidentiality and integrity of application data from the overlay network service provider. EventGuard [52] presents techniques to protect the confidentiality and integrity of events in a publish-subscribe network from the publish-subscribe overlay network nodes. FSGuard [49] extends research on cryptographic file systems [17, 7, 11, 67, 64] and presents techniques to protect the confidentiality and integrity of files in an overlay network providing file storage service.

We have developed techniques to enforce flexible access control rules on application data as required by the content provider. Our algorithms exploit application specific access structures to make our security guards more scalable and efficient. We have developed techniques to enforce access structures based on constraints using binary operators in a publish-subscribe overlay network [50]. We have developed algorithms to enforce access structures based on directed acyclic graphs and monotone access structures in a file storage services [49, 48].

**Key Distribution and Management.** We use a cryptographic access control mechanism to enforce access control

rules and maintain data confidentiality and integrity from the overlay network nodes [22]. In a cryptographic access control mechanism, access control is enforced by distributing an encryption key used to encrypt a data unit to only those users who are authorized to access that data unit. Hence, an overlay network service provider is faced with the problem of having to efficiently and scaleably distribute and manage a large number of keys. We are developing secure, efficient and scalable algorithms for group key management in a large and dynamic overlay network [47].

**Prototypes and Implementation.** We have implemented all the security guards discussed above as pluggable software modules. Our security guards *hook* onto the application code making minimal or no changes to the primary application code base. We have advocated aspect oriented programming (AoP) [15, 1] principles in our implementations thereby, making our security guards look like wrappers around the primary application code base. In particular, we have built an overlay network file service built on top of [43] comprising of security guards [51, 53, 49, 48, 45, 46] and an overlay network publish-subscribe service built on top of [8, 44] comprising of security guards [52, 50, 54, 55, 45, 46]. As a part of our future work we plan to add a scalable key dissemination protocol to both the file service and the publish-subscribe service. Our experimental results so far indicate that our guards secure an overlay network service while preserving its performance and scalability metrics.

# 4 Mitigating Denial of Service and Host Compromise Attacks

In an overlay network service model, DoS attacks can target three different layers: TCP/IP layer, overlay network layer, and the application layer. We have developed techniques to handle DoS attacks at the TCP/IP layer [45], at the overlay network layer [51, 53], and at the application layer [46]. All our security guards to defend against DoS attacks are proactive and client-transparent. These guards are complementary to reactive mechanisms [40, 16] that detect and recover from a DoS attack. Client transparency follows from the following properties of our DoS guards: (i) no changes to client-side software, (ii) no client-side superuser privileges, and (iii) clients (human beings or automated clients) can access a DoS protected online service in the same manner that they browse other online services.

DHTGuard [51] exploits the structure of a distributed hash table (DHT) based overlay network to improve its resilience to DoS attacks. We improve the resilience of overlay networks to DoS attacks using three techniques: replicating application data, redundant routing, and verifiable routing. Replicating application data ensures that the data is available to users as long as a threshold number of replicas of the application data are available. Redundant routing using independent paths ensures that two nodes can communicate with one another even if some of the nodes are not operational either due to DoS attacks or host compromise attacks. Verifiable routing mechanisms

use DHT structures to probabilistically detect malicious nodes attempting to misguide the routing algorithm; on detecting a malicious behavior, a overlay node reroutes the message via an alternate path on the overlay network.

LocationGuard [53] uses a location hiding algorithm that can hide the location of application data or an online service on a large overlay network. Location of data on an overlay network refers to the IP address of the overlay network node that hosts the service. The key intuition here is that without knowing the location of a service, it is very hard for an adversary to launch a DoS attack on the service. The location hiding algorithm introduces the concept of a location key; analogous to a cryptographic key, a user can locate any named service on an overlay network if and only if the user knows the location key associated with that data unit. LocationGuard exploits the structure of the overlay network to obfuscate lookup queries on the overlay network. The obfuscation methodology uses a probabilistic one-sided error algorithm that preserves the performance and scalability of the lookup protocol, while making it hard for an adversary to guess a location identifier even after observing polynomial number of routing queries on the overlay network (polynomial in the number of bits of the location key). We have also studied inference attacks on our location hiding algorithm and proposed location rekeying based techniques to defend against them.

Even if the adversary were to locate the overlay node hosting the target application data or online service of interest, we introduce two guards at the overlay node's firewall or IP layer to defend against DoS attacks. Port hiding [45] operates at the TCP/IP layer and hides the TCP port number of the service hosted by the overlay network nodes. Unfortunately, the TCP port number has very low entropy (16-bits) thereby, making it tricky for an overlay node to hide it from an adversary. Hence, we use a combination of cryptographic techniques and game theoretic techniques to hide the TCP port number from the adversary. The key intuition is that without knowing the destination TCP port, the DoS attack packets from an adversary can be dropped at a firewall or at an IP layer packet filter on the overlay node.

Trust token guard [46] defends an online service hosted on overlay nodes from application layer DoS attacks. Application level DoS attacks refer to those attacks that exploit application specific semantics and domain knowledge to launch a DoS attack such that it is very hard for a DoS filter to distinguish between a sequence of attack requests and a sequence of legitimate requests. We use the amount of server resources incurred in handling the request to compute a score for every request. Every user's score is embedded in a secure cryptographic token. The overlay nodes use the priority level embedded in the cryptographic token to filter user's requests at its firewall or its IP layer.

## 5 Authenticity, Confidentiality and Integrity Attacks

Several research papers have proposed reputation management systems to condone content providers providing incorrect or inaccurate application data or services [65, 25]. Reputation management systems essentially create a feedback loop that allow content consumers (users) to rate the content providers based on the quality of the data obtained from the content providers (via the overlay network service provider). In course of time, all legitimate users will rely only on the data provided by reputed content providers; hence, low quality content providers would eventually run out of business. We have identified three important vulnerabilities in reputation management systems. First, we provide a dependable trust model and a set of formal methods to handle strategic malicious nodes that continuously change their behavior to gain unfair advantages in the system. Second, a transaction based reputation system must cope with the vulnerability that malicious nodes may misuse the system by flooding feedbacks with fake transactions. Third, but not the least, we identify the importance of filtering out dishonest feedbacks when computing reputation-based trust of a node, including the feedbacks filed by malicious nodes through collusion. We have built three security guards to defend against these attacks using cryptographic techniques and Byzantine fault-tolerant techniques [54, 55, 33].

We have developed techniques to protect the confidentiality and integrity of application data from the overlay network nodes for two services: publish-subscribe service and file service. EventGuard [52] presents techniques to protect the confidentiality and integrity of events in a publish-subscribe network from the overlay network nodes. We applied cryptographic techniques adapted using application (publish-subscribe system) specific knowledge to secure the application (publish-subscribe system). We developed six guards (security wrappers) around six publish-subscribe primitives: publish, subscribe, advertise, unsubscribe, unadvertise and route to protect the publish-subscribe service against confidentiality & integrity attacks, and authenticity attacks. We have also developed a $r$-resilient publish-subscribe overlay network topology that is resilient to overlay network level DoS attacks. We layer EventGuard on top of DoS guards [45, 46] to improve the resilience of overlay nodes against TCP/IP level DoS attacks and application level DoS attacks respectively. EventGuard preserves the content-aware event routing primitive in a publish-subscribe network thereby, maintaining its performance and scalability metrics.

FSGuard [49] presents techniques to protect the confidentiality and integrity of files in an overlay network providing file storage service. We support simple *nix like access control policies. We model these access control policies using access structures based on directed acyclic graphs (DAGs [62]). We apply cryptographic techniques from hierarchical access control to efficiently implement *nix like access control policies. We have developed four guards (security wrappers) around four important file storage primitives: read, write, open, and create to protect

the file storage service against confidentiality & integrity attacks and authenticity attacks. We layer FSGuard on top of DoS guards [51, 53, 45, 46] to make the file service resilience to DoS attacks.

We have developed techniques to support flexible access control rules on the application data provided by the content providers. ContentGuard [50] extends EventGuard [52] to support access control rules in a publish-subscribe service. In a publish-subscribe service access control rules are specified as constraints. For example, a subscriber $S$ who has subscribed for a subscription filter $\phi$ = $\langle\langle$`topic`, $EQ$, `cancerTrail`$\rangle$, $\langle$`age`, $>$, 20$\rangle\rangle$ should be able to read the patient record $rec$ in an event $e$ = $\langle\langle$`topic`, `cancerTrail`$\rangle$, $\langle$`age`, 25$\rangle$, $\langle$`patientRecord`, $rec\rangle\rangle$, but not the patient record $rec'$ in an event $e'$ = $\langle\langle$`topic`, `cancerTrail`$\rangle$, $\langle$`age`, 15$\rangle$, $\langle$`patientRecord`, $rec'\rangle\rangle$. We use the semantics of operators like $<$, $>$, $\geq$, $\leq$, $substring$, $superstring$, $prefix$, and $suffix$ to implement access control rules in an efficient and scalable manner. Using the above example, the authorization key given to a subscriber who subscribes to filter $\phi$ is $K_{20}^{\texttt{age}}$. The encryption key used to encrypt the patient record in event $e$ is $K_{25}^{\texttt{age}}$ and that in event $e'$ = $K_{15}^{\texttt{age}}$. We encode the semantics of the $>$ operator by ensuring that $K_x^{\texttt{age}}$ is efficiently computable from $K_y^{\texttt{age}}$ if and only if $x > y$. Hence, the subscriber $S$ can use the authorization key $K_{20}^{\texttt{age}}$ to derive the encryption key for the event $e$ ($K_{25}^{\texttt{age}}$), while making it computationally infeasible to guess the encryption key for the event $e'$ ($K_{15}^{\texttt{age}}$). This guarantees that the subscriber $S$ can decrypt the patient record in event $e$, but not the patient record in event $e'$

We have developed techniques [48] to extend FSGuard [49] to support monotone access structures in a file storage service. In a role-based access control (RBAC) model [39], users are associated with roles, which in turn represent a set of credentials. Files are annotated with monotone Boolean expressions on credentials. Hence, a user $u$ can access a file $f$ it the user $u$'s credentials satisfy the monotone access structure associated with the file $f$. For example, a file $f$ with monotone access structure $B_f$ = $g_1 \wedge (g_2 \vee g_3)$ should be intelligible to a user $u$ if and only if $u$ has the credential $g_1$ and either credential $g_2$ or $g_3$. Hence, a user $u_1$ with credentials $\{g_1, g_2\}$ should be able to access file $f$, but not a user $u_2$ with credentials $\{g_2, g_3\}$. Also, two colluding users $u_3$ with credential $g_1$ and $u_4$ with credential $g_3$ should not be able to access the file $f$ (unauthorized privilege escalation). We apply cryptographic techniques from secret sharing [41, 5] to implement discretionary access control using monotone access structures on a file storage service. We have also developed techniques to efficiently and scaleably handle a dynamic setting wherein a user's credentials and the access control expression associated with a file vary with time.

# 6 Key Distribution and Management

Using cryptographic access control mechanisms [22] to enforce access control rules and maintain data confidentiality and integrity from the overlay network nodes makes key distribution and key management an important problem. The goal of a key dissemination protocol is to securely, efficiently and scaleably disseminate a key from a key distribution center (KDC [56]) to all the group members. Traditional solutions to key distribution and key management protocols have used two alternate communication infrastructures. In a one-to-one communication infrastructure, the KDC is responsible for initializing and updating keys by communicating on a one-to-one secure channel with every node in the system. Hence, the cost of updating a key at the KDC is $O(N)$, where $N$ is the number of nodes in the system. The use of direct secure channels from the KDC to each node ensures the confidentiality & integrity of key update messages. However, the heavy load on the KDC questions the scalability of this approach and makes the KDC vulnerable to denial of service (DoS) attacks. A multicast communication infrastructure allows more efficient and scalable key management (survey in [35]). These protocols incur only $O(\log N)$ cost at the KDC and thus scale well. For improved scalability, these protocols do not rely on expensive public-key cryptography; instead they use fast and efficient symmetric key encryption algorithms. However, these protocols are not applicable to wide-area TCP/IP networks that lack the support for IP multicast.

We propose to explore an alternative communication infrastructure to disseminate group keys. Our proposal uses a wide-area TCP/IP based overlay network to build an efficient, scalable and secure key dissemination protocol. Our aim is to develop a key dissemination protocol wherein, the key update messages are routed through the nodes (group members) on the overlay themselves. Using the computing and networking resources at the nodes (instead of the KDC) ensures the KDC does not become a performance and scalability bottleneck in the system.

The fundamental security goals of a key management protocols include: *forward secrecy* and *backward secrecy*. Key management protocols satisfy forward and backward secrecy by disseminating an updated group key as and when the group membership changes. Using the computation and bandwidth resources at the nodes to propagate the key update message largely reduces the load on the KDC and improves the scalability of our approach. On the flip side, the nodes that form the overlay network (group members) may be untrusted or malicious. Hence, malicious nodes may be able to inspect the contents of the key update message routed through them (*confidentiality*) and attempt to corrupt (*integrity*) or spoof (*authenticity*) them. Additionally, malicious nodes may collude with one another to attack the key dissemination protocol (*collusion-resistance*). In a distributed overlay network based setting failures are inevitable (*fault-tolerance*). A node failure may be unintentional, as in a crash failure, or intentional, as in a DoS attack. For example, malicious nodes may also be able to launch DoS attacks by blocking

the key update messages routed through them.

In addition to these security challenges discussed above the key dissemination protocol should be efficient and scalable (*performance* and *scalability*). Scalability in our setting is measured using the load on the KDC and the load on the nodes in the overlay network due to the key dissemination protocol. Additionally, our key dissemination protocol should terminate in a small and finite time period (*timeliness*) and the protocol should ensure that all nodes on the overlay network receive the key update message (*completeness*).

We hope to build a key dissemination protocol using the notion of *broadcast authentication*. Informally, broadcast authentication on overlay networks is defined as follows. A message $M$ is broadcast authenticated with respect a broadcast protocol $P$ and a broadcast origin $n$ if for every node $m$ that receives the message $M$ can check the authenticity of $M$ and no node $m'$ on the broadcast path from node $n$ to node $m$ (node $n$ inclusive) can undetectably corrupt or spoof the message $M$. For performance and scalability reasons, we require our broadcast dissemination protocol to exploit the semantics of the broadcast protocol, the key update algorithm, and the structure of the overlay network.

# 7 Summary of Remaining Work and Schedule

For my dissertation defense I intend on completing the following:

- Develop secure event routing algorithms for content-based publish-subscribe networks with focus on common matching operations like: numeric attribute ranges, category trees & ontology, and string based prefix and suffix matching. We propose to embed the semantics of these matching operators in a efficiently derivable key space to permit efficient and scalable key management.

- Develop key derivation algorithms for maintaining file data confidentiality and integrity using *nix-like access structures in cryptographic systems. We propose to use techniques from hierarchical access control and one-way key trees to embed *nix-like access structures.

- Develop a secure and scalable key dissemination protocol to deliver keys to a large and dynamic population users. Previous approaches use a centralized key distribution center or assume a multicast network. We propose to use the computation and communication resources available at the untrusted nodes in an overlay network for secure and scalable key dissemination.

I plan to defend in spring 2007.

# References


[1] NetFilter/IPTables project homepage. http://www.netfilter.org/.

[2] 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. In *Proceedings of the 5th International Symposium on OSDI*, 2002.

[3] J. K. B. Zhao and A. Joseph. Tapestry: An infrastructure for fault-tolerance wide-area location and routing. Technical Report UCB/CSD-01-1141, University of California, Berkeley, 2001.

[4] G. Banavar, T. Chandra, B. Mukherjee, and J. Nagarajarao. An efficient multicast protocol for content-based publish subscribe systems. In *Proceedings of the 19th ICDCS*, 1999.

[5] J. Benaloh and J. Leichter. Generalized secret sharing and monotone functions. In *Proceedings of CRYPTO*, 1988.

[6] D. J. Bernstein. SYN cookies. http://cr.yp.to/syncookies.html.

[7] M. Blaze. A cryptographic file system for unix. In *Proceedings of ACM CCS*, 1993.

[8] A. Carzaniga. Siena - software. http://serl.cs.colorado.edu/ carzanig/siena/software/index.html.

[9] A. Carzaniga, D. S. Rosenblum, and A. L. Wolf. Design and evaluation of a wide-area event notification service. In *ACM Transactions on Computer System, 19(3):332-383*, 2001.

[10] M. Castro, P. Druschel, A. Ganesh, A. Rowstron, and D. S. Wallach. Secure routing for structured peer-to-peer overlay networks. In *Proceedings of 5th OSDI*, 2002.

[11] G. Cattaneo, L. Catuogno, A. D. Sorbo, and P. Persiano. The design and implementation of transparent cryptographic file system for unix. In *Proceedings of Annual USENIX Technical Conference*, 2001.

[12] CERT. Incident note IN-2004-01 W32/Novarg.A virus, 2004.

[13] F. Dabek, M. F. Kaashoek, D. Karger, R. Morris, and I. Stoica. Wide-area cooperative storage with CFS. In *Proceedings of the 18th ACM SOSP*, October 2001.

[14] A. K. Datta, M. Gradinariu, M. Raynal, and G. Simon. Anonymous publish/subscribe in P2P networks. In *Proceedings of IEEE IPDPS*, 2003.

[15] Eclipse. Aspectj compiler. http://eclipse.org/aspectj.

[16] R. Ferguson and D. Senie. RFC 2267: Network ingress filtering: Defeating denial of service attacks which employ IP source address spoofing. http://www.faqs.org/rfcs/rfc2267.html, 1998.

[17] K. Fu, M. F. Kaashoek, and D. Mazieres. Fast and secure distributed read-only file system. In *Proceedings of the 4th OSDI, pp: 181-196*, 2000.

[18] Gnutella. The gnutella home page. http://gnutella.wego.com/.

[19] Z. Gyongyi, H. Garcia-Molina, and J. Pederson. Combating web spam with trustrank. In *Proceedings of 30th VLDB Conference*, 2004.

[20] H. Hacigumus, B. Iyer, C. Li, and S. Mehrotra. Executing sql over encrypted data in the database-service-provider model. In *Proceedings of the ACM SIGMOD*, 2002.

[21] H. Hacigumus, B. Iyer, and S. Mehrotra. Providing database as a service. In *Proceedings of 18th IEEE ICDE*, 2002.

[22] A. Harrington and C. Jensen. Cryptographic access control in a distributed file system. In *Proceedings of the 8th ACM SACMAT*, 2003.

[23] E. Hellweg. When bot nets attack. MIT Technology Review, September 2004.

[24] IBM. Application service provider business model. http://www.redbooks.ibm.com/abstracts/sg246053.html.

[25] S. Kamvar, M. Schlosser, and H. Garcia-Molina. Eigenrep: Reputation management in P2P networks. In *Proceedings of 12th World Wide Web Conference (WWW)*, 2003.

[26] S. Kandula, D. Katabi, M. Jacob, and A. Berger. Botz-4-sale: Surviving organized DDoS attacks that mimic flash crowds. In *Proceedings of 2nd USENIX NSDI*, 2005.

[27] Kazaa. Kazaa home page. http://www.kazaa.com/, 2003.

[28] A. Keromytis, V. Misra, and D. Rubenstein. SOS: Secure overlay services. In *Proceedings of the ACM SIGCOMM*, 2002.

[29] 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. In *Proceedings of the 9th International Conference on Architectural Support for Programming Languages and Operating Systems*, November 2000.

[30] B. Lampson. Protection. In *Proceedings of the 5th Princeton Symposium on Information Sceinces and Systems, pp: 437-443*, 1971.

[31] J. Leyden. East european gangs in online protection racket. www.theregister.co.uk/2003/11/12/east-european-gangs-in-online/.

[32] LimeWire. Improving gnutella protocol: Protocol analysis and research proposals. http://www9.limewire.com/download/ivkovic_paper.pdf, 2002.

[33] S. Park, L. Liu, C. Pu, M. Srivatsa, and J. Zhang. Resilient trust management for web service integration. In *Proceedings of 3rd Intl Conference on Web Services (ICWS)*, 2005.

[34] K. Poulsen. FBI busts alleged DDoS mafia. www.securityfocus.com/news/9411, 2004.

[35] S. Rafaeli and D. Hutchison. A survey of key management for secure group communication. In *Journal of the ACM Computing Surveys, Vol 35, Issue 3*, 2003.

[36] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A scalable content-addressable network. In *Proceedings of SIGCOMM Annual Conference on Data Communication*, Aug 2001.

[37] M. Ripeanu, I. Foster, and A. Iamnitchi. Mapping the gnutella network: Properties of large-scale p2p systems and implications for system design. In *IEEE Internet Computing Journal, vol. 6, no. 1, 2002*.

[38] A. Rowstron and P. Druschel. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In *Proceedings of the 18th IFIP/ACM International Conference on Distributed Systems Platforms (Middleware 2001)*, Nov 2001.

[39] R. Sandhu, E. Coyne, H. Feinstein, and C. Youman. Role-based access control models. In *IEEE Computer, Vol. 29, No. 2*, 1996.

[40] S. Savage, D. Wetherall, A. Karlin, and T. Anderson. Practical network support for IP traceback. In *Proceedings of ACM SIGCOMM*, 2000.

[41] A. Shamir. How to share a secret? In *Communications of the ACM*, 1979.

[42] V. Shoup. On formal models for secure key exchange. Technical Report RZ 3120, IBM Research, 1999.

[43] A. Singh, M. Srivatsa, L. Liu, and T. Miller. Apoidea: A decentralized peer to peer architecture for crawling the world wide web. In *Proceedings of SIGIR 2003 Workshop on Distributed Information Retrieval, Aug 2003. An extended version appears in Lecture Notes of Computer Science (LNCS) Series, Springer Verlag*.

[44] M. Srivatsa, B. Gedik, and L. Liu. Scaling unstructured peer-to-peer networks with multi-tier capability aware topologies. In *Proceedings of IEEE Transactions on Parallel and Distributed Systems (TPDS). An extended abstract of this paper appeared in 10th IEEE Intl Conference on Parallel and Distributed Systems (ICPADS 2004)*, 2005.

[45] M. Srivatsa, A. Iyengar, J. Yin, and L. Liu. A client-transparent approach to defend against denial of service attacks. Under Submission.

[46] M. Srivatsa, A. Iyengar, J. Yin, and L. Liu. A middleware system for protecting against application level denial of service attacks. Under Submission.

[47] M. Srivatsa and L. Liu. Efficient group key management protocols for large scale overlay networks. In Preparation.

[48] M. Srivatsa and L. Liu. Key derivation algorithms for monotone access structures in large file systems. Under Submission.

[49] M. Srivatsa and L. Liu. Key derivation algorithms for *nix-like access structures in cryptographic file systems. In Preparation.

[50] M. Srivatsa and L. Liu. Secure event routing in content-based publish-subscribe networks. In Preparation.

[51] M. Srivatsa and L. Liu. Vulnerabilities and security issues in structured overlay networks: A quantitative analysis. In *To Appear in the Proceedings of Springer Intl Journal on Information Security. An extended abstract of this paper appeared in Proceedings of 20th IEEE Annual Computer Security Applications Conference (ACSAC)*, 2004.

[52] M. Srivatsa and L. Liu. Eventguard: Secure event notification architecture for publish-subscribe networks. In *Proceedings of 12th ACM Conference on Computer and Communication Security (CCS)*, 2005.

[53] M. Srivatsa and L. Liu. Locationguard: Countering targeted file attacks using location keys. In *Proceedings of 14th USENIX Security Symposium*, 2005.

[54] M. Srivatsa and L. Liu. Securing decentralized reputation management using trustguard. In *To Appear in the Proceedings of Journal on Parallel and Distributed Computing (JPDC), special issue on Security in Grid and Distributed Systems. An extended abstract of this paper appeared in the Proceedings of 14th World Wide Web (WWW 2005) Conference*, 2006.

[55] M. Srivatsa, L. Xiong, and L. Liu. Xchange: A distributed protocol for electronic fair exchange. In *Proceedings of 19th IEEE Intl Parallel and Distributed Processing Symposium (IPDPS)*, 2005.

[56] W. Stalling. Cryptography and network security: Principles and practice. Prentice Hall; ISBN: 0130914290, 2002.

[57] S. Staniford, V. Paxson, and N. Weaver. How to own the internet in your spare time. In *Proceedings of USENIX Security*

*Symposium*, 2002.

[58] I. Stoica, R. Morris, D. Karger, M. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *Proceedings of SIGCOMM Annual Conference on Data Communication*, August 2001.

[59] L. Taylor. Botnets and botherds. http://sfbay-infragard.org.

[60] C. Wang, A. Carzaniga, D. Evans, and A. L. Wolf. Security issues and requirements for internet-scale publish-subscribe systems. In *Proceedings of the 35th Hawaii International Conference on System Sciences*, 2002.

[61] X. Wang and M. K. Reiter. Defending against denial-of-service attacks with puzzle auctions. In *Proceedings of IEEE Symposium on Security and Privacy*, 2003.

[62] Waterken. Web calculus. http://www.waterken.com/dev/Web/Calculus/.

[63] B. Waters, A. Juels, A. Halderman, and E. W. Felten. New client puzzle outsourcing techniques for dos resistance. In *Proceedings of 11th ACM CCS*, 2004.

[64] C. P. Wright, M. C. Martino, and E. Zadok. Ncryptfs: A secure and convinient cryptographic file system. In *Proceedings of Annual USENIX Technical Conference*, 2003.

[65] L. Xiong and L. Liu. Peertrust: Supporting reputation-based trust for peer-to-peer electronic communities. In *Proceedings of IEEE TKDE, Vol. 16, No. 7*, 2004.

[66] A. Yao. Protocols for secure computations. In *Proceedings of 23rd IEEE Symposium on FOCS, pp: 160-164*, 1982.

[67] E. Zadok, I. Badulescu, and A. Shender. Cryptfs: A stackable vnode level encryption file system. Technical Report CUCS-021-98, Columbia University, 1998.