MITIGATING ATTACKS AGAINST ADAPTATION MECHANISMS IN OVERLAY NETWORKS

A Thesis

Submitted to the Faculty

of

Purdue University

by

AAron R. Walters

In Partial Fulfillment of the

Requirements for the Degree

of

Master of Science

May 2006

Purdue University

West Lafayette, Indiana

TABLE OF CONTENTS

Page

Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

ABSTRACT

Walters, AAron R. M.S., Purdue University, May, 2006. Mitigating Attacks Against Adaptation Mechanisms in Overlay Networks. Major Professor: Cristina Nita-Rotaru.

Performance-aware measurement-based overlay networks offer increased performance and resilience to benign failures for end-to-end communication. Adaptation mechanisms are a critical component of overlay network design, enabling distributed construction of efficient overlays for performance-demanding applications in heterogeneous Internet settings. These mechanisms dynamically optimize application-centric metrics such as latency, jitter, bandwidth, and loss rate. However, end-systems are more vulnerable than core routers, making overlay networks susceptible to malicious attacks coming from untrusted outsiders, and especially from trusted (but compromised) members of the overlay. Unlike many outsider attacks, insider (or Byzantine) attacks can not be prevented by simply deploying cryptographic authentication mechanisms. In this work, we identify and classify insider attacks against adaptation mechanisms in overlay networks and demonstrate several of them against a mature, operationally deployed overlay multicast system. The attacks target the overlay network construction, maintenance, and availability and allow malicious nodes to control significant traffic in the network, facilitating further attacks such as selective forwarding, traffic analysis, and overlay partitioning. We present a comprehensive defense framework to address the identified attacks, focusing on a critical component, reducing the number of bad or unnecessary adaptations. We demonstrate the effectiveness of the newly proposed techniques through real-life deployments and emulations conducted on the PlanetLab and DETER testbeds, respectively.

1 INTRODUCTION

Multicast is a communication service that provides efficient and simultaneous information dissemination to a set of distributed nodes. The service has proven to be a critical component for collaborative applications operating over the Internet, such as video/audio conferencing, streaming media, digital content distribution, and networked multi-player games.

The Internet Protocol (IP) layer was traditionally recognized as the best layer for building multicast functionality [1]. However, IP multicast, a best effort datagram protocol, had limited deployment and acceptance due to its infrastructure dependencies. To overcome these limitations, the research community has offered an alternative architecture where multicast functionality is implemented by end-systems and not by the infrastructure supporting routers. These end-systems communicate via uni-cast links on the underlying network in order to form a multicast group built on an overlay structure. In multicast overlay networks, buffering and relaying respon-

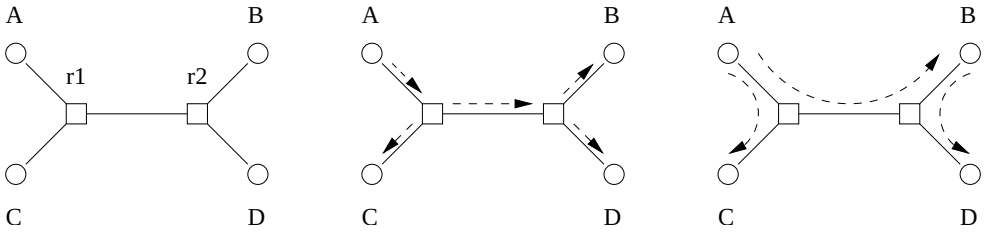


Figure 1.1. This figure graphically depicts a (a) network topology, (b) IP Multicast on that topology, (c) and an overlay multicast network. The squares represent routers (r1 and r2), while the circles represent end systems (A,B,C,D). End system A is the source of a multicast session.

sibilities are migrated from core routers toward end-systems and from the network layer to the application layer, as seen in Figure 1.1(c). Research has also demonstrated that despite the potential added inefficiencies with regard to bandwidth and latency, this migration is fully justified according to commonly held end-to-end arguments [2].

While pushing functionality to end-systems allows overlay networks to address the limitations associated with IP multicast, it also makes them potentially more vulnerable. This is due to the fact that trust is also being pushed to the fringes of the Internet, where end-nodes are often general purpose machines which are more likely to get compromised than core routers [3]. In addition, the functionality of each node is dependent on the links it maintains to a set of neighbor nodes, which can also be potentially compromised and act adversarially. As a result, end-system overlay networks are more vulnerable to malicious outsider attacks, as well as to insider attacks coming from (potentially colluding) attackers that infiltrate the overlay or compromise member nodes. In the following sections, we provide more background about overlay networks and attacks that can occur in such networks. We then state our contributions and outline the organization of this thesis.

1.1 Overlay Networks

Overlay networks are logical networks typically formed from a network layer built on top of native communication services, as seen in Figure 1.1. Such networks offer new functionality, performance, robustness, anonymity, or isolation that are not typically possible with the existing infrastructure. These services are achieved by distributing the responsibility to construct and maintain the overlay among the overlay nodes. For example, members of the overlay are often required to collaborate to provide the necessary routing infrastructure and, if necessary, marshal themselves into a topology that supports the functionality and efficiency requirements of the

overlay. The logical links of the overlay network are an abstraction of links provided by the underlying physical network.

Overlay networks provide the ability to route information using application specific criteria beyond what can typically be expressed by the IP layer. This ability has resulted in two main uses of overlay networks: distributed file sharing and multicast. File sharing, or peer-to-peer data sharing networks, create overlay networks to facilitate discovering and transferring data stored among protocol peers. Multicast, or overlay broadcast multiplexing, involves efficiently and simultaneously delivering information to interested parties with the minimum amount of delay, while maintaining a certain bandwidth.

Two types of overlays have been proposed to support these functionalities: unstructured and structured [4]. This categorization is typically tied to the degree of constraints that are placed on neighbor set determination. An unstructured overlay is characterized by unconstrained neighbor set selection. In unstructured file sharing overlays, nodes are marshalled into a random graph topology, while flooding or random walks are used to find data stored throughout the overlay network [4]. Unstructured multicast overlay schemes are generally seen as those where nodes are unconstrained in their organization of multicast trees or other distribution topologies. A perceived negative of unstructured overlays is that the lack of constraints sometimes leads to cases of unbounded inefficiencies either in the number of nodes needing to be traversed to find the desired data or the amount of time it takes to send data to a particular node.

Structured overlays were proposed in an attempt to bound these inefficiencies [5–8]. In such overlays, only a small subset of nodes meeting presubscribed organizational conditions are eligible to become neighbors of a particular node. In structured file sharing overlays, data is uniquely identified and the overlay nodes are organized in a topology that offers bounds on locating an item and the number of network hops. Structured multicast has grown out of the work in file sharing by leveraging the organization of nodes, typically some form of Distributed Hash Table (DHT)

overlay network, to build the multicast forwarding trees on top of. Thus, neighbor selection is constrained by the underlying DHT protocol. While offering bounded guarantees, the constraints associated with structured overlay networks often has a detrimental impact on the performance and resilience of the overlay, often times making them prohibitive for multicast applications.

Unstructured overlay networks often utilize measurement based adaptation mechanisms to increase or maintain performance and provide fault tolerance for end-to-end communication. These mechanisms are typically used to find some type of local performance extremum or equilibrium point. Adaptation is particularly important in overlay multicast where designers are concerned with maintaining optimum performance or maintaining at least a minimum level of quality of service. Adaptation mechanisms used by performance-driven overlays seek to dynamically optimize application-centric metrics such as latency, jitter, bandwidth, and loss rate. Such metrics are typically collected by overlay nodes through passive observation of their performance from the source (primary metrics) and through periodic probing of peer nodes about their performance from the source (secondary metrics). The resulting performance of the overlay network depends on the accurate interpretation of performance observations, as well as the correctness of the responses received from probed nodes.

1.2 Attacks in Overlay Networks

A substantial amount of work has been done to ensure overlay networks are resilient to node failures and performance degradation. The true potential of an overlay network relies on its ability to be deployed in a distributed heterogeneous environment that brings together disjoint distrusting parties in a common communication infrastructure. However, the Internet has demonstrated that protocols deployed in these open environments can no longer be designed under an implicit assumption of trust among peering nodes [9, 10]. Open environments introduce serious threats

posed by selfish and malicious nodes which may conspire to disrupt or destroy the overlay. As a result, overlay networks must be designed to be resilient to not just benign failures, but to selfish and malicious failures as well. In general, the term of *selfish* attacker is used to describe an attacker that seeks personal gain with minimum effort and without having destructive goals, while the term of *malicious attacker* is used to describe attackers with destructive goals against the system.

Contributing to the lack of trust among nodes in an open environment is the fact that nodes participating in the overlay often have conflicting objectives. These conflicting objectives are often at the heart of selfish behavior. One example of selfish behavior involves a parasitic relationship where a selfish node utilizes the resources of the overlay while trying to minimize their burden for supporting others. These attacks are also sometimes referred to as cheating or attacks on fairness and have been of particular interest in the structured file sharing overlays [11, 12]. Several solutions were proposed in attempts to address these problems including micropayment schemes [11], reputation schemes [13], barter economies [11], and game theory based mechanisms [14]. These techniques have been used in an attempt to make sure that nodes participating in the overlay take an equitable share of the resource burden. Anonymity is another technique that has been used to address selfish behavior [11] and attacks such as traffic analysis [15]. The solutions offered to address selfish behavior are inadequate to address destructive attacks performed by malicious attackers.

To date, the majority of the research addressing malicious attacks on overlay networks has focused on structured file sharing protocols. These attacks utilize the fact that the information used by overlay protocols, is often blindly trusted and unverified. As a result, misbehaving nodes have the propensity to cause a great deal of damage by generating falsified information. The initial work in this area focused on enumerating attacks that threaten the "liveness" of the main service of such systems: looking up and retrieving particular data based on the associated key. A number of attacks – that can be classified as reduction of quality attacks or full scale denial

of service attacks – against structured file sharing applications are presented in [16]. Some of the described attacks could be solved by deploying cryptographic authentication mechanisms, while other attacks, most notably against routing, require more complex solutions. Through routing attacks, attackers can control file discovery by manipulating the control and data messages routed within the overlay, poisoning the route table of neighboring nodes, or partitioning the network.

Later work [17] reemphasized the magnamity of the threat posed by routing attacks against structured overlays under a stronger adversary model where the faulty nodes can behave arbitrarily (Byzantine) and collude together to form coalitions. The authors offered a number of partial solutions which exploit the intrinsic features of structured overlays. The proposed solutions mitigate the attacks by leveraging the strong organizational constraints placed on neighbor selection in structured overlays and the existent invariant relationships between neighbors.

A subset of these types of attacks, referred as Eclipse attacks [18, 19], were subsequently studied in optimized structured file sharing overlays. In Eclipse attacks, attackers attempt to control a large fraction of the neighbors of a correct node in order to "eclipse" correct nodes, by controlling a substantial amount of messages in the overlay, and by completely mediating the interaction of a node with the overlay. One of the main vulnerabilities attackers exploit is the ability to forge identities and create large numbers of apparently distinct nodes, also known as the Sybil attack [20]. As with the previous solutions in structured overlays the solution offered by the authors focused on leveraging invariant constraints. However, instead of utilizing organizational invariants associated with node identifiers they enforce degree constraint invariants associated with neighbors, supported by anonymous auditing. Unlike previous solutions, the proposed solution supports proximity neighbor selection [21, 22], a technique used in neighbor selection to improve the efficiency of overlays. While solutions for detecting Eclipse attacks offer valuable insight into the problem space and are appropriate in structured overlay networks, they will

not be sufficient against attacks in performance driven dynamical systems such as measurement-based extremum overlay multicast.

Unlike the aforementioned examples in structured file sharing overlays, performance aware measurement-based multicast overlays are a dynamical distributed system that can be modeled by a nonlinear adaptive multi-variable control system. While previous research focused on structured overlays that attempted to optimize queries apriori based on measured values [21], there was no dynamic response to disturbances. In those systems, state was updated either lazily when a node failed or maintenance was performed periodically at large time scales under the assumption that the proximity metric reflects a static property of the underlying network [21]. Dynamical systems, on the other hand, continually attempt to adapt in response to disturbances while maintaining an acceptable level of performance towards the apriori defined objective of the system. Thus it is extremely important that these systems reduce the impact of a potential threat and maintain as high a level of performance as possible. As a result, a solution in such systems can ill afford the heavy handed constraints of structured overlays or the extra overhead of anonymous auditing. The first research to consider misbehaving nodes in this class of multicast overlay networks was [23] which described selfish behavior but did not address the problem of malicious adversaries. The only work that we are aware of to consider malicious attacks on adaptive control loops in network protocols deployed in an open environment was [9,10]. In both examples, they focused on demonstrating the problem without offering a viable solution. More specifically, [9] showed the vulnerability of TCPs adaptive congestion control mechanism in order to perform denial of service, whereas [10] focused on reduction of quality attacks against the same mechanism.

1.3 Thesis Focus and Contributions

This thesis constitutes the first effort to identify and analyze a set of advanced threats against adaptive unstructured measurement based overlay networks. Our

work builds on pioneering work in showing how attackers can exploit adaptivity in two-party protocols [9, 10], while considering the effects of insider adversaries in the context of performance-driven overlay networks. Current adaptation mechanisms lack Byzantine-resilience and assume that the information reported by probed nodes is always correct. Furthermore, such mechanisms fail to take into account the effects of Byzantine attackers on their surrounding environment. Attackers in close proximity to a given node may influence the network in order to manipulate the metrics collected and the subsequent decisions made by that node.

More specifically, we provide an introduction to attacks against standard adaptation mechanisms in unstructured performance-driven measurement-based overlay networks, an initial analysis of how such attacks can be mitigated and prevented throughout the life-cycle of the overlay, and an in-depth solution to a critical aspect of the problem: preventing poor adaptation decisions in networks influenced by attackers. Our solution lies in performing spatial and temporal outlier analysis on primary (measured) and secondary (probed) metrics to allow an honest node to make better use of available information before making an adaptation decision. Furthermore, we demonstrate the effects of the identified attacks on a mature adaptive overlay network operating over real Internet infrastructure. Finally, we experimentally show the usefulness of our outlier detection technique for preventing bad decisions in the face of Byzantine attackers. We summarize our key contributions:

1. We provide a characterization of the types of mechanisms currently used to achieve adaptivity in overlay networks and identify attacks against these mechanisms. We refer to these attacks, which target performance-driven measurement-based overlay construction, maintenance, and stability, as *attraction*, *repulsion*, and *disruption*.

2. We demonstrate the effectiveness of the above identified attacks against a well-known and operationally deployed adaptive measurement-based multicast system, ESM [24]. Our experiments, which were conducted using both real-life

deployments and emulations, demonstrate that, although ESM employs an advanced set of adaptation mechanisms, it is unable to mitigate the attacks posed by a malicious adversary.

3. We provide an analysis of the solution space for mitigating Byzantine attacks that exploit adaptivity: preventing unnecessary or unnatural adaptations, increasing stability by incorporating metrics that reflect stability into the decision process, detecting malicious behavior that results in observable degradation of service, and reacting to the detected malicious nodes.

4. We focus on providing a solution for what we believe is the most critical and challenging problem: preventing bad adaptations. We propose techniques to reduce incorrect and unnecessary adaptations by using spatial and temporal correlations to perform context-sensitive outlier analysis. A key component of our solution is based on the observation that several estimated metrics are dependent variables and the overlay and multicast tree logical networks share overlapping physical links.

5. We demonstrate the effectiveness of our defense mechanisms in the context of the ESM system through experiments conducted on the PlanetLab [25] and DETER [26] testbeds.

1.4 Thesis Roadmap

The rest of the thesis is organized as follows: we provide an overview of adaptive control systems and a survey of adaptation mechanisms employed by overlay networks in Chapter 2. Then we provide an abstract system model, attack model, and classify attacks against measurement-based adaptation mechanisms in unstructured performance-driven overlay networks in Chapter 3. We then demonstrate several attacks against the adaptation mechanisms employed by ESM in Chapter 4 and propose defense mechanisms in Chapter 5. In Chapter 6, we experimentally evaluate

our new technique. We overview related work in Chapter 7. Finally, we present our conclusions in Chapter 8.

2 ADAPTATION MECHANISMS IN OVERLAY NETWORKS

The benefits of employing adaptivity to address intermittent failures and degraded performance associated with dynamic network conditions has been recognized since the earliest days of the ARPANET [27]. Adaptation mechanisms based on measurements of network characteristics have been used more recently in the design of overlay networks [24, 28–30], wireless networks [31] and sensor networks [32].

In this chapter we provide an overview of the main adaptation mechanisms used in overlay networks. We first describe the general model of a classic feedback control system, then describe in detail adaptation mechanisms used in overlay networks.

2.1 Adaptive Control Systems

Historically, adaptive control systems have been proposed in order to build systems that possessed the mechanisms to self-regulate and adapt to external and internal disturbances [34]. Adaptive control systems can be modeled by a classic feedback loop typically found in control systems, Figure 2.1, where the two inputs are the goal and the disturbances [33]. The goal represents the desired state of the system, and the disturbances represent the aspects of the environment that are outside the control of the system, but have the potential to influence the essential variables that determine the system state. The system estimates its current state in relation to its goal by using sensors or measurements to sample the space of observed variables. The samples are then transformed into an internal representation which reflects the systems perception of the environment. As seen in Figure 2.1, during this perception process passive mechanisms are typically employed to suppress the perturbations that may have affected the observed variables and subsequently to improve the data quality. During information processing system designers also employ another set of

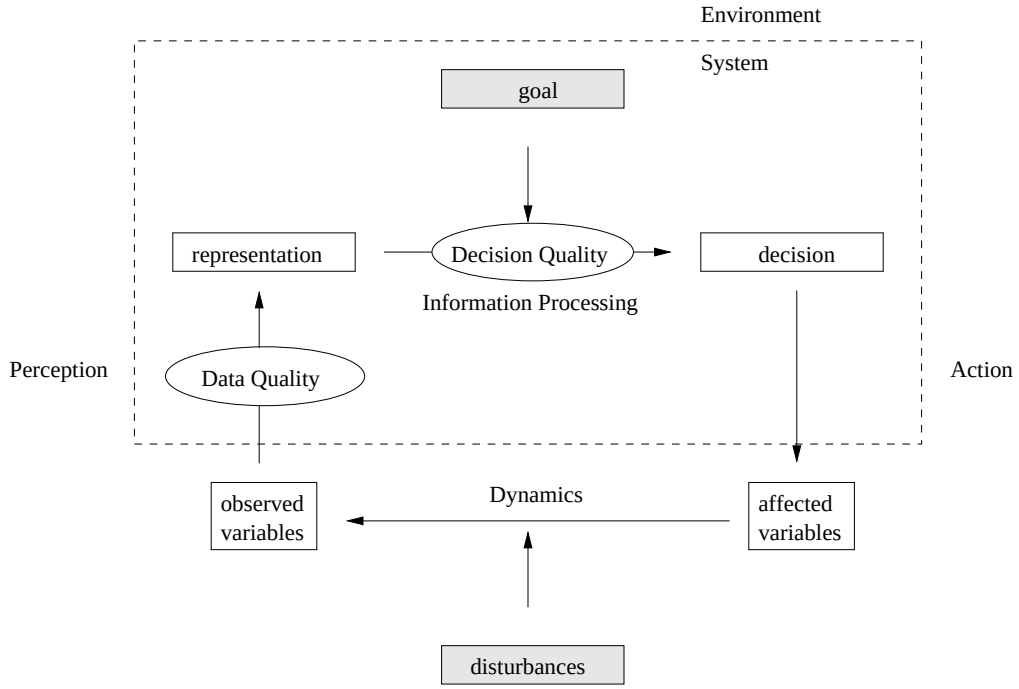


Figure 2.1. Classic feedback loop of a control system based on diagram from [33]

passive mechanisms to process the internal representation of the data to improve the decision quality. Part of that process involves evaluating the current system state, as manifested in the internal representation, in relation to its goal and determining the best change to make in order to preserve that goal. The system then makes a decision to take an action, if necessary, to counteract deviations from its goal. This is either a feedforward, proactive based on non-essential variables, or a feedback, reactive based on essential variables [33]. The taken action affects the variables that describe the environment including the essential variables that are monitored by the system. These variables are also perturbed by the unknown processes that we previously characterized as disturbances. Both actions and disturbances manifests themselves as changes in the observed variables, which will again be sampled by the sensors and the system can continually evaluate its effectiveness at maintaining its goal. This process creates an extremum control loop which attempts to achieve stability in the system while preserving its goal.

2.2 Adaptation Mechanisms in Overlay Networks

An adaptation mechanism allows a network protocol or overlay network to adjust to network environmental conditions with the goal of improving its performance and availability. The goal of these systems is to optimize performance or maintain at least a minimum level of service which characterizes the preferred state analogous with Figure 2.1. This preferred state is typically defined by application-centric metrics (bandwidth, latency, round trip time, jitter, loss-rate, etc) and is heavily dependent on the applications the overlays or protocols are intended to support. As a result, these metrics are typically perceived through passive observation or active probing. A number of mechanisms are employed in order to improve the quality of these measurements. We characterize these mechanisms as *data quality* mechanisms. Information processing relates to evaluating if the performance has degraded below the preferred state or not and deciding if there is a valid change that will help it maintain its preferred state based on some decision criteria. Finally, an action relates to either making a change if the performance has suffered and there is a suitable alternative or taking no action at that point. The affected variables are intended to be those that will help the system return to its preferred state. In these systems, the disturbances have typically focused on the churn of members or the transients of the Internet. Below we overview the main techniques that are used to augment adaptivity and improve resiliency in overlay networks, in two areas, data quality and decision quality.

2.2.1 Data Quality

A critical factor for the effectiveness of an adaptivity mechanism is the quality of the data observation and estimation, as well as the ability of the metrics used to accurately reflect the state of the network environment. An additional factor can be scaling down the data during processing in order to decrease the associated overhead. Examples of factors that influence data quality are data freshness, variability and

the presence of noise. Mechanisms related to these issues are data sampling, data smoothing, metric construction, and data summarization and aggregation.

Data sampling Data sampling is a method used to prevent staleness of information and subsequent oscillations. This sampling is typically manifested as a passive sampling window or active probing. It is desirable to have a high frequency of sampling to decrease mispredictions, as it was shown in the design of RON [30]. However, frequent sampling can introduce a significant overhead both in the system and the network environment.

Data smoothing Measured variables often exhibit a large amount of variance or errors, especially in discrete measurements of continuous valued functions. Data smoothing is a technique used to reduce and eliminate the noise and variability in the samples of physical state variables. The method is also often used to reduce the effects of erratic changes in the measured values and results in effective filtering. For example, in [35] a smoothed Round Trip Time (RTT) is used as the metric for the cost of the overlay link to prevent wild oscillations in measurements.

Metric construction Another technique commonly used to address the instabilities in measured data is metric construction. New metrics are constructed from existing metrics to improve the estimation process. These methods can be used to elucidate relationships between metrics useful for estimation that may not have been discernible when considering the metrics in isolation. An excellent example of the importance of metric construction can be found in [36], where a new metric is constructed by combining the latency as perceived by a neighbor node with the RTT to the neighbor. The newly constructed metric is then coupled with the bandwidth measurement. The authors demonstrate a substantial improvement over the approaches that considered both latency and bandwidth in isolation.

Summarization and aggregation Summarization and aggregation are two techniques that are employed to provide high scalability and improve stability. These techniques are used by the Border Gateway Protocol (BGP) [37], allowing it to scale by limiting the amount of information that is considered by the large number of routers. The obtained stability is at the expense of being less aggressive in the path exploration and maintenance, which sometimes inhibits BGP's ability to recover from faults [38].

2.2.2 Decision Quality

The response taken by an adaptivity mechanism has an associated cost that must be weighed against the degree of benefit that could occur as the result of the adaptation. Under some network conditions such decisions could lead to instabilities [39–41], such as oscillatory behavior commonly referred to as *flapping*, where nodes rapidly switch between seemingly equal alternatives. New techniques were deployed to mitigate these phenomena and provide a tradeoff between responsiveness to changes and instabilities. Examples of such techniques are damping and hysteresis. Below we provide more details about techniques related to the decision process.

Utility discretization Utility measures are thresholded relative measurements used to quantify the potential usefulness of making a particular change. They are usually tied to the optimization strategy employed by the estimation algorithm used when taking a response. Because often times the numeric metrics that drive these utility functions have large variances and frequent updates, discretization is used to reduce the values of continued metrics by dividing the range of metric values into intervals. Utility discretization is used when the change results in substantial improvements over a threshold. For example, in [41] a routing change is only made from the current path when the improvement in bandwidth is above some threshold. Other examples include RON, where routes are changed to obtain at least a

50% improvement in throughput [38], and GoCast [42], which avoids "futile minor adaptations" by requiring new neighbors also have a 50% improvement in RTT.

Randomization A common technique used to address deterministic and symmetric characteristics of distributed systems and network protocols is randomization. The technique is used to offer another dimension of variability to reduce the predictable behavior of network protocols. For example, randomization has been used to reduce the likelihood of a "thundering herd" effect where many nodes attempt to switch to the same link [43]. Another example is using randomization for path selection [41] to reduce the loss rates.

Damping Frequent changes due to ephemeral failures can cause significant instability in systems. Damping was proposed as a stabilizing technique to reduce unwanted or excessive responses to network conditions and limit the propagation of unstable information. Damping creates a temporal diminution of metric qualities characteristic for generating oscillatory behavior. The benefits of the technique were demonstrated in the context of BGP. For example, damping has been used within BGP to suppress route changes caused by link flapping by distinguishing consistently unstable routes from routes that experience ephemeral failures [40]. Another example can be found in [41], where it is demonstrated that increasing the amount of time between route change decisions can improve the accuracy of measurements and inhibit the frequency of change at the expense of reactivity.

Hysteresis Hysteresis is a technique commonly used in protocols to add a slowing effect to changes. This is achieved by adding a history dependence to the system where previous readings can influence the estimation effects of subsequent readings. Many protocols deploy different types of mechanisms to introduce hysteresis to the system. For example, RON demonstrated that smoothing was not enough to avoid flapping and thus it used smoothing hysteresis to avoid flapping between measurably equal routes [38]. Hysteresis was introduced by giving a bonus to the "last good"

route. Hysteresis is coupled with the aforementioned randomization in Tapestry to further reduce the likelihood of a "thundering herd" effect [43] .

2.3 Summary

We provided an overview of the main adaptation mechanisms used in overlay networks related to the two main components of any adaptation mechanisms: data quality and decision quality. A brief analysis of the above techniques indicates that the data quality phase depends on the accurate interpretation of performance observations, as well as the correctness of the responses received from probed nodes. Decision quality depends on the ability to accurately evaluate the current state of the system and evaluate action alternatives. Both these mechanisms can be influenced by malicious or selfish attackers that do not always behave according to the protocol. In the next chapter, we explore such adversarial models and their impact on the adaptation mechanisms used in overlay networks.

3 ATTACKS AGAINST ADAPTATION MECHANISMS IN UNSTRUCTURED PERFORMANCE-DRIVEN OVERLAY NETWORKS

Adaptive measurement-based overlay networks optimize performance or maintain at least a minimum level of service by addressing disturbances existent in the network and benign failures. Such networks are not designed to overcome malicious attacks performed by adversaries that compromise overlay nodes and interfere not only with the system services, but also with the adaptation mechanisms themselves. As the benefits of adaptive measurement-based protocols are evident, what is needed is to enhance these mechanisms with resilience to Byzantine attacks.

In this chapter we describe and analyze attacks against the aggressive self-organization protocols associated with optimized multicast overlay networks. We focus on attacks against adaptive dissemination, a system component that allows a system to change the paths by which data is disseminated in response to changes in conditions that perturb its pre-subscribed goals. The attacks leverage the fact that adaptive dissemination must make two very difficult decisions to achieve its objectives:

- When should the overlay dissemination structure change?
- If a change should be made, how should the overlay change?

We first describe an abstract system model, then our adversary model, and finally present the attacks, which we refer to as *attraction*, *repulsion*, and *disruption*.

3.1 System Model

In this section we provide an abstract system model for performance-driven measurement-based unstructured overlay networks. Such networks provide support

for single-source broadcasting applications, that are high-bandwidth (hundreds of kilobits per second or more), and real-time but not interactive. In these applications, the source is continually available, and failure of the source results in the failure of the service. Direct communication may exist between every source and receiver because there exists an IP path between them. However, it is desirable to keep the overhead at the source low, and hence use of this communication mechanism is limited.

The system consists of a set of participating nodes that communicate with unicast channels using an underlying physical network substrate. The nodes have similar functionality with the exception of the source of the data. While each node may be a recipient of that data, it is also responsible for contributing to the routing infrastructure. The overlay construction is completely self-organized and distributed. No node has complete knowledge of the dissemination topology.

The goal of the system is to optimally disseminate large amounts of information, either continuously or over substantial periods of time, simultaneously to a large subset of nodes. The specifics of the optimality goals are often imposed based on the application using the overlay. The state of a node in the system is characterized by a vector of k application dictated variables, $V =< v_1, v_2, ...v_k >$ - referred to as essential variables, that relate to the dynamics of the network environment. These variables are continuously measured during a nodes participation in the overlay network. They are not independent. As disturbances arise, the system continues to evolve by converging to equilibrium points, which relate to a state in the dynamical system where it stops changing since it is able to support the goals of the application. These equilibrium points are typically called attractors [33,44], since the system state is drawn to them. Attractors may be different for each node depending on their local network environment. The system will typically stay at the attractor until it becomes unstable which occurs when the system state is no longer suitable for the current network environment. Then the node evaluates the current set of potential attractors which offer the promise of driving the system back to equilibrium.

In order to support this functionality, each node maintains a neighbor set of size s, a routing table of size r and the upstream member of the overlay responsible for sending the data (parent). The nodes in the neighbor list are continually updated via a membership dissemination protocol to reflect a current subset of nodes that are participating and reachable in the overlay. There are no constraints on the members of a nodes in a neighbor set. Periodically the nodes in the neighbor set are evaluated using passive observation and active probes to evaluate their utility and attractiveness as a new upstream router. The neighbor set represents only partial information, no node has complete knowledge of the dissemination topology only information that can be measured or probed locally. The routing table represents a subset of nodes that the node is responsible for routing data to (children). The size of this subset, out-degree, is limited by a system characteristic referred to as saturation degree S. The saturation degree represents the number of concurrent data streams the node is able to support before saturating its underlying physical network link.

3.2 Attacker Model

This thesis studies attacks that attempt to subvert message delivery in unstructured measurement-based multicast overlay networks using a constrained-collusion Byzantine failure model similar to that proposed in [17]. In this model, a set of N mutually distrusting distributed nodes form an overlay using an unstructured measurement-based multicast overlay protocol. We assume a malicious adversary has access to the same data as any legitimate user, including the cryptographic keys stored on compromised nodes. The percentage of malicious nodes within the overlay is bounded by f $(0 \leq \mathrm{f} < 1)$, where the set of malicious nodes can be made up of disjoint independent colluding coalitions or a single coalition. The nodes in these subset coalitions are either malicious from the moment they join the overlay, having bypassed the authentication mechanism, or they are compromised at some point after

the overlay has been realized. In this model we also assume intra-coalition cooperation between members of a coalition but not inter-coalition cooperation between coalitions, since if they are cooperating we will assume it is the same coalition.

The adversaries target the routing topology and parent selection process that are taken by non-malicious nodes by storing, inserting, dropping, modifying, and/or misrouting either data messages or routing information [17] within both the routing and overlay-layers. In addition, the malicious nodes can interfere with the adaptation mechanisms themselves showing an arbitrary behavior. The malicious nodes are colluding to maximize the the amount of influence the conspiring malicious nodes can have on the topology of the overlay network. Below we provide more details about the attacks.

3.3 Attack Descriptions

The attacks performed by the adversarial coalitions under this failure model exploit the fact that as performance-driven overlay adapts its dissemination topology, the overlay must make goal oriented optimization decisions that are extremely difficult in a decentralized self-organizing system. We classify the attacks into three main categories: enticing the system to take an action that it would not normally make, persuading the system not to take an action it would make under normal conditions, and finally confusing the system so that it keeps taking incorrect actions.

In the context of multicast overlays, the main asset is the broadcast data. Thus, one of the main goals of an insider adversary is to control as much of the data disseminated in the overlay as possible. This can be achieved if the adversary, having infiltrated the overlay, manages to manipulate the path selection or the multicast structure maintenance to its advantage. Based on their effect on the control of path selection, we classify these attacks as *attraction attacks*, *repulsion attacks*, and *disruption attacks*. Any of these attacks can be conducted by an adversary by affecting the observed and collected metrics used by the adaptation mechanisms as follows:

- the malicious nodes lie about the observation space; In this case, the adversary manipulates the secondary sources of information used by the adaptation mechanism.

- the malicious nodes impose an artificial influence toward the observation space; In this case, the adversary manipulates the primary sources of information measured by a node.

3.3.1 Attraction Attacks

The first type of attack relates to deceptively increasing the likelihood that the system makes a decision that would not allow it to maintain its presubscribed goals. This class of attacks is similar to the Eclipse attacks in structured overlay networks [18, 19]. Such attacks are a form of "bait-and-switch", where observed data is manipulated by a malicious node in order to draw attention. In such attacks, the malicious nodes are always presenting the network conditions to be better than they are, with the goal of gaining control over significant traffic. The attack can also target one particular node, in which case the attacker will persuade the victim to attach to a malicious parent in the dissemination structure. For example, if the dissemination structure is a tree, the goals of the attacker can be to attract many nodes to itself as children or to obtain a higher position in the tree. The final goal of the attack can be manipulating data, performing traffic analysis, man-in-the-middle attacks, causing disruption for specific nodes by isolating them, or selectively dropping packets for a particular destination.

A basic way to perform the attack is for a malicious node to falsify the answers to probe requests from correct nodes to deceptively create the perception of a route with higher utility from the perspective of the victim node. The attacker can exploit characteristics, such as low frequency in the data sampling, to prolong the effects of his malicious action on the data smoothing mechanism. As a result, the utility function will create an incorrect adaption since the utility gain does not reflect reality.

For example, if the utility function is based on the bandwidth from the source, a malicious node can attract other nodes in the tree by lying about its bandwidth from the source every time it is probed. The utility function will incorrectly choose to adapt and choose the malicious node since it appears that the change will guarantee a better bandwidth from the source.

As opposed to lying about the environment, attackers can also perform this attack by artificially manipulating the measurements of the current environment. This would relate to an attack against the directly observable primary sources of information that are used to evaluate link state. An example of this type of attack in structured overlay was described in [17]. In this example, when a non-malicious node sends a probe to a malicious node, a conspiring node closest to the non-malicious node responds. As a result the conspiring nodes can make the malicious nodes appear closer (more attractive) and increase the probability of being selected during adaptation. Thus the attackers are manipulating the observation space of the victims in order to increase the likelihood of them gaining augmented control.

3.3.2 Repulsion Attacks

The second form of attack relates to reducing the likelihood of the system making a change that it would normally make under non-malicious conditions as a goal oriented optimization or towards maintaining a steady state. This relates to deceptively making a particular decision unattractive or less attractive.

These attacks in adaptive routing seek to reduce the attractiveness of other nodes or misrepresent the sufficiency of their own abilities. This is accomplished by means of lying and defamation in responses to active probes or by manipulating the physical or logical infrastructure in a way that creates the perception of routes with lower utility. As in the case of attraction attacks, repulsion attacks can target one particular node. The ultimate purpose of such attacks is to offer the malicious node

opportunities for free-loading, traffic pattern manipulation, augmenting attraction attacks, or to just cause disruption.

One way a malicious node can conduct the attack is by lying about its performance. An example of such an attack is when a malicious node lies about route costs (i.e., hop count) in order to convince other nodes that it has a bad connection and thus should not be selected as a parent. The malicious node will then obtain a reduced burden. A particular attack that falls into this category was analyzed through simulation in [23]. The authors showed that selfish nodes (i.e., nodes that want to obtain advantage over other nodes but do not have destructive goals as malicious nodes) can selfishly improve their performance by manipulating distance measurements.

An attacker may choose to manipulate the existing environment, rather than lie about it, by exerting an influence of aversion toward the partially observable link state estimation. This constitutes an attack against primary sources of information which are metrics that are directly observed by the node. One example of this attack was demonstrated in [30], where a flooding attack was conducted to demonstrate how quickly the overlay could respond. However, in this case, the attack was executed by an external attacker. An example of an internal repulsion attack was presented in [23]. In this case, a selfish node delayed its probe responses to affect the RTT measured by the node performing the probing. In both cases, the attackers manipulate the observation space of the victim to make things seem worse than they actually are.

3.3.3 Disruption Attacks

The third form of attack involves attempting to continuously confuse the system so that it keeps making incorrect decisions and is unable to find a steady state (equilibrium point) or maintain its apriori described goals. Disruption attacks target the availability of the network by using the adaptivity mechanisms to turn the system

against itself. An attacker can create significant disruption in the overlay by injecting or influencing the observation space metric data to generate self-destructive responses as a result of unnecessary adaptations. The ultimate goal of such attacks is to affect the infrastructure that supports the overlay with the intent to prevent or degrade service. These attacks can be classified as a form of denial of service (DOS) and can result in jitter, flapping, or partitioning the overlay.

An example from this class of attacks is the attack against TCP [9], which is an attempt to deny bandwidth to TCP flows by manipulating the observation space to create the perception of network congestion. The magnitude of the disruption is generally based on obtaining a steady-state [39] and a temporal measure of how long it takes to reach this state. The work in [10] generalizes the work presented in [9], as a form of low-rate reduction of quality (ROQ) attack that focuses on attacking adaptive control loops that drive resource allocation and affect the perceived service of a system.

3.4 Summary

In this chapter we presented an abstract system model for performance-driven measurement-based unstructured overlay networks. We also described the constrained-collusion attacker model that we will assume throughout the rest of the document. Finally we classified and described several attacks that exploit the data quality aspect of adaptivity mechanisms. Next, we examine the practicality of these attacks in a real-world network deployment. In addition, we study the degree to which mechanisms used within the adaptation decision are able to detect and recover from the attacks we identified.

4 CASE STUDY: END SYSTEM MULTICAST

In the previous chapter we described attacks against the aggressive self-organization protocols associated with optimized multicast overlay networks. In this chapter we demonstrate through experimental results how the attacks identified in Chapter 3 can be used against the ESM overlay system. We first provide an overview of ESM and the adaptation mechanisms it uses. We selected ESM to demonstrate the attacks because of its maturity, extensive deployment, and particularly because of the advanced set of techniques it uses for augmenting its adaptivity. We then describe in detail the attacks we experimentally performed against ESM. Our experiments show that even though ESM employs almost all of the mechanisms discussed in Section 2.2, it is unable to mitigate the attacks posed by a malicious insider adversary.

4.1 Overview of ESM

ESM [24] is an unstructured measurement-based overlay multicast system that shifts the multicast infrastructure to end systems and forms a peer-to-peer overlay network. The system is largely used for broadcasting live events including academic conferences such as SIGCOMM and INFOCOM. ESM uses an application level multicast protocol that builds an overlay tree for distributing multicast content, as seen in Figure 4.1.

An important component of ESM is *group management*, which ensures that the overlay remains connected in the face of dynamic group membership and failure of members. There are two main mechanisms that contribute to the group management functionality. The first mechanism enables a node to join the overlay multicast system. A node that wants to join the overlay first contacts the source that provides it with a set of nodes from overlay (about 30). Each of the nodes in the set can

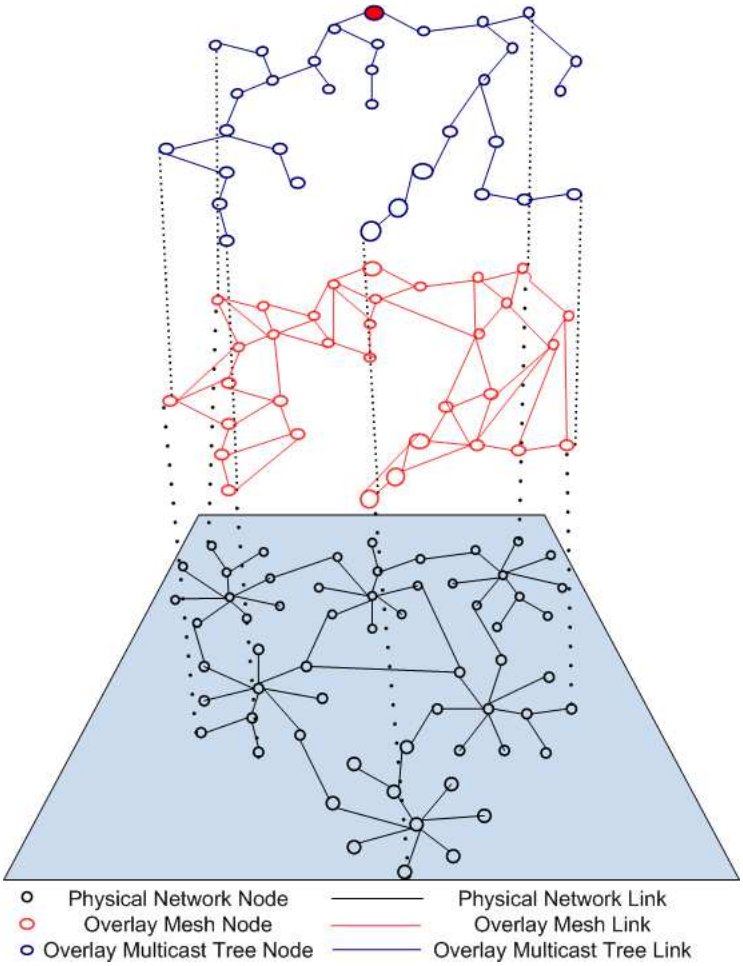


Figure 4.1. Illustration of the physical network and the two logical networks: the overlay network and the multicast tree.

serve as a potential joining point in the overlay. The joining node will then select one of those as a parent and thus become part of the overlay. The second group management mechanism enables nodes to maintain partial membership information about other members present in the system, even though nodes may join or leave the system. This is achieved by using a gossip-like mechanisms that periodically distributes the list of neighbors maintained by each node.

Another key component of ESM is the *adaptation mechanism* used to dynamically change the multicast tree to improve application performance or maintain it

when network conditions change. More specifically, this adaptivity serves to improve suboptimal overlay meshes that can occur as a result of random initial neighbor selection, aggressive partition repair, multicast group membership changes, and the transient nature of underlying physical network conditions.

ESM employs several techniques from the set we identified in Section 2.2 to augment adaptation. Data sampling and data smoothing are used to address variations in the metrics considered: available bandwidth, latency, RTT and loss rate. Both passive observation and probing are used in collecting the data used to make the adaptation decision. ESM also employs a number of combined metrics, damping, randomization, hysteresis and several different utility functions to address instabilities in the observed data. Below we provide more details about how these techniques are supported in ESM. The values of the parameters we discuss below were selected empirically by the ESM creators and were demonstrated to provide good performance for several deployments [45].

A dedicated subsystem in ESM, the *performance agent*, is responsible for collecting the metrics used in the adaptation scheme. This agent runs periodically (every 7 seconds) and sends probe requests to a random subset of neighboring nodes. Each probe response from a neighbor includes a set of variables $< b, l, s, p >$ consisting of the bandwidth it perceives from the source b, the latency l, the saturation level s, and the path p. The probing also enables each node to determine the RTT to a neighbor node, which is used to augment the other metrics. In addition, every node also maintains its own view of the network conditions by passively observing the data traffic it is receiving from the source. Both the passively observed metrics and the metrics measured via probing are smoothed using an exponential smoothing function to obtain a long-term average. Based on the types of responses or lack of responses the performance agent will prune the neighbor table. The neighbor table is augmented through the group management mechanisms described above.

The goal of the adaptation in this system is to choose an upstream router (parent) in the dissemination tree that will allow it to maintain its bandwidth and latency

requirements as specified by the supported application. This is achieved based on a utility function. Currently, ESM supports three utility functions: one based on bandwidth, one based on latency, and one based on a combination of bandwidth and latency. The bandwidth utility function constructs a metric by combining the data offered by the neighbor in the probe response with the measurements made passively. If the bandwidth currently being received by the node is equal to the source rate or the relative improvement is less than 10% of the current bandwidth, the utility value is set to 0. If a node's bandwidth is greater than the current bandwidth being received and has a greater discretized utility function than the current parent, then the utility value is augmented. The latency utility function is constructed by merging the link latency with the route latency. If the new latency is less than the current latency then the utility is set to the relative improvement over the current metric. On the other hand, if the link metric to the current parent is less than 5, then the utility is also augmented.

The decision whether to change the tree or not is made locally by each node based on the utility gain (computed as described above) and a damping factor, used to induce stability. First, the damping factor will determine if switching the parent will indeed happen. If a node has recently changed parents with high frequency, a change is less likely to occur. In addition, a randomization technique is also used to avoid the case where several nodes try to change to the same parent. Each node maintains a switch probability indicating that the parent change will indeed happen. A random number is selected before making the switch decision. If the random number is greater than the switch probability, no parent change takes place. Otherwise, the change to the newly selected parent will take place and the current parent is placed on a pending drop list, which is flushed after a period of time. Once the change happens, the switch probabilities are reduced if there have been many switches recently and the switch probability is above a threshold.

The process of selecting the parent to which a given node will switch works as follows. Each node maintains links to its parent in the tree and its neighbors in

the overlay. The agent running on each ESM node periodically processes the cached probe responses to build a selection table of possible candidates for becoming its new parent, referred to as the "short list." Nodes which are currently saturated, descendants of this node, and those that have recently failed a bandwidth test are not considered. If there is no utility gain, no node is selected as a potential candidate and the process will be repeated next cycle. If several nodes are possible candidates (attractors), then the first node on the list is selected as the new parent. The selection process also uses hysteresis to generate a negative bias against nodes that have performed poorly as experienced through primary observations. As a result, those peers who have been chosen as a parent and performed poorly are less likely to be chosen again.

4.2 Attacks Against ESM

In this section we concretely demonstrate examples of the attacks identified in Section 3.3 in the context of the ESM system through experiments performed in the DETER and PlanetLab testbeds. We first provide a brief overview of the testbeds and experiment setup, then we provide details about the attacks.

4.2.1 Testbed and Experiment Setup

We conducted our experiments on the PlanetLab [25] and DETER [26] testbeds. PlanetLab is the most well-known live Internet testbed. It consists of machines hosted by research institutions and allows for the deployment of overlay networks and distributed services. Currently, over 275 active research projects are using PlanetLab as their testing platform. We use PlanetLab in our experiments because it provides the opportunity to study ESM under real-world conditions.

In addition, for experiments that could be disruptive to PlanetLab, we used the DETER testbed. DETER is a shared testbed infrastructure that is specifically designed for cyber-security research. Unlike PlanetLab, DETER is an emulation

Table 4.1
Effect of one malicious (lying) node on an ESM deployment of 30 nodes on PlanetLab over a 90 minutes run.

Experiment	Chosen	Changes	Unique Children	Unique on Path	Ave(sec)
Lying	72	369	10	12	794.42
Not Lying	15	216	8	9	72.69

testbed that allows us to model real networks in terms of latency and network topology. In addition, DETER provides a stable controlled environment and repeatable experiments. This allowed us to isolate the issues we were investigating and overcome some of the uncontrollable variability that can be found on PlanetLab.

There are several parameters that characterize ESM deployments and our experiments. The most important are the number of nodes in the overlay, the degree saturation for the nodes, and the constant bit rate of the multicast source. In our experiments we selected values specific to the application characteristics we focus on. All the values we selected are similar with the ones used in previous ESM deployments [45]. We use overlays of 30 and 50 nodes and experiment durations of 30 and 90 minutes. Nodes usually join after the experiment begins and leave before it ends, with an average participation of 25 minutes and 85 minutes per node. We use a saturation degree of 4-6 nodes. In all the experiments below, we use a constant bit rate of 480 Kbps, which is sufficient to transmit video at two different qualities and one audio channel.

4.2.2 Attraction Attacks

All the experiments demonstrating attraction attacks were performed on the PlanetLab testbed.

Compromised nodes may use their insider position to lie about their bandwidth, latency, and saturation with the goal of attracting as many nodes as possible as chil-

dren in the multicast tree. To demonstrate the effect that one malicious node, who exploits the adaptive nature of ESM, has on the multicast tree construction, maintenance, and stability, we ran the following experiments. One randomly selected node lies every probing cycle about bandwidth, latency, and saturation. This experiment was performed using 30 nodes, a degree of 6, and a duration of 90 minutes. We summarize our findings in Table 4.1. When it is not lying, the malicious node is selected only 15 times as a parent by other nodes. When it is lying, the malicious node is selected 72 times, almost 5 times more often. When the node is lying, the overlay becomes more unstable, as can be seen in the large number of total parent changes. This increased instability can be attributed to the fact that the new child will eventually realize the bait-and-switch and change again. In addition, Table 4.1 shows that the lying node manages to get selected on paths such that the traffic of 12 other nodes (out of 30) goes through the malicious node.

We next investigate the degree to which a higher percentage of malicious nodes can affect the network. As in the case of one malicious node, the experiment was performed using an ESM deployment of 30 nodes with a saturation degree of 4 nodes and a duration of 90 minutes. The results of the experiment are summarized in Figure 4.2. The graph depicts the percentage of nodes that have at least a malicious node on their path to the source at some point during the experiment. In addition, it also shows how many of these nodes have chosen a malicious node as a parent. Finally, it shows how many of the decisions to change the parent were decisions that resulted in selecting a malicious node. Note that the malicious nodes were randomly selected and stronger attacks may exists by choosing nodes that have very good network connections.

As shown in Figure 4.2, a network in which 20% of nodes are malicious will result in those nodes controlling a significant amount of the traffic to other (non-malicious) nodes. For example, 20% of malicious nodes succeed in convincing more than 50% of the nodes in the network to select a malicious node as a parent. This potentially allows the adversary to monitor all traffic for the nodes it tricked into selecting it as

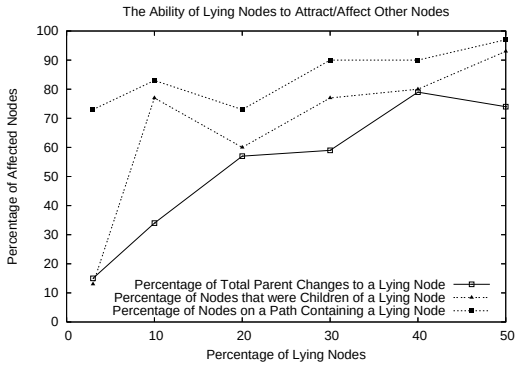


Figure 4.2. This graph demonstrates the effect of attraction attacks on the correct nodes as a function of the percentage of malicious nodes. The experiment was conducted on PlanetLab with an ESM overlay of 30 nodes, for a duration of 90 minutes.

a parent. In the case where 30% of the nodes are malicious, 90% of the nodes in the network have at least a malicious node on their path to the source. In this case, the malicious node can potentially affect any node that is positioned lower in the tree.

Having malicious parents can result in a severe degradation of service if the malicious parent decides to selectively drop data. One interesting aspect is to examine if the stability techniques and the decision function are able to detect the bad adaptations. In Figure 4.3, we demonstrate the potential impact of malicious nodes that use their positions on the tree to drop data traffic. The graphs plot the bandwidth averaged over all receivers as a function of time. The experiments are performed using an ESM deployment of 50 nodes with a saturation degree of 4 nodes. The duration of each experiment is 30 minutes, which corresponds to the duration of a conference presentation. At a predetermined time during each experiment (about 12 minutes since they joined the overlay), malicious nodes begin to drop 100% of the data traffic that they receive through the data dissemination tree. We vary the percentage of malicious nodes among 10%, 30%, and 50% of the total receivers to demonstrate the performance degradation that results from introducing malicious faults into the

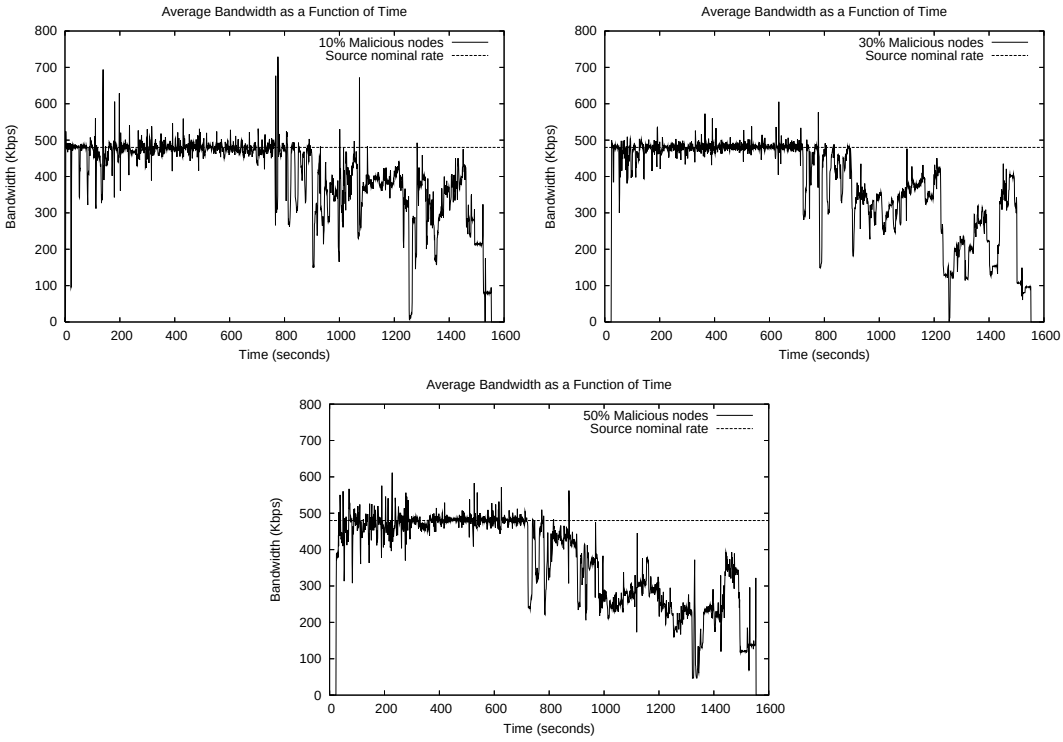


Figure 4.3. These graphs demonstrate the effect on average bandwidth in ESM as a function of the number of malicious nodes. The experiment was conducted on PlanetLab with an overlay of 50 nodes.

system. In each experiment, the malicious nodes are chosen at random from the receiver set and there is no colluding communication between the malicious nodes.

In the case when 10% of the nodes behave maliciously about 10 minutes into the experiment, the average bandwidth depicted in the graph is shown to start decreasing. For the remainder of this experiment, the average bandwidth remains below the source bit rate of 480 Kbps. A similar effect is observed in the cases where 30% or 50% of the nodes are malicious. As expected, increasing the percentage of malicious nodes has a greater effect upon the system's performance. However, it can be noted that the effect of 10% nodes is already significant and increasing the number of malicious nodes to 30% and 50% does not change the effect on the average bandwidth dramatically. We believe this is because the tree structure is very vulnerable since it does not have any redundant paths and 10% of malicious

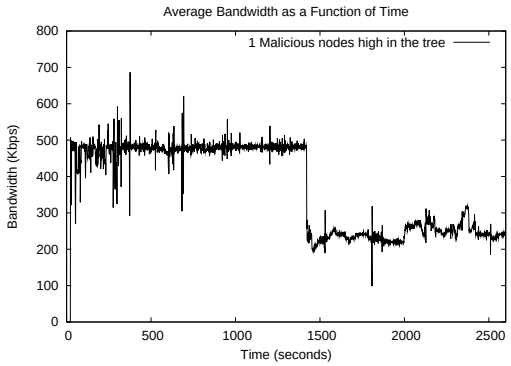
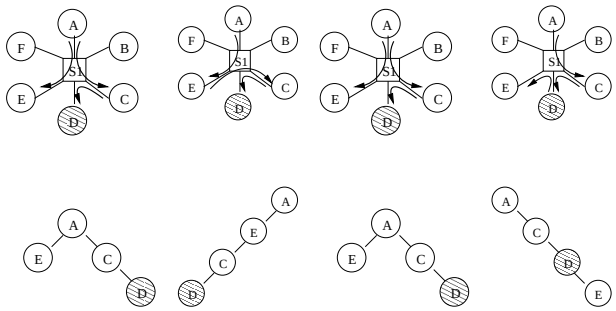


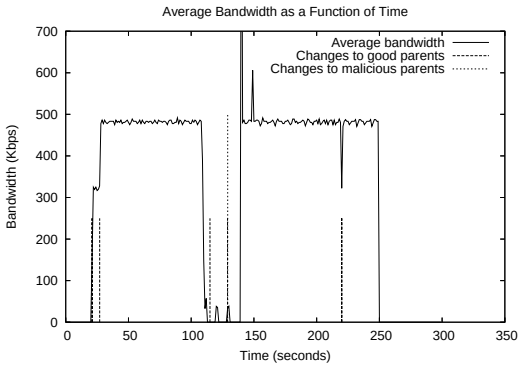
Figure 4.4. This graph demonstrates the effect on average bandwidth in ESM of one malicious node that manages to maintain a high position in the tree while dropping 100% of the traffic, starting at 1400 seconds. The experiment was conducted on PlanetLab with an overlay of 50 nodes.

nodes are enough to obtain advantageous positions in the tree. Note that in our experiment the nodes lied to obtain the advantageous position. The communication overhead associated with the attack is basically zero. As it can be seen in Figure 4.3 the adaptivity mechanisms react slowly and they do not manage to avoid selecting the malicious nodes from the tree structure by the end of the broadcast.

The closer a malicious node is to the source, the more nodes there are that use the malicious node in their path to the source. Thus, compromising nodes connected directly or near the source makes this attack more devastating. In Figure 4.4, we demonstrate the effect of only one malicious node that lies during each probe cycle and manages to maintain its position in the tree near the source. The experiment was conducted using 50 nodes, a saturation degree of 4, and a duration of 90 minutes. As it can be seen, the effect is devastating as soon as the node starts dropping traffic. In spite of this, the adaptivity mechanisms are not able to react for more than 15 minutes.



(a) (b) (c) (d)



(e) The effects of the attack on the average bandwidth

Figure 4.5. An example demonstrating a repulsion attack against the ESM multicast overlay system in a controlled experiment on DETER. (a) shows the overlay and the multicast tree before attack, (b), (c) and (d) show topology changes in the multicast tree as a result of the attack, at times 115 seconds, 128 seconds and 129 seconds from the beginning of the experiments, respectively. The result is that node E is manipulated by the attacker to attach to malicious node D, although this makes E to be three hops away from the source, instead of one at the beginning of the attack.

4.2.3 Repulsion Attacks

The experiments demonstrating repulsion attacks were performed in the stable, controlled environment provided by the DETER testbed.

We demonstrate a repulsion attack performed by exerting an artificial influence of aversion toward the partially observable link state estimation in order to manipulate the routing tree topology. This attack example was also motivated by the fact that, while performing attraction attacks, we noticed certain nodes that were directly attached to the source, generally powerful positions within the tree, that could not be enticed by the lying node. Thus, we wanted to analyze how difficult it would be to actually displace these nodes by providing an external influence on their observation spaces.

In Figure 4.5, we show a simple topology to emphasize the susceptibility of the ESM protocol to repulsion attacks. We use a star topology composed of six nodes, all of which are connected with 100 Mbps links to the switch, s_1. On the Internet, s_1 would relate to any focal point for a network link we are attempting to manipulate. There is also no background traffic so each node has the potential to receive full bandwidth. The ESM protocol is configured to use a saturation degree of 2 and the utility function used takes into account both bandwidth and latency. In our example, node A is the source, nodes C, D, and E are end-systems in the overlay, nodes B and F are outsiders who collude with D, a malicious node that has infiltrated the overlay. During the attack, nodes B and F generate traffic to augment the attack of malicious node D, which lies about its bandwidth, latency, and saturation. Similar results of the attack will be obtained if nodes B and F are trusted members of the overlay attempting to improve their position in the tree or influence the path the data takes from the source to themselves or others.

As shown in the graph in Figure 4.5(e), the overlay converges to a stable structure, shown in Figure 4.5 (a), after about 30 seconds, at which point the mean bandwidth is approximately the same as the source rate (480 Kbps). Topology changes prior to this point were due to nodes attaching to the overlay tree, as represented by the impulses in Figure 4.5(e). The attack begins at 115 seconds when nodes B and F begin flooding 30 seconds worth of traffic at the source, node A. After several seconds of traffic, the attack is able to generate the first disturbance in the tree when node

C detaches from the source and chooses node E as its new parent 4.5 (b). This occurs despite the fact that C now has an extra hop to the source. Then, 14 seconds later, C switches back to its previous position, but the overlay has yet to stabilize (Figure 4.5 (c)). Next, under a second later, node E detaches from the source and, instead of choosing node C, chooses the malicious node, D, as its parent (Figure 4.5 (d)). Note that node E was previously directly connected to the source but is now connected three hops away. The changes after 200 seconds are due to nodes leaving the experiment.

One important aspect of the experiment is the amount of traffic generated by the attackers. Two nodes create the attack by filling the 100 Mbps link with a 30 second burst of traffic and in some experiments it only required a 5 second burst. Note that in real Internet deployments, the cost of the attack will be substantially less since links will typically have a lower bandwidth.

A variant of the attack is to target the active probes on which the victim node relies. In this case, the victim's peers will be made to look unappealing for changes, thereby increasing the chances of the malicious node to move upward in the tree.

4.2.4 Disruption Attacks

In our experiments we have also been able to demonstrate a number of disruption attacks that could be performed by an attacker against the ESM protocol. Disruption attacks are generally composed of two interesting aspects that differentiate them from the other two forms of attacks we previously discussed. The first aspect relates to putting the feedforward and the feedback mechanisms at odds with each other. This creates a situation where the system keeps overcompensating for bad decisions and becomes unstable. The second aspect relates to persistency of the attack. Since the negative feedback used for adaptation will eventually drive the system to stability, disruption attacks require nurturing by the attacker.

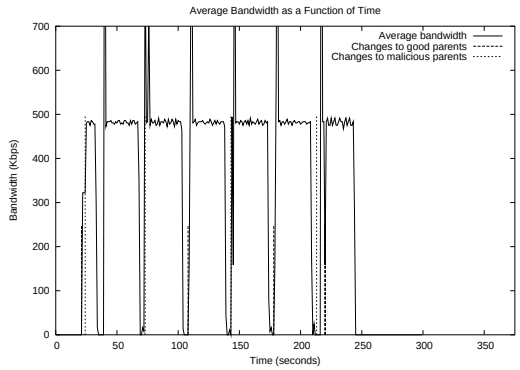


Figure 4.6. An example demonstrating a disruption attack against the ESM multicast overlay system in a controlled experiment on DETER. The experiment was performed using the same experimental setup as 4.5. In this example, the attackers periodically sent 5 second bursts of traffic at the focal point creating constant churn in the system as it tried to stabilize. Changes to good parents: 21.04(2), 108.07, 178.09, 220.03(2). Changes to bad parents: 24.08, 73.06, 143.08, 213.10.

Figure 4.6 demonstrates an example of a disruption attack where the attackers once again exerts an artificial influence, traffic, towards a focal point of the overlay topology. The main difference from previous attacks is that the artificial influence is done periodically in order to de-stabilize the infrastructure. In the experiment in Figure 4.6, the attackers sent 5 second bursts of traffic every 30 seconds. This is similar to the attacks performed in [9,10] which targeted the TCP congestion control. In Figure 4.6, we can see that using this technique the attacker can keep the system in a constant churn as it keeps trying to restabilize itself. Despite the fact that the attack was using only 5 second bursts of traffic, the attacker was able to drive the system to change parents at almost every cycle.

Another example of a disruption attack is summarized in the results of Table 4.1. In this attack, by continually maliciously advertising better performance then they could actually support, the attackers succeeded in destabilizing the system. As

a result, the system experienced a dramatic increase in parent changes from 216 to 369 during the experiment duration.

4.3 Summary

In this chapter we presented end system multicast (ESM) as a case study of an unstructured measurement-based overlay network. This included an overview of the ESM system and how it relates to our abstract system model. We also enumerated the advanced set of mechanism used to improve the robustness of ESMs adaptation. Finally, through experimental results on a real deployment of ESM we demonstrated how despite these advanced steps it is not protected from a malicious adversary.

5 SOLUTION SPACE: DESIGN AND IMPLEMENTATION

In this chapter we propose a solution framework consisting of several components that are necessary for providing a comprehensive solution for mitigating Byzantine attacks that exploit adaptivity in overlay networks. These components are: reducing unnecessary and unnatural adaptations, increasing stability by incorporating metrics that reflect stability into the decision process, detecting and adapting to malicious behavior, and responding to malicious nodes. In addition, we provide an in-depth description for what we believe is a critical aspect of the solution that has not been addressed adequately in previous research: preventing unnecessary or bad adaptations. As the attacks we are concerned with are performed by compromised nodes controlled by adversaries, cryptographic mechanisms deployed to provide authentication and encryption will not be able to mitigate the attacks by themselves. The solution space components we describe below are complementary to cryptographic techniques and assume that authentication and integrity data protection are provided.

5.1 Solution Components

The primary cause of the attacks we identified is the ability of the attacker to influence the adaptation process. This includes attacking both the feedback and feedforward mechanisms that attempt to regulate or control the system to maintain the systems goal. Thus, the most important component of a solution is preventing, or at least constraining the likelihood of, these unnecessary or bad adaptations. A perfect prevention will render the other components useless. However, perfect prevention is not possible since the attacker may create results that are not necessarily

observable making perfect prevention impossible. We identify four complementary components that the solution must have in order to address these attacks.

1. *Reducing Unnecessary or Bad Adaptations*

   The adaptation process relies on two sources of information one passively observed by each node and the other obtained by probing a set of peer nodes. By blindly accepting information and lacking contextual awareness, correct nodes have a propensity for making bad adaptations. This relates to the more general problem found in many network protocols associated with "blind acceptance" of routing metrics [46]. Thus, the system must possess a mechanism to filter out the metrics reported by malicious nodes that may adversely influence the adaptation decision. Components that address this problem are useful for mitigating attraction, repulsion, and disruption attacks.

2. *Increasing Stability*

   Reducing the number of unnecessary adaptations has the potential to increase the stability. However, we believe that explicit efforts must also be taken to reduce the overhead and the number of unnecessary changes by integrating stability in the function that drives the adaptation decision. Early research in adaptive control emphasized the importance of improving the stability of the system [34]. This is even more important in a malicious open environment where reducing the number of adaptations is obviously important since with every adaptation there is always the probability of making a bad one. This is an area that has received very little attention in performance-driven overlays.

3. *Detecting and Recovering from Malicious Attacks*

   Enhancing the adaptation decision with Byzantine robustness is not sufficient since no method will have perfect accuracy. The two mechanisms described above will still result in few bad adaptations. The number of bad adaptations is lower bounded by the probability of a node of a random subset being chosen

from the random neighbor set. The purpose of the first two mechanism is to reduce the ability of an attacker to artificially augment this probability to manipulate the overlay. Thus, the solution space must include a component that allows a node to quickly detect and recover when such an adaptation occurs. In this case, if the attack has observable effects such as severe degradation of the delivery service, the system must possess recovery mechanisms that allow a node to determine that it is connected to a malicious parent and then to find a new parent.

4. *Responding to Malicious Nodes*

   Another important aspect of any defense mechanism relates to responding to malicious nodes. The previous component addresses a node taking an immediate reaction to a malicious parent. We believe a solution must also take action and neutralize the threat of malicious nodes on the overlay system. Without taking such actions the convergence of the protocol, as well as the overall overhead, will increase as malicious nodes continue to interfere with the system. Overlay networks have often been designed to deal with node failures but an open problem remains the fact that overlay networks typically do not have a mechanism to remove nodes that are misbehaving [11, 16].

5.2 Reducing Unnecessary and Bad Adaptations

We now present an in-depth description for the most critical component of a comprehensive solution: reducing the potential for poor adaptation decisions in overlays influenced by attackers. In this section we begin by classifying techniques that have been used in structured overlays for dealing with false information and discuss their usefulness in unstructured overlays for reducing bad adaptations. This includes a discussion of the usefulness of exploiting system invariants and topology-awareness. We then introduce a new technique that evaluates temporal and spatial correlations among data in the system to detect outliers and reduce the ability of malicious

nodes to influence the adaptive responses. We describe how this technique provides protections not afforded by the other mechanisms.

5.2.1 Invariant Relationships

Invariant relationships define specifications between aspects of a system that hold while the system is operating in a correct state. These relationships are used to maintain the semantic integrity of a system [47]. Checking such invariants has been successfully used for evaluating the integrity of systems [47].

Identifying and checking system or network invariants has been proposed also to detect attacks against stuctured overlays [16–19]. In the context of structured overlays, this technique has the advantage that the checks can be generated offline based on the overlay and application it is intended to support before the system is even deployed. The more restrictive the communication model the stronger the invariant relationships. Thus, this technique takes advantage of the fact that structured overlays typically impose strong constraints on the routing tables that must be maintained for the system to function correctly. For example, previous research has leveraged strong invariants with the constrained partitioning of node identifiers in structured overlays to detect attacks [16, 17]. More recent work on defending against Eclipse attacks provides another example of imposing invariants on objects related to the in-degree and out-degree of an overlay node. This allows participating nodes to detect Eclipse attacks on the system performed by a relatively small number of nodes [18, 19].

Invariant relationships can also be leveraged to enhance the feedforward regulation mechanism of an adaptive system such as a measurement-based unstructured overlay. Below we discuss how such techniques can be used in the context of ESM. The decision process for ESM focuses exclusively on using local characteristics in order to feedforward adaptivity decisions. We propose to define local characteristics as either primary observations or secondary observations that it probes for from a

single hop away in the overlay. These probes are typically sent to a random subset of participating nodes and the responses contain the RTT, bandwidth, latency, and path from the source to that node. A node can leverage the path information in order to evaluate the semantic integrity of the dissemination structure and the invariant relationships within the dissemination tree. The accumulated path information in a probe cycle can also be used to generate a partially observable global context from the perspective of the node. The invariant relationships in this case include monotonic relationships such as decreasing bandwidth or increasing latency or verifying attested to relationships such as parent-child relationships and parent-child stay times.

5.2.2 Topology Awareness

Another technique which has been used in structured overlays to improve performance has been topology awareness [48–50]. Information about the underlying topology, garnered through measurement based heuristics, during path selection is leveraged in order to increase reliability and performance by expediting recovery from path outages and congestion.

Contextual awareness can also be useful in dealing with repulsion and disruption attacks. We propose to leverage information collected across multiple layers of logical networks. ESM is particularly amenable to this technique because there are current three logical networks that instantiates its infrastructure. The first network is the logical BGP network which describes the routing infrastructure. The second is the overlay that is built and the final logical network is the multicast tree that is built on top of the overlay. We propose to modify ESM such as part of the probe cycle a node will also enumerate the path between itself and all of its neighbors over the routing infrastructure as determined by BGP. This allows correlating first order observations with a periodically developed view of the BGP topology. By performing such correlations we can determine where a particular disturbance may be located

and improve both if and how an adaptation will take place. This approach basically embeds lower level information to help improve the decisions made at the higher layers of the logical network.

5.2.3 Outlier Detection by Using Local Spatial and Temporal Correlation

The solution we propose in this work which has not be previously explored in overlay networks consists of determining which nodes are advertising inconsistent metrics by performing outlier analysis on the information received from probed nodes and used in the decision process. We believe this solution will allow us to address problems with local attacks and larger numbers of attackers that could not be addressed with the aforementioned techniques used in isolation, while imposing a minimal overhead on the system. An outlier is a data point that is significantly different from the rest of the data in the observation space based on some measure of distance. The nodes detected as outliers will then be discarded from the potential parent set so they will not be able to influence the multicast tree structure. The detection is performed locally by each node using spatial and temporal correlations. The *spatial outlier detection* compares the reported metrics received from each node in the set of probed nodes. The *temporal outlier detection* examines the consistency in the metrics received from an individual probed node over time. Considered metrics include probed latency, probed bandwidth and RTT.

In order to prevent suspicion from other nodes, a malicious node must insure that any lie it tells is:

1. consistent with what the other peers are reporting during a probe cycle about current conditions (external with respect to the rest of the world)

2. consistent with the bandwidth, latency, and influence yielded towards the RTT (internal with respect to other metrics within a set of dependent variables)

3. consistent with what it said in the past.

The spatial outlier detection targets the first and second aspects of consistency, while the temporal outlier detection targets the second and third aspects. These consistencies relate to measured correlation relationships.

The intuition behind our solution is that an attacker will have difficulty lying consistently because it does not have perfect knowledge of the observation space and does not have a guaranteed feedback loop to coordinate with other attackers. For example, a set of colluding malicious nodes will have difficulty lying in a manner consistent with information provided by other peers probed during the same probe cycle. This is because malicious nodes cannot accurately predict the random subset of nodes that will be queried during the probe cycle and only have a finite amount of time, the probe period, to coordinate. In addition, the intrinsic dependency existent in several measured variables requires attackers to make sure that the "fake" metrics vary in a consistent manner. This dependency relates to the fact that all the metrics tuples are measuring characteristics of the underlying substrate path between nodes and these paths overlap.

A key component of our approach is using the Mahalanobis [51] distance as the mechanism to detect outliers. The Mahalanobis distance has several advantages that make it appropriate for our problem:

- It has been shown to be better than other distance functions for detecting outliers with multiple attributes [52]. In our case, we can use several attributes in the detection process since each node reports latency, RTT, and bandwidth.

- It takes into account the variance and covariance of the attributes that are measured by scaling each variable based on its standard deviation and covariance. This means that the attributes with high variance receive less weight than components with low variance.

- It takes into consideration the correlation between attributes and how the measured attributes change in relation to each other. This makes it appropriate for

our environment where there is a dependency between the attributes reported by each node.

Spatial Outlier Detection

Currently, the utility function at each node relies on the observation tuple consisting of <bandwidth, latency, RTT>, recorded every probing cycle. Our spatial outlier detection uses the same observation space as the utility function. Thus, it does not add any communication overhead. Spatial outlier detection is performed during each probing period as follows. The observation tuples are used to compute the centroid of the data set. We then compare how far the observation tuple for each node is away from the centroid. The comparison is done by computing the Mahalanobis distance between each node and the centroid. As previously mentioned the Mahalanobis distance takes into account the variability of the variables and the correlation between variables. It is computed as follows:

$$d(\vec{x}, \vec{y}) = \sqrt{((\vec{x} - \vec{y})^T C^{-1} (\vec{x} - \vec{y}))}$$

$\vec{x}$ and $\vec{y}$ are the feature vectors which in this case include bandwidth, latency, and RTT. $\vec{x}$ is the value from the probe response and $\vec{y}$ is the average value that was calculated. C^{-1} is the inverse covariance matrix developed from the observation tuples.

Two important special cases must be considered. The first case is when there are not enough observation tuple responses received during a probe cycle. In this case we compare the observation tuples received against the most recent centroid, if available. The other special case is when there is no variance between the observation tuples that are received. (We actually encountered this situation when running experiments on a high speed local area network.) In this case, we can not compute the Mahalanobis distance since the determinant of the covariance matrix becomes zero. To address the problem we randomly choose a potential parent from that probe

set of observation tuples and compare it to the most recent centroid, if available. If no centroid is available we postpone the decision to the next probe cycle.

Temporal Outlier Detection

The use of temporal outlier detection is motivated by the fact that a node's view of its peers is manifested through periodic samplings of metrics that can be correlated over time. We use incremental learning to develop models for the nodes currently in the peer group of a node during the course of a multicast session. Incremental learning allows our models to improve over time as more data is collected and old data is decayed [51]. In this context, we use temporal correlation to maintain a sense of history within the system which allows us to compare the metrics received in the current cycle with the information we have received in the past from those nodes in our current peer list.

The technique we use is based on the "simplified Mahalanobis distance" presented in [51]:

$$d(x, \vec{y}) = \sum_{i=0}^{n-1}(|x_i - \vec{y_i}|/(\vec{\sigma_i} + \alpha))$$

In this equation n is 3, related to the three metrics we are currently using <bandwidth,latency,RTT>, $\vec{\sigma_i}$ is the standard deviation, and α is the smoothing factor. In order to reduce the overhead of maintaining the entire set of observations we also make a simplifying assumption that the metrics are statistically independent. We trade-off accuracy in the distance function related to the covariance of the metrics for the amount of data we must maintain. Thus, we do not need to maintain the entire history of sampled values, which continues to grow over time.

Each node will additionally maintain the mean, standard deviation, and sample count associated with the observation tuple within the routing table entry for each of the peers. These values, which are stored in the routing table, will represent the temporal centroid associated with the respective peer. This centroid is incrementally updated with observations received during each probe cycle, as in [51], using

the technique Knuth described in [53]. At the end of the probe cycle the latest observation tuple associated with each peer is compared with the centroid using the Mahalanobis distance. If the distance is greater than the threshold, then this node is considered a temporal outlier.

Utility Driven Spatio-Temporal Fusion

We now show how the spatial and temporal outlier detection techniques are used during the adaptation process, using a technique similar to a codebook [54]. The decision process checks if there are a large number of nodes that are substantially far from their historical centroid (temporal outliers). If there are, then an adaptation does not occur during this probe cycle (artificial influence of aversion repulsion attack). If not, then the decision process continues. Next, the peer nodes are ranked according to their spatial outlier distance from the centroid. Peer nodes are then traversed moving from those nodes closest to the centroid to those nodes farthest from the centroid. The node that is closest to the centroid, is neither a spatial or horizontal outlier, and has passed the utility function (made it to the short list) is chosen as the new parent. At this point, the new parent request is sent. If no peer is found which meets these criteria, then no adaptation is performed during this probe cycle.

5.3 Increasing Stability

Previously we extolled the importance of stability considerations when building robust overlay networks. Dynamical measurement based routing infrastructures have a propensity for experiencing instability. A system is typically characterized by a minimum utility threshold in relation to some performance metric, which is generally intricately tied to the the goal and intent of the system. As long as the random variable being monitored stays above this threshold, the system is experiencing acceptable performance and it does not necessitate a change. On the other hand, once

the random variable falls below this utility threshold, the system performs a state change, or adaptation, in an attempt to return to an acceptable level of performance. Within adaptive unstructured overlay networks, these state changes are manifested in the form of parent changes. We view stability as the ability of the system to maintain an apriori established level of operation within specified tolerances and constraints under varying external conditions as measured through both primary and secondary measurements. Our solution to increase system stability is by including several metrics to reflect stability such as the time a node was connected to his current parent, the frequency of changes, or even the degree of variance in metrics. Stability can also be reflected in the correlation between previous feedforward and feedbackward actions taken. By making adaptations that consider stability as part of their optimization function, the nodes perceived as unstable will be pushed to the fringes of the tree as no other node will select them as a parent.

5.4 Detecting and Recovering from Bad Adaptations

Another important component of an adaptive overlay is the ability to maintain the presubscribed goals of the network by detecting and responding to perturbations. The first mechanism of detecting deviations in performance often comes for free as being a adaptive control system. If the system begins to detect degraded service it will seek a new equilibrium point since the malicious parent will cause the system to become unstable in the current network conditions. As a result, the malicious parents will lose their attractiveness. The problem is that not all malicious behavior manifests itself as disturbances in performance. In order to deal with attacks that do not create immediately observable ramifications to the adaptation loop, the system must also have mechanisms to detect violations of invariant relationships that exist within the system. This is similar to the type of techniques that are employed in structured overlays [16–19]. This could include such things as invariants in the dissemination structure or invariants in neighbor table or routing table. As in the case

of preventing unnecessary adaptations the path information can also be exploited as a means of verifying invariant relationships.

5.5 Responding to Malicious Nodes

As previously mentioned, it is imperative for the system to continue to function autonomously despite a fraction of the nodes, *f*, being malicious. Though, in many cases, these defenses cannot be maintained indefinitely and it would be advantageous for the system to have a mechanism to neutralize a suspected threat. Overlay networks have often been designed to deal with node failures but an open problem remains in the fact that overlay networks typically don't have a mechanism to remove nodes that are misbehaving [11,16]. The major difficulties come from trying to recognize and eliminate malicious nodes in a decentralized open architecture, such as an overlay, where a member node cannot unequivocally trust the information from its peers. Contributing to this agreement problem is the fact that it is often hard to distinguish between a misbehaving node and the ephemeral and transient changes and failures in the Internet's routing infrastructure.

In order to address these problems, we propose to leverage a hybrid approach that allows us to address the lack of a completely centralized architecture and the uncertainty that makes draconian node ejection difficult. Our solution exploits two important aspects of the system. The first is the limited capacity trusted source and the other is the random gossip based peer updates sent across the richly connected mesh, upon which the dissemination tree is built. In our solution, we have explored a two stage mechanism of gradual response where each node of the overlay maintains two dichotomous lists: a local suspects list and the global black list.

The suspects list is a local list maintained by each node composed of the peers it has locally detected as acting suspicious. Once a node is placed on the suspects list, it will be displaceable as child node, they will not be chosen as a parent, they will not be propagated during gossip updates (amplified), gossip membership updates

will also not be accepted from them, and finally they may eventually be reported to the source. During the random gossip based peer updates, the nodes of the overlay will also compare their suspects lists. Those nodes that exist across multiple suspects lists will be reported to the trusted source. In essence, these nodes on the suspects list will be shunned towards the edges of the tree thereby reducing their potential impact on other nodes.

Based on the reports it receives from its peers and other information, the trusted source creates a global blacklist. The nodes on the blacklist have been ejected from the overlay for a period of time or permanently exiled. The contents of this list are completely controlled by the trusted source and are periodically disseminated both over the dissemination tree and gossip updates. The nodes on the blacklist are immediately removed from their position in the overlay, the encryption keys will be changed, and the other nodes will no longer initiate connections to them nor accept connections from them. The major difference between the suspects list and the blacklist is the fact that the nodes on the blacklist will no longer be allowed to be children and they will no longer be probed. The trusted source builds this list based on information it has collected from the nodes of the overlay where each node has the ability to offer evidence towards a peers inclusion in black list. If enough nodes continually attest to the suspiciousness of a particular node the source may determine to globally blacklist that node.

5.6 Summary

In this chapter, we identified several components that are necessary for providing a comprehensive solution for mitigating Byzantine attacks that exploit adaptivity in overlay networks. These components are: reducing unnecessary and unnatural adaptations, increasing stability by incorporating metrics that reflect stability into the decision process, detecting and adapting to malicious behavior, and responding to malicious nodes. We also described in details novel techniques which have not

been adequately explored in overlay networks but would contribute substantially towards improving their robustness.

6 EXPERIMENTAL RESULTS

The goal of an attacker is to maximize its influence beyond what comes naturally by participating non-maliciously in the system. Our solution, the utility driven spatio-temporal outlier detection, constrains the ability of the attackers to artificially influence the system. (In a constrained system the variety of states the system can assume is smaller than the variety of states that can be conceived [33].)

In this chapter we demonstrate the effectiveness of the outlier detection methods proposed in the previous chapter and elucidate why this is an important contribution to the problem of resilient adaptivity. We show through experimental results that the utility driven spatio-temporal technique can be used to constrain the mechanisms by which the attacker can influence the system and bound the magnamity of that influence. The temporal correlations bound the variety of states the attacker can assume in relation to things reported in the past and the network substrate between nodes. The spatial correlations bound the states of the attacker based on the current state of other nodes and current network conditions in the system. In the following experiments we isolate different aspects of this complex system in order to maintain some experimental control and demonstrate the effectiveness at bounding the attacker.

6.1 Testbed and Experiment Setup

We conducted our experiments on the PlanetLab [25] and DETER [26] testbeds. The experiments are similar to those presented in the Section 4.2, but this time our outlier detection mechanisms have been added. There are several parameters that characterize our experiments. We use ESM deployments of 30 and 50 nodes as specified. All experiments were run for 90 minutes. The node join-leave pattern is

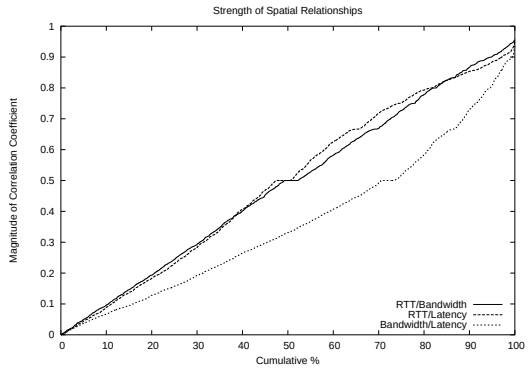


Figure 6.1. A cumulative distribution function representing the strength of the relationship between the metrics used for spatial correlations (RTT, bandwidth, latency). The metrics were collected during a 90 minute experiment with 118 nodes and consisted of 3305 spatial correlations.

a uniform join; all the nodes joined in the beginning and stayed for an average of 80 to 85 minutes. We use a degree saturation of 4 nodes. The source generates a constant bit rate of 480 Kbps. The probe cycle was set to 7 seconds.

6.2 Spatial Outlier Detection

As previously mentioned, the spatial outlier detection algorithm that we deploy utilizes the Mahalanobis distance since it has proven useful for multiple attribute outlier detection, allows us to leverage the correlation between attributes, and takes into account the attributes variance when calculating the distance function. In this protocol spatial outlier detection is being performed over three attributes: RTT, latency, and bandwidth. The distance function performs better when the attributes are correlated and there are noticeable differences in variances of the attributes. Figure 6.1 displays a cumulative distribution function representing the strength of the relationships between metrics collected over 3305 spatial correlations performed in an experiment run for 90 minutes with 118 nodes. From this graph we can see that there is a noticeable correlation between the attributes. Figure 6.2 represents

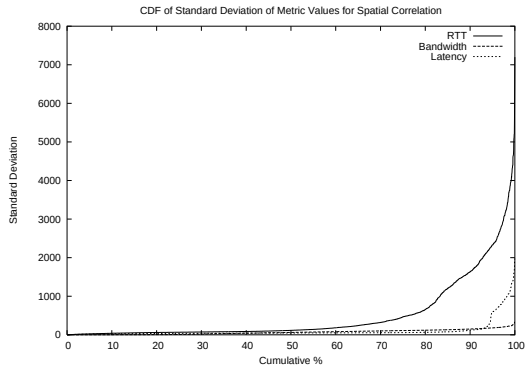


Figure 6.2. A cumulative distribution function representing the standard deviation associated with each of the metrics used for spatial correlations (RTT, bandwidth, latency). The metrics were collected during a 90 minute experiment with 118 nodes and consisted of 3305 spatial correlations.

a cumulative distribution function of the attributes standard deviation in the same experiment. From this graph, we see that there is a noticeable difference in the variances associated with the different attributes being used and thus it makes sense that the variables be weighted accordingly. As a result the Mahalanobis can leverage these characteristics of the data to improve the distance function.

Now that we have motivated the use of the Mahalanobis distance for spatial outlier detection, we begin the experiments by focusing on a single attacker and then we will expand to consider the case of colluding attackers as described in the attack model.

Effectiveness of Mahalanobis Distance at Detecting Outliers: The purpose of the first experiment is to demonstrate the effectiveness of the Mahalanobis distance in distinguishing spatial outliers thereby constraining the way an attacker can lie. The goal of the experiment is to demonstrate the techniques ability to detect outliers in the probe set and demonstrate these abilities across a heterogeneous set of environments. This experiment also demonstrates the fact that an attacker who lies consistently to different peers will manifest itself differently to each of those nodes

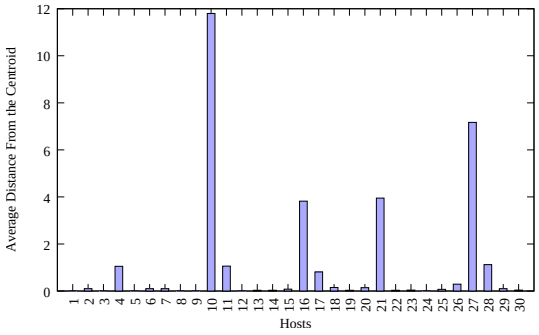


Figure 6.3. Nodes identified as outliers on an ESM deployment of 30 nodes on PlanetLab over a 90 minutes run using spatial outlier detection.

since each node develops it own centroid based on information measured locally. It is not possible for the attacker to predict this centroid without having complete knowledge of the node being attacked. In this example, the malicious node has the goal of obtaining the largest influence possible on the overlay and thus claims to have the best bandwidth (480Kbps), the best latency (0), and no saturation (false). This is an example of an attraction attack where the malicious node tries to convince neighboring nodes that it is a stronger equilibrium point. One random node was chosen to be malicious on an overlay of 30 nodes deployed on the PlanetLab testbed.

At the end of each probe cycle we calculated the centroid from all the probe responses received during the probe cycle and the distance of each probe response from the centroid. Next, we calculated the average distance from the centroid for each probe cycle for a single node. Finally, we calculated the average distance across all the end-system nodes involved in the experiment. This number is generated based on a total of 539,739 probe responses that were received during 19,465 probe cycles. Figure 6.3 presents the average distance each node was from the centroid across all the probe cycles and averaged across all end systems. Several nodes are detected as outliers, including the malicious node 4. The graph indicates that the malicious node was seen as anomalous on average across all the nodes in the system, despite the

Table 6.1
Averaged bandwidth, RTT, and latency and average distance from the average centroid for a deployment of 30 nodes on PlanetLab over a 90 minutes run.

Node	RTT	Bandwidth	Latency	Distance
4	109.83	480.00	0.00	1.05
10	2678.88	89.43	612.64	11.80
11	185.74	480.00	0.00	1.06
16	479.46	445.62	218.30	3.82
17	192.25	469.41	2.00	0.81
21	522.58	449.61	195.81	3.95
27	484.06	160.88	167.00	7.17
28	342.11	469.39	144.10	1.12
	265.36	**454.65**	**64.64**	

Table 6.2
Number of unique hosts that each of the outlier nodes appeared in the short list of.

Node ID	4	10	11	16	17	21	27	28
# Short Lists	9	2	18	2	3	2	1	9

fact that their centroids were independently developed. As a result, it demonstrates the utility of Mahalanobis distance for distinguishing outliers.

Effectiveness of Spatial Outlier Detection: We now analyze the results in detail and show how, by combining the utility function with the spatial outlier information, the system can avoid making decisions influenced by malicious adversaries. Table 6.1 presents the average values for the probed metrics of RTT, bandwidth, and latency for all noticeable outliers. The bottom row shows the average centroid

Table 6.3
The effectiveness of spatial outlier detection at improving parent selection on a deployment of 50 nodes on PlanetLab over 90 minute runs.

	Malicious Selected	Total	Percentage
No lying	8	427	1.61%
Lying	649	1122	57.84%
Spatial	84	519	16.18 %
Spatial/Temp	22	282	7.80 %

of all nodes for each of these metrics. The right-most column shows the resulting distance between each outlier node and the centroid. In addition, Table 6.2 presents the number of times that outlier nodes in Figure 6.3 were considered as possible candidates for a parent, based only on the utility decision function.

As shown in Figure 6.3, there are 8 significant outliers. One of them, node 11, is the trusted source and can therefore be discarded from the outlier detection. Information provided in Table 6.1 shows that nodes 10, 16, 21, and 27 are far away from the centroid due to their poor performance. Thus, they will not be selected in the short list as possible parents because they will not pass the utility function. This is supported by Table 6.2, which shows that nodes 10, 16, 21, and 27 were considered as potential parents for at most two nodes, which were most likely nodes in the same respective LAN as the outlier. While less dramatic, this is also the case with node 28, which had an average RTT greater than twice the mean. Based on this data, it is intuitive to see that it is possible to develop a threshold to distinguish the aforementioned nodes, as well as 4 and 17, as outliers. Empirically, we have found that an effective threshold typically lies between 1 and 2, although this could be set based on the security requirements of each node. Because they were flagged as outliers, these nodes would not have been chosen as parents. This avoids selecting the malicious node, 4, as a parent.

Note that the experiment also shows that a node could be identified as an outlier because its performance is much worse than the performance of the other peers in the probe set, much better than other nodes in the probe set, or simply inconsistent. Regardless of the cause, by not choosing outliers the system achieves increased stability.

To demonstrate the effectiveness of spatial correlation and the Mahalanobis distance function at improving the parent selection process and the stability of the system, we repeated the same experiment as above for a deployment of 50 nodes and recorded the number of parent changes that took place for the duration of the experiment. The outcome of these experiments is shown in Table 6.3. The numbers in the table are summed across the 50 nodes in the experiment. The results indicate that using our outlier detection scheme has dramatically reduced the likelihood of choosing the malicious node as a new parent. Our method also dramatically improved the stability of the network, as measured by a decrease in parent changes, in spite of the presence of the malicious node. In fact, the number of adaptations is comparable to the number of adaptations that would occur with no malicious nodes present in the network.

As previously stated, the method does not completely eliminate bad adaptations. However, we believe this indicates the need to augment better decision techniques with an ability to detect and recover from the inevitable bad adaptations that will still occur. Our outlier detection mechanism is therefore only one major component of a comprehensive solution.

Coalitions of Attackers and Spatial Outlier Detection:

The previous experiment demonstrated the effectiveness of the spatial correlation for detecting outliers with a single aggressive attacker. In the following experiments, we consider the constrained collusion model presented in Section 3.2. These experiments focus on demonstrating the cost incurred by a group of attackers in order to increase their influence on the system during adaptation and bypass the outlier detection mechanism. A simplified graphical representation of our spatial outlier

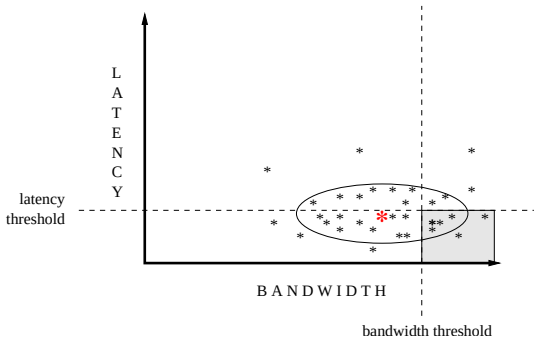


Figure 6.4. A simplified graphical representation of our spatial outlier detection technique. The dots represent data points in two-dimensional space of bandwidth and latency values collected during a probe cycle. The highlighed dot in the middle of the ellipse represents the centroid and the ellipse represents the spatial outlier threshold. The dashed lines represent the thresholds established each probe cycle by the utility function and the shaded region represents the area where a data point must lie to be considered as a new parent.

detection technique can be seen in Figure 6.4 and will be used to facilitate the discussion. It is simplified by the fact that the diagram does not consider the third dimension of RTT but this simply adds another constraint on the attacker and it could complicate the discussion. Since an attacker will not be able to predict the centroid, represented by the bold data point at the center of the ellipse in Figure 6.4, without complete knowledge of the node being attacked, the group of colluding attackers may attempt to shift the centroid so they are not perceived as outliers anymore. We use probe responses from a randomly selected probe cycle collected from real data on PlanetLab to demonstrate what would be necessary for colluding attackers to influence the outlier detection technique. In these experiments we will assume that all faulty nodes are colluding in order to maximize the potential damage and we will compare the effectiveness of different attack techniques in this collusion model.

In the first experiment we consider attackers that claim to have the best performance. As previously mentioned, this increases the likelihood of a parent change

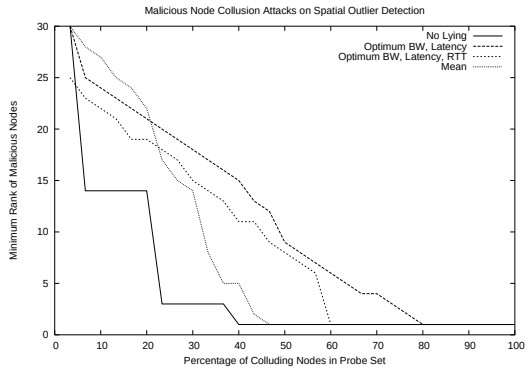


Figure 6.5. Using a representative probe cycle from data collected on PlanetLab we demonstrate the percentage of colluding nodes necessary to influence parent selection. These experiments were run with four attack scenarios but their goal is for a colluding node to be ranked first and chosen as a parent.

taking place and a malicious node being chosen. Note that a node will not be chosen as parent if he cannot pass the utility function, so by claiming the best performance the attacker guarantees that it passes the utility function as a possible parent. The utility function relates to the way these thresholds are dynamically generated each probe cycle. In Figure 6.4, this is represented by the shaded box which is above the bandwidth threshold and below the latency threshold. In order for a node to be chosen as a new parent it must be in the shaded region, the short list, within the spatial outlier threshold, represented by the ellipse, and the closest to the centroid based on the Mahalanobis Distance.

During this experiment, the malicious nodes agree to lie consistently on a set of predefined values, RTT was 0, the latency 0, and the bandwidth was 480 Kbps. We investigate what will be the required coalition size in order to make a conspiring malicious node be chosen as the new parent. In order to pass the outlier detection, a malicious node from the coalition, has to be selected as the next parent. In Figure 6.5 this attack is named “Optimum BW, Latency, RTT” and we can see that a single attacker claiming the best performance was ranked 25th among the 30 neighboring

nodes in this probe cycle, so he was not selected as a parent. In the case of a malicious coalition it took 60% of participating nodes conspiring before a malicious node was guaranteed to be chosen as the next parent.

In the previous experiment we assumed the most aggressive attacker model in attempting to maximize its ability to pass the utility function. In that case, at least one of the attackers would have had to have a 0 RTT with target node and the ability to intercept the messages to all the other malicious nodes.

In the next experiment we assume that the randomly selected nodes have to reply to their messages so they were unable to artificially reduce their RTT, but they are able to lie about bandwidth and latency. In Figure 6.5, this experiment is referred to as "Optimum BW, Latency". As seen in the data, the single attacker was ranked 30th out of 30 nodes during this probe set. In the coalition case, 80% of the nodes in the probed neighbor set need to be malicious before they could manipulate a malicious node closest to the centroid. The experiment demonstrates that both the number and type, primary or secondary, of metrics used by the outlier detection defense can make it difficult for the attacker to maintain consistency and subsequently increases the effectiveness of the spatial outlier detection.

Finally, we consider the case where the malicious nodes coordinate and take the average of the latency and bandwidth that they would have reported if they did not act maliciously. In this case we assume that the RTT cannot be artificially reduced. In Figure 6.5, this experiment is called "Mean". As in the previous example the single attacker again starts ranked 30th. In the case of a coalition between a set of malicious nodes, 47% of the nodes needed to be colluding to deterministically guarantee that a malicious node would have been chosen. This demonstrates that if the attackers have more information, then they can reduce the amount of work necessary for subverting the spatial outlier detection mechanism. In this case, work was measured by the size of the colluding coalition. This extra coordination is also constrained temporally by the RTT (primary) observable as demonstrated in the previous experiment. The experiment also demonstrates that despite the fact that

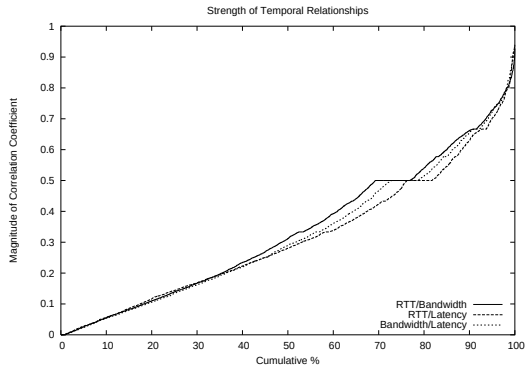


Figure 6.6. A cumulative distribution function representing the strength of the relationship between the metrics used for temporal correlations. These were collected during a 90 minute experiment with 118 nodes and consisted of 4500 temporal correlations.

the collusion is easier than with previous methods it still requires almost 50% of the nodes in the probe set being compromised before the colluding nodes obtain their goal.

As stated in the beginning the goal of an attacker was to maximize its influence beyond what comes naturally by participating non-maliciously in the system. Thus, the final case considered in Figure 6.5 is called "No Lying" and it relates to conspiring nodes that report metrics as if they did not act maliciously. In this case 40% of the nodes must act maliciously in this probe cycle before a malicious node was chosen as parent. Thus, it can be seen that lying about metrics is no longer an effective technique for obtaining the attackers goal. As we can see in these result the spatial outlier technique we describe does a good job at constraining the behavior of an attacker and reducing their ability to artificially augment their influence on the system.

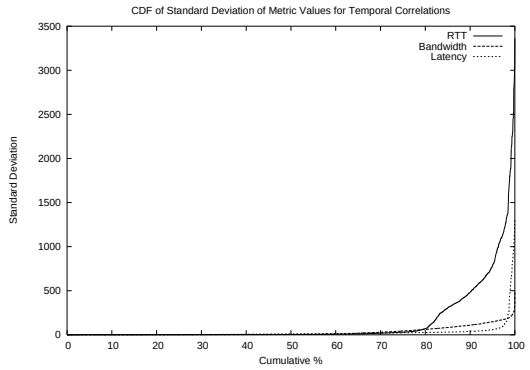


Figure 6.7. A cumulative distribution function representing the standard deviation associated with each of the metrics used for temporal correlations. These were collected during a 90 minute experiment with 118 nodes and consisted of 4500 temporal correlations.

6.3 Temporal Outlier Detection

In the previous section we described the applicability of the Mahalanobis distance for performing spatial outlier detection. In this section we explore the usefulness of the simplified Mahalanobis distance for performing temporal outlier detection. The temporal outlier detection uses the same variables as the spatial outlier detection. Figure 6.6, represents a cumulative distribution function representing the strength of the relationships between metrics collected over 4500 temporal correlations performed in an experiment run for 90 minutes with 118 nodes. As it can be seen in the figure, these variables are not correlated as strongly during temporal correlation as they are in spatial correlation. Because of the difficulty of maintaining the state necessary to perform the Mahalanobis distance we make the simplifying assumption that the variables are independent. Figure 6.7 presents a cumulative distribution function of the attributes standard deviation in this experiment. From this graph, we see that the noticeable difference in variance values can be used to improve the distance function. We leverage the variance in order to normalize the values.

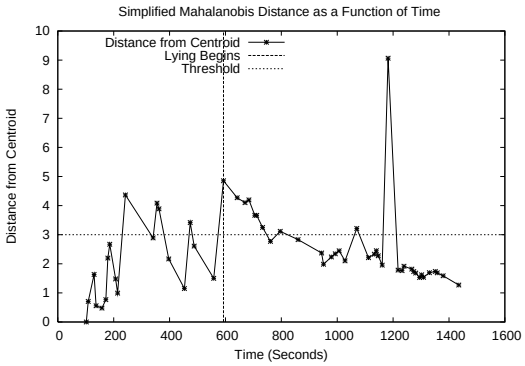


Figure 6.8. The effectiveness of detecting temporal outliers on a 50 node ESM deployment on PlanetLab

To demonstrate the effectiveness of temporal correlation and the "simplified Mahalanobis distance" function at detecting outliers, we conducted the following experiment on the PlanetLab testbed. An overlay of 50 nodes was deployed, and a random node was chosen to act maliciously by sending false reports about its metrics. Figure 6.8 presents the distance function as measured by a non-malicious node in the experiment. During our experiments the threshold for the simplified distance function was set to 3, which intuitively means that each metric can differ at least one standard deviation from its mean value. The graph presents the first 24 of the 90 minutes of the experiment, focusing on details around the point when the node begins to lie. In the first 9 minutes we see a number of ephemeral fluctuations, which is typical as the correlation begins to improve its model over time. The non-malicious node first detects that the malicious node has started lying at 593 seconds, the initial distance being measured at 4.85 from the centroid. Then 7 out of the next 8 measured distances are above the threshold value as the malicious node continues to lie. This lasts for about 200 seconds before the observation tuple begins to be seen as normal as the centroid has adapted to these falsified values. After this point we see that the model begins to converge again. This effect on convergence demonstrates also why a complete solution must include the ability to respond and neutralize malicious nodes.

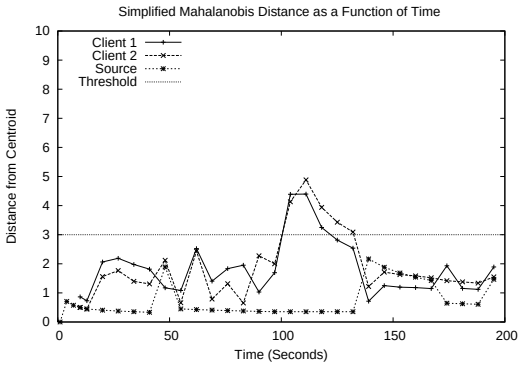


Figure 6.9. This graph demonstrates the effectiveness of detecting temporal outliers on an ESM deployment on DETER during repulsion attack

It may also be possible for a node to try and create a high variance in the metrics it reports in order to manipulate the distance function. This type of activity motivates the need for including metrics that reflect stability in the decision process, since a node with extreme variations in metrics is undesirable from a stability perspective.

Temporal outlier detection is also particularly effective at detecting repulsion attacks as seen in Figure 6.9. This would be an effective indicator if an adaptation should be made or not. Topology awareness would be necessary to help determine what change should be made.

Utility Driven Spatio-Temporal Fusion

An example of how the decision process works can be seen in Figure 6.10. The data used is based off the information provided in Figure 6.3. Remember that node 4 was the lying node. The subset of nodes that responded to the probe request are ordered in increasing distance from the centroid. In order to decide if a new parent will be selected and who that parent will be, the performance agent traverses the list beginning at those nodes closest to the centroid. The first node encountered that is on the short list (the performance agent previously selected as potential parent)

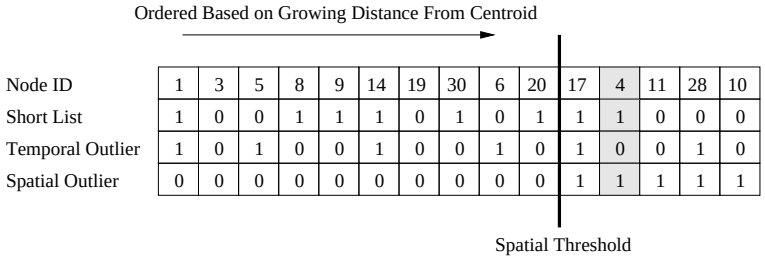


Figure 6.10. This is an example of how the utility driven spatio-temporal fusion is performed during a probe cycle based on data from Figure 6.3. The 15 nodes represent a possible probe set. In this example Node 4 was the lying node and node 8 was chosen as the new parent

and is not a temporal or spatial outlier is selected as the new parent. In Figure 6.10, node 1 is rejected for being a temporal outlier, node 3 is rejected for not being a suitable parent, node 5 is rejected for not being a suitable parent and a temporal outlier, and finally node 8 is selected as the new parent. The performance agent ceases traversing the list once it reaches the spatial threshold.

The effectiveness of this method can be also seen in the last row in Table 6.3. The row presents the results of combining the temporal and spatial correlation in the utility driven spatio-temporal fusion. In these experiments the threshold for spatial outlier detection was set at a conservative 1.5 and the threshold for temporal outlier detection was set to 3. From the results we can see that the combined technique has dramatically reduced the number of times the malicious node was chosen as a parent. With our outlier detection mechanisms enabled only 7.80% of the changes made during the experiment were to a malicious parent. In comparison, 57.84% changes to malicious parents occurred when the ESM protocol was run without our mechanisms. In addition, the combination of spatial and temporal outlier detection reduced the total number of changes that were made during the 90 minute experiment. Our method resulted in 66.04% of the total number of changes made during the experiment with no malicious nodes. Despite this reduction in the number of

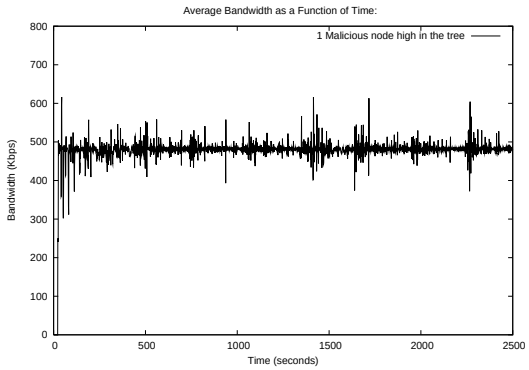


Figure 6.11. This graph demonstrates the ability of the utility driven spatio-temporal fusion mechanism to mitigate the effects seen in Figure 4.4 where one malicious node manages to maintain a high position in the tree while dropping 100% of the traffic. As in Figure 4.4, the attack begins at 1400 seconds. The experiment was conducted on PlanetLab with an overlay of 50 nodes.

changes, there was no noticeable degradation on average bandwidth, indicating that our method inhibited potentially unnecessary adaptations.

The ability of the utility driven spatio-temporal fusion to mitigate threats can be seen in Figure 6.11 which depicts the same experiment as in Figure 4.4, but with the utility driven spatio-temporal fusion enabled. Remember that in Figure 4.4, one malicious node was able to have a significant impact on the average performance of the overlay multicast tree for an extended period of time by ascertaining a powerful position in the tree and dropping 100% of the traffic. As it can be seen in Figure 6.11, our attack mitigation mechanism is able to prevent the malicious node from obtaining and maintaining the powerful position in the tree and thus the node is unable to inflict the previously described damage on the system as a whole.

6.4 Overhead

We present the overhead of our defense techniques by analyzing bandwidth, memory, and CPU utilization. Our methods do not introduce any extra bandwidth utilization since they use information that is already being exchanged between nodes.

In fact, as an artifact of reducing the number of adaptations the amount of control traffic being sent is also reduced. The memory utilization for spatial correlation only lasts for the span of a probe cycle and requires maintaining the observation tuple associated with each of the nodes in the probe set. This requires storing three additional values in the route table for the peer set maintained by each node. In order to perform the temporal correlation we modify the route table entries to store nine additional values: mean, standard deviation, and count for each of the three metrics.

Additional CPU utilization occurs only when the performance agent has selected a short list of possible parents. If the utility function does not find any suitable parents then no additional computations are performed during that probe cycle. The computational complexity is bound by the number of nodes in the probe set which is constant. The computation of the temporal outliers is a constant time calculation performed for each of the nodes in the probe set. The calculation of the spatial correlation is also computed in constant time.

6.5 Summary

In this chapter we demonstrated the effectiveness of the newly proposed outlier detection mechanism the reduce bad or unnecessary adaptations. We also demonstrated its abilities to constrain the attackers propensity to artificially influence the system. As a result we are able to limit the influence the attacker can have on the system beyond what they are capable of by participating non-maliciously. We believe this new technique complements the invariant relationships and topology awareness methods discussed in the previous chapter.

7 RELATED WORK

This chapter provides a review of relevant related work that influenced the research we have presented. In this thesis we discussed the importance of adaptive overlay systems and demonstrated the seriousness of making these systems resilient to malicious insiders that would thwart their usefulness. We also demonstrated how techniques from anomaly detection could be used to effectively constrain attackers of the system. This chapter begins by discussing the important foundations in adaptive control and its importance to the Internet, especially overlay networks. Then we delve into attacks against these systems and ideas that influenced the solutions we presented for providing the resilience necessary to build survivable unstructured measurement-based overlay infrastructures.

7.1 Adaptive Systems

It was often recognized that living organisms possessed the mechanisms to self-regulate and adapt to changes, biology refers to adaptation as "an advantageous conformation of an organism to changes in its environment" [34]. In the late 40's, Norbert Wiener named this discipline of general systems research cybernetics [34]. This work inspired the introduction of adaptive systems in control theory during the late 50's as systems which monitor their performance at meeting their presubscribed objective and then adjust accordingly to obtain improved peformance [55]. In the late 60's, emphasis turned to a particular class of systems called extremum adaptive systems, which were driven to optimize the performance as measured through particular essential variables [34]. During these early days, adaptive system were seen as important mechanisms for dealing with uncertainty and unknowns associated with potential disturbances.

Recently, there has been a resurgence in interest in adaptive systems but they are now being called by a number of different names including autonomic systems, self-managed systems, self-adaptive, self-organizing, and survivable systems. These systems were proposed as as intrusion prevention systems [56], SCADA (Supervisory Control And Data Acquisition), software systems [57], Web services [58], and overlay routing [59]. While often times the adaptivity is being used to address certain types of disturbances, previous work did not consider disturbances that are the results of concerted malicious attacks. Stronger forms of the attacks occur when the attacker is actually part of the system. The research in this thesis addresses this problem in a dynamical distributed system, adaptive end-system multicast. We explored how we can exploit the benefits of adaptivity while still providing resiliency against attacks.

7.2 Adaptivity in Network Protocols

The benefits of employing measurement based adaptivity to address intermittent failures and degraded performance associated with dynamic network conditions have been recognized since the earliest days of the ARPANET [27]. The dynamical nature of those protocols was considered an important feature in dealing with delays and congestion. As the designers of these protocols observed that the added dynamism could also lead to problems, these protocols began to evolve. As the Internet continued to grow the dynamics started to be replaced by a need for scalability, stability, and simplicity, which resulted in less frequent updates and more summarization and aggregation. This evolution led to the current version of BGP, which has been able to scale as the size of the Internet has exploded. While the Internet was able to scale effectively with the use of BGP, it became less resilient and the performance began to suffer. As a result, new protocols and architectures proposed moving adaptation mechanisms into end-to-end protocols including TCP and overlay networks. For example the use of adaptation in overlay networks has been demonstrated to offer the ability to improve the resiliency and improve route selection [30] .

7.3 Attacks Exploiting Adaptivity

This increase in the use of adaptivity in end-to-end protocols has also begun to elucidate the complications with deploying these mechanisms in an open environment such as the Internet. As previously mentioned research has been done to classify and demonstrate attacks against structured overlay networks [16,18,19,21] but this work has not focused on attacks against adaptive unstructured overlays as the examples presented in our research. The first research to consider misbehaving nodes in this class of unstructured multicast overlay networks was [23] which described selfish adversarial behavior (i.e. nodes that want to obtain advantage over other nodes, but do not have destructive goals as malicious adversaries) where nodes can selfishly improve their performance by manipulating distance measurements and cheating as independent end-systems. Our work is different in the fact that considers a malicious attacker and presents results in the context of a real system in real deployments over the Internet.

Previous research has shown the vulnerability of the TCP adaptivity mechanisms, i.e. the congestion control mechanism, to attacks from malicious outsiders [9]. The authors showed that by manipulating the end-system's perception of network congestion, the adaptivity mechanism could be used to perform a low-rate DOS attack with severe effects on TCP throughput.

In [10], the authors generalize the attack against TCP [9] as a form of low-rate ROQ attack targeting adaptive control loops that drive resource allocation and affect perceived service of a system (bandwidth, jitter, etc). The authors model the problem analytically by using control theoretic models. They cast the adaptivity as an optimization process where multiple control loops adaptively converge to a stable operating point. The analysis focuses on attacks that create noisy feedback for the controllers where the mechanisms being used by the attacker are short bursts of traffic (square wave pattern). Simulations and experiments demonstrate the validity of models for these attacks on active queue management (AQM) techniques and

TCP's AIMD rules for congestion control. The work points out the interesting observation that the more aggressively or greedily a protocol attempts to optimize the more susceptible it becomes to these attacks.

The fundamental difference between the work in [10] and our work lies in the adversarial model: We consider Byzantine adversaries, which can cause stronger attacks without using a significant computational effort, and unlike TCP we have no assumption of trust between the two end-systems. The attacks identified in [10], are more general, than those presented in [9], and the transients created by the attack are similar to those experienced in normal conditions. As a result, it makes it difficult for the resource to realize it is under attack since it would have to monitor a large range of times scales. In our case, the nature of the attacks and of the application and deployment environment allows us to go one step further than [10] and propose a solution. We have demonstrated that detection is possible by using context sensitive observation spaces and correlated information associated with the same information that drives the adaptation.

7.4 Anomaly Detection

Anomaly detection was first formally introduced as a mechanism to detect security violations by Denning [60], who represented the behavior of subjects using statistical metrics and models of observable activity. In this work she also demonstrated the advantages of having heterogeneous statistical models across subjects based on a local samplings of their observation space, a technique we exploit in our solution. Anomaly detection has also been previously leveraged to address insider threats in distributed protocols. For example, the benefits of using statistical anomaly detection to detect insider attacks against link-state routing protocols has been demonstrated in [61]. The proposed method used intrusion detection systems which do not require changes to the protocol but passively monitor the network looking for perturbations in the observation space. In our work, we use context sen-

sitive anomaly detection that is incorporated into the protocol so it has the semantic understanding of the data. The major advantage of our approach is that the observation space of the detection mechanism and adaptation mechanisms are tightly coupled allowing application centric semantically rich Byzantine detection. This tight coupling is necessary in autonomic systems and also reduces it susceptibility to classic obfuscation attacks on intrusion detection [62].

Recently, the benefits of the Mahalanobis distance for statistical anomaly detection have been demonstrated in the context of network intrusion detection [51, 63]. In [63] the authors present a comparative study of detection schemes based on data mining techniques and one of the distance schemes they evaluate relies on Mahalanobis distance. In [51] the authors discuss an unsupervised packet payload network anomaly detector based on a "simplified Mahalanobis distance" that was used to detect worm attacks. While this previous work was run on data sets, our research was peformed in a live distributed system and we used an incremental unsupervised variation of the Mahalanobis distance to perform both spatial and temporal correlations.

In the technique we presented we focused on reducing the likelihood of making unnecessary or unnatural adaptations as opposed to letting them happen and then trying to detect them. In essence trying to improve the quality of the feedforward actions taken by the system. Recently, similar work has been done in the context of inter-domain routing messages for BGP. In [64] the authors use these techniques to reduce the likelihood of a router accepting invalid routes. Anomaly detection is used to detect inconsistencies in the topology information and geographical location data. A solution proposed for the multiple origin AS conflicts in BGP also makes use of similar techniques [46].

7.5 Spatial and Temporal Correlations

Correlation has been used in both sensor network and ad-hoc networks for the detection of malicious nodes [65,66]. In ad-hoc networks, research has demonstrated how strong feature correlation found in normal behavior can be used to detect attacks [65]. Most of this research focused on the evaluation of off-line data, which was developed in a simulator, to distinguish outliers. The main differences with our work, besides the fact that we focus on overlay networks, is that the correlation is actually incorporated in-line with the protocol as it tries to adapt, analysis was performed on the Internet with real data, and we are fusing multiple correlations to improve our predictive abilities. In [66], they augment a sensor network with spatio-temporal correlation to detect misinformation being injected into the sensor streams. But in our research we are concerned with an attacker manipulating the control information in order to influence system adaptation.

Spatial and temporal correlations were also previously used in the context of network security. A notable work in this aspect is [54] where authors use temporal and spatial correlations to trace back attacks and detect attack scenarios, using a large amount of information available from intrusion detection systems, firewalls, and different software logs. Unlike the approach in [54], which was more general, our work focuses on overlay networks and does not look for correlations, but exploits the fact that they exist to detect inconsistent metrics and find suspicious nodes.

7.6 Summary

This chapter provided a review of related work that influenced the research presented in this thesis and demonstrated the motivation for the project. Adaptivity has been and continues to be an important characteristic for dealing with uncertainties and unknowns associated with disturbances. Open environments like the Internet must be built resilient so that adaptation mechanisms do not become a potential liability and vulnerability of the system. Anomaly detection offers the ability to add

diversity to distributed system and constrain a malicious adversary. In this work we also demonstrate the advantages of using application centric semantically rich detection with advanced correlation.

8 CONCLUSION AND FUTURE WORK

In this thesis we provided a characterization of the mechanisms currently used to achieve adaptivity in extremum overlay networks and identified insider attacks against these mechanisms. We believe that the attacks are relevant in other contexts, as adaptivity mechanisms are used in the design of sensor and wireless networks. The attacks are successful because adaptivity mechanisms are often not designed to be resilient to degenerate inputs from malicious insiders.

We demonstrated the effectiveness of the newly identified attacks against a well-known adaptive multicast overlay network, ESM [24]. Our experiments conducted in real-life deployments and emulations, demonstrate that although ESM employs an advanced set of adaptivity mechanisms it is unable to mitigate the attacks posed by a malicious adversary.

We provided an initial analysis of how such attacks can be mitigated and prevented throughout the life-cycle of the unstructured measurement-based overlay, and an in-depth solution to a critical aspect of the problem: preventing poor adaptation decisions in networks influenced by attackers. Our solution relies in performing spatial and temporal outlier analysis on primary (measured) and secondary (probed) metrics to allow an honest node to make better use of available information before making a feedforward adaptation decision.

We demonstrated the benefits of using our outlier detection mechanisms to improve the adaptation process and the overall stability of the system in the context of ESM through experiments conducted in real deployments. Our techniques must be combined with a detection and response mechanism to eliminate the malicious nodes. In future work we would like to address this aspect, which will also allow us to experiment with a larger number of malicious nodes under a diverse attack pattern.

While the experiments in this research were done on a relatively small scale, unlike previous research this work was done on real systems deployed in the Internet as opposed to simulations. In future work we plan to address the scalability of this solution by increasing the deployments sizes and possibly augmenting the analysis with simulation. Another area currently under research is developing a mathematical model to represent the system. In addition, we would like to investigate in depth how to further decrease the number of unnecessary changes by integrating metrics of stability in the function that drives the adaptation. We would also like to formalize the analysis of the current techniques being explored and consider different attack models.

LIST OF REFERENCES

LIST OF REFERENCES

[1] S. E. Deering. Multicast routing in internetworks and extended LANs. In *SIGCOMM '88: Symposium Proceedings on Communications Architectures and Protocols*, pages 55–64, New York, NY, USA, 1988. ACM Press.

[2] J. H. Saltzer, D. P. Reed, and D. D. Clark. End-to-end arguments in system design. *ACM Transactions in Computer Systems*, 2(4):277–288, 1984.

[3] 2005 e-crime watch survey – survey results. `http://www.cert.org/archive/pdf/ecrimesurvey05.pdf`.

[4] Miguel Castro, Manuel Costa, and Antony Rowstron. Should we build Gnutella on a structured overlay? *SIGCOMM Computer Communication Review*, 34(1):131–136, 2004.

[5] Ion Stoica, Robert Morris, David Karger, M. Frans Kaashoek, and Hari Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. *SIGCOMM Computer Communication Review*, 31(4):149–160, 2001.

[6] Antony I. T. Rowstron and Peter Druschel. Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. In *Middleware '01: Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms*, pages 329–350, London, UK, 2001. Springer-Verlag.

[7] Sylvia Ratnasamy, Paul Francis, Mark Handley, Richard Karp, and Scott Schenker. A scalable content-addressable network. In *SIGCOMM '01: Proceedings of the 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, pages 161–172, New York, NY, USA, 2001. ACM Press.

[8] Ben Y. Zhao, John D. Kubiatowicz, and Anthony D. Joseph. Tapestry: An infrastructure for fault-tolerant wide-area location and routing. Technical report, Berkeley, CA, USA, 2001.

[9] Aleksandar Kuzmanovic and Edward W. Knightly. Low-rate TCP-targeted denial of service attacks: the shrew vs. the mice and elephants. In *SIGCOMM '03: Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, pages 75–86, New York, NY, USA, 2003. ACM Press.

[10] Mina Guirguis, Azer Bestavros, and Ibrahim Matta. Exploiting the transients of adaptation for RoQ attacks on internet resources. In *The 12th IEEE International Conference on Network Protocols (ICNP'04)*, 2004.

[11] Dan S. Wallach. A survey of peer-to-peer security issues. In *International Symposium on Software Security*, pages 42–57, 2002.

[12] William J. Bolosky, John R. Douceur, David Ely, and Marvin Theimer. Feasibility of a serverless distributed file system deployed on an existing set of desktop pcs. In *SIGMETRICS '00: Proceedings of the 2000 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, pages 34–43, New York, NY, USA, 2000. ACM Press.

[13] D. Dutta, A. Goel, R. Govindan, and H. Zhang. The design of a distributed rating scheme for peer-to-peer system. In *Workshop on Economics of Peer-to-Peer Systems*, 2003.

[14] Mike Afergan and Rahul Sami. Repeated-game modeling of multicast overlays. *IEEE INFOCOM 2006. The 25th Conference on Computer Communications*, 2006.

[15] Michael K. Reiter and Aviel D. Rubin. Crowds: Anonymity for web transactions. *ACM Transactions on Information and System Security*, 1(1):66–92, 1998.

[16] Emil Sit and Robert Morris. Security considerations for peer-to-peer distributed hash tables. In *IPTPS '01: Revised Papers from the First International Workshop on Peer-to-Peer Systems*, pages 261–269, London, UK, 2002. Springer-Verlag.

[17] Miguel Castro, Peter Druschel, Ayalvadi Ganesh, Antony Rowstron, and Dan S. Wallach. Secure routing for structured peer-to-peer overlay networks. In *OSDI '02: Proceedings of the 5th Symposium on Operating Systems Design and Implementation*, pages 299–314, New York, NY, USA, 2002. ACM Press.

[18] Atul Singh, Miguel Castro, Antony Rowstron, and Peter Druschel. Defending against eclipse attacks on overlay networks. In *Proceedings of the 11th ACM SIGOPS European Workshop*, Leuven, Belgium, September 2004.

[19] Atul Singh, Tsuen-Wan Ngan, Peter Druschel, and Dan Wallach. Eclipse attacks on overlay networks: Threats and defenses. In *The 25th Conference on Computer Communications*, Barcelona, Spain, April 2006.

[20] John R. Douceur. The Sybil attack. In *IPTPS '01: Revised Papers from the First International Workshop on Peer-to-Peer Systems*, pages 251–260, London, UK, 2002. Springer-Verlag.

[21] Miguel Castro, Peter Druschel, Y. Charlie Hu, and Antony Rowstron. Exploiting network proximity in peer-to-peer overlay networks. Technical report, Microsoft Research, Cambridge, UK., 2002.

[22] K. Gummadi, R. Gummadi, S. Gribble, S. Ratnasamy, S. Shenker, and I. Stoica. The impact of DHT routing geometry on resilience and proximity. In *SIGCOMM '03: Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, pages 381–394, New York, NY, USA, 2003. ACM Press.

[23] Laurent Mathy, Nick Blundell, Vincent Roca, and Ayman El-Sayed. Impact of simple cheating in application-level multicast. In *INFOCOM*, 2004.

[24] Yang-hua Chu, Sanjay G. Rao, and Hui Zhang. A case for end system multicast (keynote address). In *SIGMETRICS '00: Proceedings of the 2000 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, pages 1–12, New York, NY, USA, 2000. ACM Press.

[25] Planetlab. `http://www.planet-lab.org/`.

[26] Deter. `http://www.isi.edu/deter/`.

[27] D. Clark. The design philosophy of the DARPA internet protocols. In *SIGCOMM '88: Symposium Proceedings on Communications Architectures and Protocols*, pages 106–114, New York, NY, USA, 1988. ACM Press.

[28] Suman Banerjee, Bobby Bhattacharjee, and Christopher Kommareddy. Scalable application layer multicast. In *SIGCOMM '02: Proceedings of the 2002 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, pages 205–217, New York, NY, USA, 2002. ACM Press.

[29] John Jannotti, David K. Gifford, Kirk L. Johnson, M. Frans Kaashoek, and James W. O'Toole, Jr. Overcast: Reliable multicasting with an overlay network. In *USENIX OSDI 2000*, 2000.

[30] David G. Andersen. Resilient Overlay Networks. Master's thesis, Massachusetts Institute of Technology, May 2001.

[31] Piyush Gupta and P. R. Kumar. A system and traffic dependent adaptive routing algorithm for ad-hoc networks. In *The 36^{th} IEEE Conference on Decision and Control*, December 1997.

[32] Santashil PalChaudhuri, Rajnish Kumar, Richard Baraniuk, and David B. Johnson. Design of adaptive overlays for multi-scale communication in sensor networks. In *The International Conference on Distributed Computing in Sensor Systems (DCOSS 2005)*, June 2005.

[33] F. Heylighen and C. Joslyn. Cybernetics and second order cybernetics. In R. Meyers, editor, *Encyclopedia of Physical Science and Technology, 3rd Edition*, volume 4, pages 155–170. Academic Press, 2001.

[34] Kumpati S. Narendra and Anuradha M. Annaswamy. *Stable Adaptive Systems*. Prentice-Hall, Inc., Englewood Cliffs, New Jersey, USA, 1989.

[35] Daniel Bauer, Sean Rooney, Paolo Scotton, Sonja Buchegger, and Ilias Iliadis. The performance of measurement-based overlay networks. In *QofIS*, pages 115–124, 2002.

[36] Yang-hua Chu, Sanjay G. Rao, Srinivasan Seshan, and Hui Zhang. Enabling conferencing applications on the internet using an overlay multicast architecture. In *ACM SIGCOMM 2001*, San Diago, CA, August 2001. ACM.

[37] Y. Rekhter and T. Li. A Border Gateway Protocol 4 (BGP 4). Internet Engineering Task Force: RFC 1771, March 1995.

[38] David Andersen, Hari Balakrishnan, Frans Kaashoek, and Robert Morris. Resilient overlay networks. volume 35, pages 131–145, New York, NY, USA, 2001. ACM Press.

[39] Craig Labovitz, Abha Ahuja, Abhijit Bose, and Farnam Jahanian. Delayed internet routing convergence. *IEEE/ACM Transactions on Networking*, 9(3):293–306, 2001.

[40] Zhuoqing Morley Mao, Ramesh Govindan, George Varghese, and Randy H. Katz. Route flap damping exacerbates Internet routing convergence. In *SIGCOMM '02: Proceedings of the 2002 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, pages 221–233, New York, NY, USA, 2002. ACM Press.

[41] Mukund Seshadri and Randy H. Katz. Dynamics of simultaneous overlay network routing. Technical Report CSD-03-1291, University of California, Berkeley, November 2003.

[42] Chunqiang Tang and Christopher Ward. Gocast: Gossip-enhanced overlay multicast for fast and dependable group communication. In *DSN '05: Proceedings of the 2005 International Conference on Dependable Systems and Networks (DSN'05)*, pages 140–149, Washington, DC, USA, 2005. IEEE Computer Society.

[43] Ben Y. Zhao, Ling Huang, John D. Kubiatowicz, and Anthony D. Joseph. Exploiting routing redundancy using a wide-area overlay. Technical Report CSD-02-1215, U. C. Berkeley, Nov 2002.

[44] Kenji Leibnitz, Naoki Wakamiya, and Masayuki Murata. Biologically inspired self-adaptive multi-path routing in overlay networks. *Communications of the ACM*, 49(3):62–67, 2006.

[45] Yang-hua Chu, Aditya Ganjam, T.S. Eugene Ng, Sanjay G. Rao, Kunwadee Sripanidkulchai, Jibin Zhan, and Hui Zhang. Early experience with an internet broadcast system based on overlay multicast. In *USENIX Annual Technical Conference, General Track*, pages 155–170, 2004.

[46] Xiaoliang Zhao, Dan Pei, Lan Wang, Dan Massey, Allison Mankin, Shyhtsun Felix Wu, and Lixia Zhang. Detection of invalid routing announcement in the internet. In *DSN '02: Proceedings of the 2002 International Conference on Dependable Systems and Networks*, pages 59–68, Washington, DC, USA, 2002. IEEE Computer Society.

[47] Nick Petroni, Tim Fraser, AAron Walters, and W.A Arbaugh. An architecture for specification-based detection of semantic integrity violations in kernel dynamic data. In *15th USENIX Security Symposium 2006*.

[48] J. Han, D. Watson, and F. Jahanian. Topology aware overlay networks. *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, 4:2554–2565, 2005.

[49] Marcel Waldvogel and Roberto Rinaldi. Efficient topology-aware overlay network. In *Proceedings of HotNets-I*, Princeton, NJ, USA, October 2002.

[50] Marcel Waldvogel and Roberto Rinaldi. Efficient topology-aware overlay network. *SIGCOMM Computer Communication Review*, 33(1):101–106, 2003.

[51] Ke Wang and Salvatore J. Stolfo. Anomalous Payload-based Network Intrusion Detection. In *Proceedings of the Recent Advances in Intrusion Detection (RAID) Conference*, September 2004.

[52] C.T. Lu, D. Chen, and Y. Kou. Multivariate spatial outlier detection. *International Journal on Artificial Intelligence Tools, World Scientific*, 13(4):801–812, December 2004.

[53] Donald Ervin Knuth. *The Art of Computer Programming, 2nd Ed. (Addison-Wesley Series in Computer Science and Information)*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1978.

[54] Guofei Jiang and George Cybenko. Temporal and spatial distributed event correlation for network security. In *American Control Conference (ACC)*, 2004.

[55] R.F Drenick and R.A. Shahbender. Adaptive servomechanisms. *AIEE Transactions*, 76(2):286–292, 1957.

[56] Michael E. Locasto, Ke Wang Angelos D. Keromytis, and Salvatore J. Stolfo. Flips: Hybrid adaptive intrusion prevention. In *Recent Advances in Intrusion Detection, 8^{th} International Symposium*, pages 82–101, September 2005.

[57] Paul Robertson and Brian Williams. A model-based system supporting automatic self-regeneration of critical software. In *IFIP/IEEE International Workshop on Self-Managed Systems & Services*, May 2005.

[58] George Porter and Randy Katz. Effective web service load-balancing through statistical monitoring. In *IFIP/IEEE International Workshop on Self-Managed Systems & Services*, May 2005.

[59] Kenji Leibnitz, Naoki Wakamiya, and Masayuki Murata. Biologically inspired adaptive multi-path routing in overlay networks. In *IFIP/IEEE International Workshop on Self-Managed Systems & Services*, May 2005.

[60] Dorothy E. Denning. An intrusion-detection model. *IEEE Transactions on Software Engineering*, 13(2):222–232, 1987.

[61] Diheng Qu, Brian M. Vetter, Feiyi Wang, Ravindra Narayan, S. Felix Wu, Y. Frank Jou, Fengmin Gong, and Chandru Sargor. Statistical anomaly detection for link-state routing protocols. In *ICNP '98: Proceedings of the Sixth International Conference on Network Protocols*, page 62, Washington, DC, USA, 1998. IEEE Computer Society.

[62] T. Ptacek and T. Newsham. Insertion, evasion, and denial of service: Eluding network intrusion detection. Technical report, Secure Networks, Inc., 1998.

[63] Aleksandar Lazarevic, Levent Ertoz, Vipin Kumar, Aysel Ozgur, and Jaideep Srivastava. A comparative study of anomaly detection schemes in network intrusion detection. In *Proceedings of the Third SIAM International Conference on Data Mining*, 2003.

[64] Christopher Krügel, Darren Mutz, William Robertson, and Fredrik Valeur. Topology-based detection of anomalous bgp messages. In Giovanni Vigna, Erland Jonsson, and Christopher Krügel, editors, *RAID*, volume 2820 of *Lecture Notes in Computer Science*, pages 17–35. Springer, 2003.

[65] Yi an Huang, Wei Fan, Wenke Lee, and Philip S. Yu. Cross-feature analysis for detecting ad-hoc routing anomalies. In *ICDCS '03: Proceedings of the 23rd International Conference on Distributed Computing Systems*, page 478, Washington, DC, USA, 2003. IEEE Computer Society.

[66] Sapon Tanachaiwiwat and Ahmed Helmy. Correlation analysis for alleviating effects of inserted data in wireless sensor networks. In *MobiQuitous*, pages 97–108. IEEE Computer Society, 2005.