A Framework for Mitigating Attacks Against Measurement-Based Adaptation Mechanisms in Unstructured Multicast Overlay Networks

AAron Walters, David Zage, *Member, IEEE,* and Cristina Nita-Rotaru, *Member, IEEE*

***Abstract*—Many multicast overlay networks maintain application-specific performance goals by dynamically adapting the overlay structure when the monitored performance becomes inadequate. This adaptation results in an unstructured overlay where no neighbor selection constraints are imposed. Although such networks provide resilience to benign failures, they are susceptible to attacks conducted by adversaries that compromise overlay nodes. Previous defense solutions proposed to address attacks against overlay networks rely on strong organizational constraints and are not effective for unstructured overlays. In this work, we identify, demonstrate and mitigate insider attacks against measurement-based adaptation mechanisms in unstructured multicast overlay networks. We propose techniques to decrease the number of incorrect adaptations by using outlier detection and limit the impact of malicious nodes by aggregating local information to derive global reputation for each node. We demonstrate the attacks and mitigation techniques through real-life deployments of a mature overlay multicast system.**

***Index Terms*—Adaptivity, Insider Attacks, Overlay Networks, Security**

I. Introduction

MULTICAST overlay networks were proposed as a viable application level multicast architecture to overcome the scarcity of native IP multicast deployments. The approach moves buffering and relaying functionality from core routers to end-systems.

Many multicast overlay networks optimize application-specific performance goals such as bandwidth, latency, jitter, and loss rate by dynamically adapting the overlay topology. The adaptation improves suboptimal overlay meshes resulting from random initial neighbor selection, aggressive partition repair, group membership changes, and transient conditions in the underlying physical network. To decrease the control overhead, each node maintains only a set of neighbor nodes and an upstream node. A node monitors its performance from the multicast source and periodically probes its neighbor nodes about their own performance. When the performance becomes inadequate, a node changes its upstream node. We refer to this process as *adaptation* and to the mechanisms used to achieve it as *adaptation mechanisms*. There are no constraints in the selection of the neighbor set and no imposed constraints in the resulting overlay. Such networks are referred to as *unstructured overlay networks* to differentiate them from *structured overlay networks* [1], where the overlay topology offers predefined bounds and organizational invariants by constraining the set of nodes eligible to become neighbors of a given node. Examples of multicast systems using structured overlay networks include Scribe [2] and SplitStream [3]; examples of multicast systems using unstructured overlays include ESM [4], Nice [5], and Overcast [6].

While pushing functionality to end-systems allows overlay networks to achieve better scalability, it also makes them vulnerable as trust is pushed to the fringes of the Internet where end-nodes are more likely to be compromised than core routers [7]. Overlay networks are more susceptible to insider attacks conducted by attackers that infiltrate the overlay or compromise some of its nodes. One attack that does not require significant work from the attacker is to exploit the adaptation mechanisms by influencing the accurate interpretation of performance observations and the correctness of the responses received from probed nodes. As a result, an attacker can influence the overlay construction and maintenance, controlling a significant part of the traffic. This facilitates further attacks such as selective data forwarding, cheating, traffic analysis, and overlay partitioning. Some attacks, such as selective data forwarding, may ultimately be noticed by the victim so they can be effectively addressed by deploying a posteriori detection mechanisms. Other attacks, such as traffic analysis, do not have immediately observable results. It is thus critical to address the primary attacks that allow the adversary to control the overlay structure maintenance.

Previous work addressing malicious attacks on overlay networks focused on structured overlays [8], [9], [10], [11], [12], [13] used for file sharing applications. The attacker may control the file discovery by manipulating the control and data messages routed within the overlay, poisoning the routing table, or partitioning the network. The proposed mitigation techniques leverage the strong organizational constraints imposed on neighbor selection and the invariant relationships between neighbors. While solutions for attacks in structured overlay networks offer valuable insights into the problem space, they are not appropriate for unstructured overlay multicast networks where no structural constraints exist between neighbors.

In this paper, we focus on identifying, demonstrating, and mitigating insider attacks in unstructured multicast overlay networks. The attacks exploit adaptation mechanisms that

Manuscript received October 25, 2006; revised September 7, 2007; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor D. Towsley

A. Walters is with 4tphi Research, Washington, D.C., USA, awalters@4tphi.net

D. Zage and C. Nita-Rotaru are with the Department of Computer Science and CERIAS, Purdue University, 305 N. University St., West Lafayette, IN 47907 USA, {zagedj,crisn}@cs.purdue.edu

Digital Object Identifier X

these networks use in order to maintain application-specific performance. Current adaptation mechanisms assume that the information reported by probed nodes is always correct and fail to take into account the effects of malicious attackers on their surrounding environment. Unlike previous work demonstrating attacks exploiting adaptivity [14], [15], our work considers the effects of insider adversaries in the context of overlay networks. A subset of the attacks we identified may also be conducted against unicast overlays, but they may be less effective depending on the nature of the metric optimized by the overlay.

We summarize our key contributions:

• 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 the attacks as *attraction*, *repulsion*, and *disruption*.

• We provide an analysis of the solution space for mitigating insider attacks that exploit measurement-based adaptation: preventing incorrect or unnecessary adaptations, increasing stability by incorporating metrics that reflect stability into the decision process, detecting observable malicious behavior such as degradation of service, and isolating the malicious nodes.

• We propose techniques to reduce incorrect or 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 logical networks share overlapping physical links.

• We propose techniques to isolate malicious nodes by aggregating the local suspicious behavior derived using outlier detection to build a global reputation for each overlay node.

• We demonstrate the effectiveness of the identified attacks and the benefits of our defense mechanisms in the context of a well-known and operationally deployed multicast system, ESM [4], through experiments and emulations conducted on the PlanetLab [16] and DETER [17] testbeds, respectively.

Roadmap: The rest of the paper is organized as follows. We specify our system and attack models in Section II. We discuss adaptation mechanisms employed by overlay networks and identify attacks against them in Section III. We propose defense mechanisms in Section IV. We present experimental results demonstrating the attacks and the defense techniques in Section V. We overview related work in Section VI and conclude our work in Section VII.

II. System and Attacker Model

System Model: We focus on overlay networks providing support for single-source broadcasting applications that are high-bandwidth (hundreds of kilobits per second) and real-time, but not interactive. The system consists of a set of nodes and a data source node communicating via unicast links. All nodes but the source have similar functionality. The nodes are not only receivers of data, but also contribute to the routing process. The source is assumed to be continually available.

The overlay construction is self-organized and distributed. No node has complete knowledge of the dissemination topology. Each node maintains a neighbor set, a routing table and the upstream node forwarding the data, referred to as the node's *parent*. The neighbor set represents only partial topology information and consists of nodes that are currently reachable in the overlay. This set is bootstrapped at join time by contacting the source and is continually updated via a membership protocol. There are no constraints placed on the members of a node's neighbor set, also referred as *peers*. The routing table represents a set of nodes that the node is responsible for routing data to, also referred to as *children*. The size of this set is limited by a system characteristic called *saturation degree*, representing the number of concurrent data streams the node is able to support before saturating the allocated bandwidth of the underlying physical network link.

Each node maintains a set of performance variables for each member of its neighbor set. These variables consist of bandwidth (throughput), latency (one-way delay), and RTT. We focus on overlays using TCP as the data transport protocol, thus loss rate is not considered. The variables are continuously measured by using passive observation and active probes. A node uses the collected performance metrics to select a new parent from its neighbor set if the performance becomes inadequate.

Attacker Model: We consider a constrained-collusion Byzantine adversary model similar to that proposed in [11], with a system size of N and a bounded percentage of malicious nodes f ($0 \leq f < 1$) behaving arbitrarily. The set of malicious nodes is partitioned into disjoint coalitions with intra-coalition cooperation possible. We assume a malicious adversary has access to all data at a node as any legitimate user would (insider access), including cryptographic keys stored at a node. This access can be the result of the adversary bypassing the authentication mechanisms or compromising a node through other means. Nodes cannot be completely trusted although they are authenticated. We assume that data authentication and integrity mechanisms are deployed and we focus only on attacks directed at the adaptation mechanisms. We assume the source is trusted and cannot be compromised. We refer to correct nodes as benign nodes or non-malicious nodes to distinguish them from malicious nodes. Even if a malicious node is dropping forwarded traffic, we assume that it legitimately receives and uses the inbound data traffic.

III. Attacks Exploiting Measurement-Based Adaptation in Overlay Networks

Any adaptive network protocol based on measurements involves periodically observing and estimating the network conditions, followed by making an adaptation decision. For unstructured multicast overlays, the adaptation decision consists of a node selecting a new parent based only on weighing the associated costs versus benefits quantified through a utility function [18].

Previous work studied the quality of the data observation and estimation, as well as the ability of the metrics to accurately reflect the state of the network. Examples of factors that influence data quality include data freshness, variability and noise. Mechanisms proposed to address these issues are data sampling [19], data smoothing [20], metric construction

[21], as well as data summarization and aggregation [22]. Previous work also studied instabilities [23], [24], [25], such as the oscillatory behavior referred to as flapping, occurring when nodes rapidly switch between seemingly equal alternatives. New techniques such as utility discretization [25], [26], randomization [25], [27], damping [24], and hysteresis [27], [19] were deployed to mitigate these phenomena and provide a tradeoff between responsiveness to change and instability.

None of the mechanisms described above take into account adversarial environments. Compromised overlay nodes can take advantage of the adaptation process to gain control over overlay traffic by lying about their observed performance metrics or artificially influencing the performance metrics observed by other nodes to manipulate the path selection or the overlay topology. We classify these attacks as *attraction attacks*, *repulsion attacks*, and *disruption attacks*. More details about each attack can be found in [28].

Attraction attacks are a form of "bait-and-switch" attacks, where a malicious node manipulates the observed data in order to present the network conditions as better than they are. The attack can also target one particular node, in which case the attacker persuades the victim to attach to a malicious parent in the tree. The final goal of the attack can be manipulating data, performing traffic analysis, performing man-in-the-middle attacks, causing disruption for specific nodes by isolating them, or selectively dropping packets for a particular destination. The victim will make an incorrect change since the perceived benefit does not reflect reality.

Repulsion attacks seek to reduce the attractiveness of other nodes or misrepresent their ability, with the ultimate goal of free-loading, traffic pattern manipulation, or augmenting attraction attacks. As in the case of attraction attacks, repulsion attacks can target one particular node. One way a malicious node can conduct the attack is by lying about its performance. For example, a malicious node may lie about route costs (i.e., hop count) in order to convince other nodes that it has a bad connection and thus it should not be selected as a parent. The malicious node will then obtain a reduced burden while still taking advantage of the system.

Disruption attacks target the availability of the network by using the adaptation process 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.

IV. Defending Against Attacks in Adaptive Overlay Networks

In this section, we describe a comprehensive solution for mitigating insider attacks that exploit adaptation in overlay networks. Since the attacks we are concerned with are performed by compromised nodes controlled by adversaries, the solution space components we describe below are complementary to authentication and integrity mechanisms.

A. Solution Space

We identify four components that a framework designed to address insider attacks against adaptivity mechanisms must include. Due to lack of space, we present a high-level description of all of them and a detailed description of two components: reducing incorrect adaptations and isolating malicious nodes. More details about each component can be found in [29].

• *(A1) Reducing incorrect adaptations:* A node makes adaptation decisions based on two types of information: the performance from the source measured directly by each node and the performance of the neighbor nodes obtained by probing a random set. By blindly accepting the information reported by the potentially malicious probed nodes, a benign node may make incorrect decisions. We propose to prevent incorrect adaptations by detecting and filtering out outliers in the metrics reported by probed nodes. Our method evaluates temporal and spatial correlations among data in the system.

• *(A2) Increasing stability:* Reducing the number of unnecessary adaptations may increase the stability and decrease the incorrect adaptations, while reducing the overhead. We propose to integrate stability metrics such as the time a node was connected to his current parent, the frequency of changes, or the degree of variance in metrics into the function that drives the adaptation.

• *(A3) Detecting observable malicious behavior:* The methods proposed above may still result in some incorrect adaptations. As the attacks exploiting adaptation are often used to further attack the multicast service, a node may observe a degradation of quality of service. This allows additional detection mechanisms to be employed. Unlike (A1), which is focused on preventing incorrect adaptations, this component reacts to degradation of service resulting from incorrect adaptation. We propose that every node uses the low-bandwidth, bidirectional unicast link that it shares with the source to provide feedback to the source about the received data. The link is also used by the source to inform member nodes about the state of the overlay structure to allow them to detect inconsistencies in the metrics reported by peers. The structural information can be trusted as it is sent by the source and protected cryptographically from modifications.

• *(A4) Isolating malicious nodes:* Filtering malicious information is not sufficient, as the malicious nodes may continue to interfere with the system, unless further action is taken to isolate them. We propose a gradual response where each node of the overlay creates a local suspects list derived using outlier detection. The local suspicious information is aggregated at the trusted source by using a reputation system to derive a global black list. The suspects list allows nodes to quickly make corrective actions locally, while the global list allows nodes to share information about malicious nodes in the system.

B. Reducing Incorrect Adaptations

The primary cause of the identified attacks is the ability of the attacker to influence the adaptation process by manipulating the performance metrics. We propose to detect inconsistent metrics by performing outlier analysis on the information received from probed nodes and used in the decision process.

An *outlier* is a data point that is significantly different (greater than a threshold) from the rest of the data in the observation space based on a measure of distance.

Each node periodically performs a *probe cycle*, in which it sends probe requests for system metrics to a random subset of neighboring nodes and receives responses for a fixed interval of time. The detection is performed locally by each node at the end of the probe cycle 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. Our outlier detection does not affect the link stress in the system, as it uses the metrics already reported by nodes: latency, bandwidth and RTT. Both latency and RTT are utilized because they are highly correlated metrics collected in different manners (one is probed, while the other is measured). In order to avoid being suspected by benign nodes, a malicious node must insure that any lie it tells: (1) is consistent with what the other peers are reporting during a probe cycle about current network conditions, (2) ensures consistency between the different dependent metrics (bandwidth, latency, and RTT), and (3) is consistent with metrics it reported 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. Spatial and temporal data correlations have been previously shown effective in detecting network attack scenarios [30]. Unlike the general approach in [30], our work does not look for correlations but exploits the fact that they exist to detect suspicious nodes.

The intuition behind our solution is that the dependency existent in the measured variables requires attackers to make sure the "fake" metrics vary in a consistent manner. This dependency results from a fundamental characteristic of end-system multicast systems – the distribution tree overlaps with the routing infrastructure. Lying is made more difficult by the fact that, in most cases, attackers can only make the RTT worse, because it is a measured attribute, and yet, at the same time, the RTT must remain consistent with both the bandwidth and latency. Our solution also forces an attacker to lie consistently with other peers. This is difficult to achieve as an attacker does not have perfect knowledge of the observation space, must accurately predict the random subset of nodes that will be probed, and only has a finite amount of time (the probe period) to coordinate with other attackers.

Our approach uses the Mahalanobis [31] distance to detect outliers. We selected this distance function because it has been shown effective at detecting outliers with multiple attributes [32], scales each variable based on its standard deviation and covariance, and takes into account how the measured attributes change in relation to each other. These features make it appropriate for our environment where there is a dependency between several of the attributes reported by each node.

Spatial outlier detection. The outlier detection is performed by a node as follows. Each probe cycle, the node first computes the centroid of the data over the three dimensional space formed by the observation tuples from all probed nodes. An *observation tuple* is represented by bandwidth, latency, and RTT. The node then computes the Mahalanobis distance between the observation tuple from each probed node and the centroid as follows [31]:

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

where $\vec{x}$ and $\vec{y}$ are the feature vectors consisting of 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 computed from the observation tuples. Any node whose Mahalanobis distance is greater than k away from the centroid is considered to be an outlier. If there are not enough tuples during a probe cycle, the tuples are compared with the most recent centroid. If there is no variance between the received observation tuples, the Mahalanobis distance cannot be computed since the determinant of the covariance matrix becomes zero. In this case, a node is randomly selected from that probe set of observation tuples and compared to the most recent centroid. If no centroid is available, the decision is postponed to the next probe cycle.

Spatial threshold selection. The threshold for our outlier detection can be mathematically derived as in [33], [34], assuming a multivariate Gaussian distribution for the metrics vector. The contours of equal probability of this distribution create a 3-dimensional ellipsoid and the outlier threshold reflects the probability of a vector being within the ellipsoid whose semi-axes are determined by k. The probability that a random vector lies within the ellipsoid increases with the value of k. Thus, for a given value of k the probability that a probed tuple lies within the ellipsoid can be computed as:

$$P = -\frac{1}{\sqrt{2\pi}} + 2\left(\frac{1}{\sqrt{2\pi}}\int_0^k e^{\frac{y^2}{2}}dy\right) - \sqrt{\frac{2}{\pi}}ke^{\frac{-k^2}{2}} \tag{2}$$

We initially selected a k of 2.37, creating a threshold which half of the probes would successfully pass. Through testing in over 539,739 probe responses during 19,465 probe cycles, we found an ellipsoid determined by a threshold of k equal to 1.5 will contain approximately 80% of the nodes. Thus, we selected a threshold of 1.5 for our experiments. This variation from the mathematically derived value can be attributed to the fact that the used metrics do not form a perfect normalized distribution and have a smaller variance than assumed in Equation 2. A node may select smaller threshold distances for stronger security guarantees, with the drawback that it may find itself isolated due to aggressive filtering.

Temporal outlier detection. We use temporal correlations to detect inconsistencies in the performance metrics reported over time by a node. We develop models for the peers of a given node during the course of a multicast session by using incremental learning. Our technique is based on the "simplified Mahalanobis distance" presented in [31]:

$$d(x,\bar{y}) = \sum_{i=0}^{n-1}(|x_i - \bar{y_i}|/(\bar{\sigma_i}+\alpha)) \tag{3}$$

where n is the number of metrics, three in our case (bandwidth, latency, and RTT), $\bar{\sigma_i}$ is the standard deviation, and α is a smoothing factor empirically set to .001 to help to avoid over-fitting and reduce false positives [31]. We trade-off accuracy of the distance function to minimize the amount of data a node must store by assuming that the metrics are statistically independent. Thus, a node does not need to maintain the whole history. Instead, a node maintains a temporal centroid for each peer consisting of the mean, standard deviation, and sample count computed from the observation tuples received over time. The centroid for each peer is incrementally updated with observations received during each probe cycle, as in [31], using the technique Knuth described in [35]. At the end of the probe cycle, the latest observation tuple for each peer is compared with the corresponding temporal centroid using the simplified Mahalanobis distance.

Temporal threshold selection. We used a threshold of 3.0 for our temporal outlier detection, to allow each of the three features to vary within one standard deviation from their temporally developed mean. The value was chosen based on the formula of the simplified Mahalanobis distance as in [31].

Spatio-temporal outlier detection. We combine the two outlier detection mechanisms described above by using a codebook technique similar to [30]. The peer nodes are ranked according to their spatial outlier distance from the spatial centroid and traversed from the closest to the farthest node. The node that is the closest to the spatial centroid and it is not a spatial or temporal outlier is chosen as the new parent. If no peer is found meeting these criteria or if there are a large number of temporal outliers, no adaptation is performed during that probe cycle.

C. Isolating Malicious Nodes

Once malicious nodes are detected, corrective measures must be taken to isolate them and minimize their effect on the overlay network. Without appropriate response mechanisms, the overall system performance may suffer as the malicious nodes continue to interfere with the system. We propose a two-prong approach which uses local observed behavior to generate an immediate local bias against misbehaving nodes and subsequently allows the overlay to construct and share global knowledge about the malicious nodes. Each node creates and maintains a local suspects list. The list is periodically sent to the trusted source which uses the collected information to construct a global list of nodes that must be banned due to their malicious behavior. The source periodically sends the generated global black list to all nodes in the overlay structure via a gossip-based protocol. Building and disseminating the global list has a higher cost than maintaining just the local list at each node. However, it has the advantage that it allows the overlay to quickly converge to a stable equilibrium point and achieve higher throughput as malicious nodes are more quickly removed from the overlay. This is because each node does not have to experience the malicious nodes' actions before being able to avoid them. Nodes are informed through the global black list and consequently are able to select beneficial parents.

1) Local Response: Each node takes immediate action based on a local suspects list created by tracking the behavior of neighbor nodes. This is achieved by recording any inconsistent metrics detected when performing outlier analysis on the information from the probed nodes. Every node computes a *suspicion value* for each neighbor based upon how far away the reported metrics were from the spatial and temporal centroids. The computation of the suspicion value also takes into account each node locally sharing information with the other nodes about its suspects list.

$$q_{ij} = \begin{cases} q_{ij} + \alpha(\frac{|U_j - \overline{U}|}{\sigma_U}) + \beta(\frac{|V_j - \overline{V}|}{\sigma_V}) + N_r & \text{if } j \text{ is outlier} \\ \frac{3q_{ij}}{4} - 1 & \text{otherwise} \end{cases} \quad (4)$$

The suspects list is updated at the end of each probe cycle after the system has performed outlier detection. Equation 4 presents the computation of a single suspicion value during a probe cycle: q_{ij} is the suspicion value of node i for peer j, U is the list containing the Mahalanobis distances measuring spatial distance for i's peers, V is the list containing the temporal distance for i's peers, $\overline{U}$ and $\overline{V}$ are the averages of each list, σ_U and σ_V are the standard deviations of U and V respectively, and N_r is a counter representing how many other nodes reported j as suspicious. Each measure is weighted independently (α, β, and γ) to allow the response to be tailored to the application or network conditions. Intuitively, our scheme assigns suspicion such that the greater the distance between the observed data for a node and the centroid of the entire data set, the greater the local suspicion value a node is assigned. We also assign more weight to the direct observations (i.e. how far a node's distance is from the centroid) than to indirect observation (i.e. how many nodes reported a node as suspicious). The suspicion value is increased every probe cycle if the probed node is an outlier. Otherwise, as seen in the second part of Equation 4, the suspicion values undergoes decay to accommodate transient network conditions and allow nodes to be removed from the suspects list, eventually enabling a node that behaves well to be reconsidered as a parent.

Once a node is placed on the suspects list, it can be displaced as a child node, it will not be chosen as a parent, its gossiped information will not be propagated, and it may eventually be reported to the source as being *malicious*. A node will decide if it considers a neighbor node *malicious* by comparing the suspicion value against a threshold, Δ. If the value is higher than the threshold, then the node is reported as malicious to the other nodes and the source.

Malicious nodes may collude and send false information causing non-malicious nodes to incorrectly suspect their peers. To prevent this, even if node i has a positive suspicion value for node j, i will not report a negative reputation for j to the source unless i has directly experienced suspicious/malicious behavior from j. This approach still allows honest nodes to build up a strong negative reputation through indirect observations, but holds global-response at bay until the malicious node directly treats the honest node badly. Even if a node is marked as suspicious, it will still receive service from the overlay and will remain a member of the topology. In this manner an honest node cannot be isolated from the overlay unless *every* overlay node peer it attempts to use as a parent is malicious and drops all traffic to the honest node.

2) Global Response: Our global response mechanism creates a global representation of the trust in each node in the overlay by using a reputation system to aggregate the individual suspicion values from the local suspects list at the source. The nodes with a trust value below a specified threshold are added to a global black list disseminated to all nodes. We adapt a well-known distributed reputation system, EigenTrust [36], to the trust model of our application in which the source node is trusted. We selected the EigenTrust algorithm because its trust value aggregation method is robust to malicious nodes and coalitions. We make several modifications to the EigenTrust algorithm to tailor it to our application.

The intuition behind EigenTrust is that a node i forms a broader trust value (t_{ik}) in node k by asking its neighbors to report their trust in k and weighting those opinions by i's trust in its neighbor: $t_{ik} = \sum_j c_{ij} c_{jk}$, where c_{ij} is the normalized local trust value of node i in node j. The value c_{ij} is calculated as follows:

$$c_{ij} = \begin{cases} \frac{max(s_{ij},0)}{\sum_j max(s_{ij},0)} & \text{if } \sum_j max(s_{ij},0) > 0 \\ p_j & \text{otherwise} \end{cases} \tag{5}$$

where s_{ij} represents the non-normalized local trust value at node i in node j and p_j represents the default trust value for the node j. As the only trusted node for our application is the source, $p_j = 1$ if the node is the source and $p_j = 0$, otherwise. A node will have a local trust value of zero with all nodes it has not interacted with and initially only trust the source. When a node i first interacts with another node j, it forms a new non-zero local trust value, s_{ij}, based on the "goodness" of the interaction.

Algorithm 1: Basic EigenTrust Algorithm

```
$\vec{t}^0 = \vec{p}$;
while $\delta > \epsilon$ do
    $\vec{t}^{k+1} = C^T \vec{t}^k$;
    $\vec{t}^{k+1} = (1-\lambda)\vec{t}^{k+1} + \lambda\vec{p}$;
    $\delta = \left\|\vec{t}^{(k+1)} - \vec{t}^{(k)}\right\|$
end
```

Using the basic EigenTrust Algorithm presented in [36] and reproduced in Algorithm 1, the idea of transitive trust can be extended to formulate a system wide trust ranking by formulating the summation of local trust values as a matrix multiplication. Through each iteration of multiplying the global trust vector $\vec{t}$ by the aggregated local trust values contained in C, the algorithm intuitively represents asking successively further nodes opinions of their neighbors. After each iteration, each of the trust values stored at source in $\vec{t}$ is normalized to guarantee that meaningful comparisons between values can be performed, but not that all trust values add up to one. The calculation continues until the convergence of $\delta = \left\|\vec{t}^{(k+1)} - \vec{t}^{(k)}\right\| < \epsilon$, where ϵ is empirically set to .0001. To guarantee that the calculation will converge, the pre-trusted nodes trust vector, $\vec{p}$, is used as the starting vector ($t^0 = \vec{p}$). To mitigate the effects of malicious coalitions of nodes cooperating to subvert the reputation system, the pre-trusted nodes are favored with a certain weight, λ, after each iteration. Since the source is the only trusted node in the system, it is the only node to start out with a positive reputation. The source's pre-trusted weight, λ, was empirically set at 0.3 to minimize the convergence time of the EigenTrust algorithm while still providing valuable feedback. Once the trust values drop below a specified threshold, Ψ, the system will consider the identified node as being malicious.

In addition to tailoring the EigenTrust algorithm to our environment, we also take into account that while EigenTrust was designed to use and create positive reputation, our outlier detection produces only negative reputation about a node. We convert the local suspicion values into a positive form suitable for the EigenTrust algorithm.

V. Experimental Results

We demonstrate through experimental results the identified attacks and our outlier detection and response techniques in the context of the ESM overlay multicast system. We selected ESM because of its maturity, extensive deployment, and the advanced set of adaptation techniques it employs. Our experiments show that, although ESM employs an advanced set of adaptation mechanisms, it is unable to mitigate the attacks posed by a malicious adversary. Our outlier detection and response mechanisms were able to reduce the impact of malicious nodes without adding to the link stress in the system or requiring significant storage.

A. Overview of ESM

ESM [4] is a multicast system mainly used for broadcasting live events such as academic conferences. We provide a high-level description below. For further details, the reader is referred to [29]. ESM forms a peer-to-peer overlay tree for distributing multicast content. A node changes its parent in the overlay to maintain and improve application performance. Both passive observation and probing are used to collect data used to make the adaptation decision. ESM uses data sampling and data smoothing to address variations in the metrics considered: available bandwidth, latency, and RTT. ESM also employs a number of combined metrics, damping, randomization, hysteresis and three utility functions to address instabilities in the observed data. The three utility functions are based on: bandwidth, latency, and a combination of bandwidth and latency. A damping factor is used to induce stability and a randomization technique is used to avoid the case where several nodes try to change to the same parent.

The parent selection algorithm is presented in Algorithm 2. In order to select a new parent, a node first computes a list of potential candidates from its neighbor set. Nodes which are currently saturated, descendants, or did not respond when recently probed are not considered. If there is no utility gain, no node is selected and the process will be repeated next cycle. If several nodes are candidates, then the first candidate is selected as the new parent. The selection process uses hysteresis to generate a negative bias against nodes that have performed poorly in the past.

B. Testbed and Experiment Setup

To study the attacks and defense mechanisms under real-world conditions, we conducted our experiments on the PlanetLab [16] Internet testbed. In addition, for repulsion and disruption attacks that could have been disruptive to PlanetLab, we used DETER [17], a testbed that provides a stable, controllable emulation environment for network security research.

Algorithm 2: Parent Selection Algorithm Used in ESM

```
   Input: Set of probe responses tuples (<bandwidth, latency, RTT>)
   Output: New Parent Node
1  Create list of potential parent candidates, $PCL$, excluding nodes: 1)
   currently saturated, 2) descendant of this node, and 3) unresponsive
2  foreach $rnode$ in $PCL$ do
       // Combined Metrics Component
3      if (utilityGain(throughput, latency) $\geq$ currentUtility*1.1) then
4          keep $rnode$ in $PCL$;
5      else
6          remove $rnode$ from $PCL$;
7      end
8  end
9  foreach $rnode$ in $PCL$ do
       // Hysteresis Component
10     if (rnode has performed well in the past) then
           // Dampening Component
11         if (local node has not switched parents recently) then
               // Randomization Component
12             if (node switch probability $>$ rand()) then
13                 Select $rnode$ as next parent;
14                 return $rnode$
15             end
16         end
17     end
18 end
19 return no change
```

The PlanetLab Internet testbed is explicitly designed to provide a research platform for large scale distributed experimentation of peer-to-peer systems [37]. In order to mitigate the possible limitations of using a testbed, such as those addressed in [37], experiments were run at different times of the day and different days of the week, nodes were randomly chosen to validate the statistical significance of results, and sets of nodes were chosen which spanned multiple operational and administrative domains. We use 60 minute long ESM deployments of 100 nodes in which the nodes join after the experiment begins and leave before it ends, with an average participation time of fifty-five minutes. As in previous ESM deployments [38], nodes are probed every seven seconds, the saturation degree of benign nodes is six, and the source constant bit rate is 480 Kbps. All experiments use these parameters unless otherwise noted.

C. Attack Effectiveness

1) Attraction Attacks: We demonstrate the effect that a single coalition of one malicious node, who exploits the adaptive nature of ESM, has on the multicast tree construction, maintenance, and stability. One randomly selected node performs an attraction attack in which it lies every probe cycle about having the best bandwidth (480Kbps), latency (0ms), and no saturation. When the node is honest, it is selected only 5 times as a parent by other nodes. However, when the node is malicious, it is selected 172 times, or almost 35 times more often. The malicious node also causes the overlay to become more unstable as can be seen in the 24% increase in 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 its parent again.

We next consider the effect on the benign nodes when a percentage of randomly selected malicious nodes perform attraction attacks. Metrics we investigate are: the percentage of nodes that have at least one malicious node on their path from the source, the percentage of nodes that have a malicious node as a parent at some point during the experiment, and the number of parent change decisions that resulted in selecting a malicious node. The results of the experiment, summarized in Fig. 1, demonstrate that even a small percentage of malicious nodes will affect the majority of benign nodes in the overlay. Fluctuations in the general trends of the curves result from the use of real-world experimentation and randomly selected malicious nodes. The greater the number of malicious nodes located near the source in the overlay topology, the greater the effect will be on the overall system.

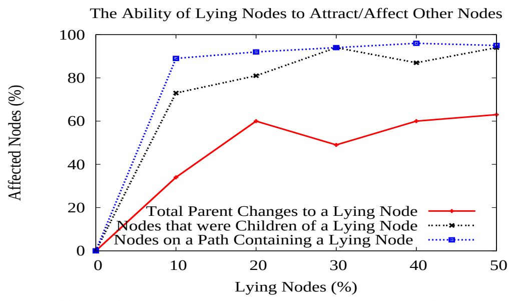


Fig. 1. The effect of attraction attacks on benign nodes for an ESM overlay of 100 nodes on PlanetLab for a duration of 60 minutes.

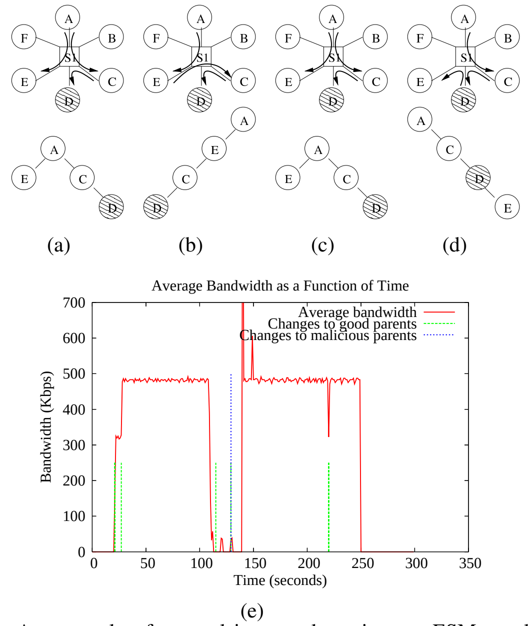


Fig. 2. An example of a repulsion attack against an ESM overlay in a controlled experiment on DETER. (a) represents the overlay and the multicast tree before attack while (b), (c) and (d) are the topology changes in the multicast tree as a result of the attack. Node E is manipulated by the attacker to attach to malicious node D, although this causes E to be three hops away from the source, instead of just one. (e) The average bandwidth with topology changes denoted by the solid and dashed impulses representing good and malicious parent changes respectively.

2) Repulsion Attacks: While performing experiments, we noticed that nodes with very good performance, such as those directly attached to the source, could not be easily fooled by lying nodes. We demonstrate a repulsion attack where an attacker affects the partially observable link state estimation in order to make a node incorrectly believe that the performance from the current parent is inadequate. While there are many

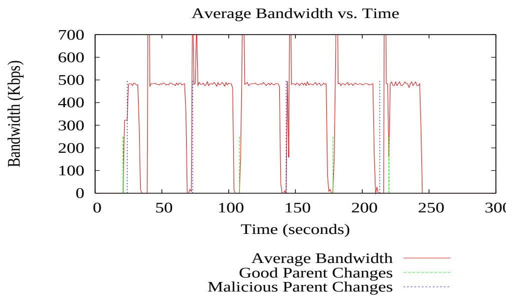


Fig. 3. An example of a disruption attack against an ESM overlay in a controlled experiment on DETER. The experiment was performed using the same experimental setup as Fig. 2. The attackers periodically sent 5 second bursts of traffic at the focal point S1, creating constant churn in the system. Topology changes are denoted by the solid and dashed impulses representing good and malicious parent changes respectively.

possible attack methodologies, this method of attack was chosen for demonstrative purposes because it has been used against other types of adaptative protocols [15], [14], [19].

Fig. 2 presents a star topology composed of six nodes, all of which are connected with 100 Mbps links to switch S1. For demonstrative purposes, ESM is configured to use a saturation degree of two. In our example, node A is the source and nodes B through F are end-systems in the overlay. Nodes B and F collude with D. During the attack, nodes B and F generate traffic to augment the attack of malicious node D which lies about its bandwidth (480Kbps), latency (0ms), and saturation (none). The traffic generated by nodes B and F causes the malicious node D to appear far better than the nodes burdened by the extraneous traffic.

The overlay initially converges to the stable structure seen in Fig. 2(a), at which point the mean bandwidth is approximately 480 Kbps. Topology changes occur at the impulses seen in Fig. 2(e). The attack begins at 115 seconds when nodes B and F begin flooding for 30 seconds towards the source, node A. The attack is able to generate several changes in the tree, with the final result that node E detaches from the source and, instead of choosing node C, chooses the malicious node D, as its parent in Fig. 2(d). Note that node E was previously directly connected to the source but it is now connected three hops away. The changes after 200 seconds are due to nodes leaving the experiment.

The effort incurred to cause such an attack consists of saturating the 100 Mbps link with a short 30 second burst of traffic. In real Internet deployments, this will be substantially less since links will typically have a lower capacity.

3) Disruption Attacks: Fig. 3 demonstrates an example of a disruption attack where the attacker exerts an artificial influence (extraneous traffic) towards the switch S1 in the overlay topology. The switch S1 is a focal point in the overlay topology, handling all of the network traffic for the targeted section of the network. The main difference from previous attacks is that the artificial influence is done periodically in order to destabilize the infrastructure. In the experiment in Fig. 3, the attacker sends 5 second bursts of traffic every 30 seconds. It should be noted that the attack actually targets the medium a node shares with its parent, making it difficult to detect through traditional methods. This is similar to the attacks performed in [14], [15] which targeted the TCP congestion control. Fig. 3 shows that by using this technique, the attacker can keep the system in a constant churn as it keeps trying to stabilize itself. Despite the fact that the attacker was using only 5 second bursts of traffic, parent changes occurred in the overlay at almost every probe cycle.

D. Effect of Malicious Nodes on Average Bandwidth

We studied the effect multiple malicious nodes can have on the overlay topology if they decide to selectively drop data. In Fig. 4, we demonstrate the impact malicious nodes that use their position in the tree can exert on the bandwidth of benign nodes. The graphs plot the bandwidth averaged over all receivers as a function of time. Malicious nodes start dropping 100% of the data traffic received through the data dissemination tree fifteen minutes after they joined the overlay. We vary the percentage of malicious nodes to 10%, 30%, and 50% of the overlay size to demonstrate the performance degradation that results when more nodes behave maliciously.

We define the relative strength of a particular attack as:

$$\tau = \frac{B_{norm} - B_{adv}}{B_{norm} \times N_{adv}} \qquad (6)$$

where B_{norm} and B_{adv} represent the average throughput in the absence and presence of adversaries respectively, and N_{adv} is the number of adversaries. Intuitively, τ represents the amount of damage an attack created in the system. The greater the performance degradation observed in the system between when the malicious nodes are passive and active (the difference between B_{norm} and B_{adv}), the higher the value of τ and the more damage an attack inflicts on the overlay.

Fig. 5 depicts τ varying over the percentage of the traffic dropped. As it can be seen, the greater the amount of data traffic a malicious node drops, the greater the effect it has on the system. The drop in the effectiveness of the attacks as the malicious nodes drop high percentages of data (100%) is due to ESM categorizing the malicious nodes as unstable links based on past experienced bandwidth and having a bias against choosing them as parents. Fig. 5 also shows the intuitive notion that the greater the number of malicious nodes, the greater effect there is on the system. It can be noted that just 10% malicious nodes have a significant effect on the average bandwidth. We believe this is because a percentage of 10% malicious nodes is enough to obtain advantageous positions in the vulnerable tree structure which has no path redundancy.

E. Effectiveness of Outlier Detection

To demonstrate the effectiveness of our outlier detection at improving the parent selection process and the stability of the system, we considered one malicious attacker and recorded the number of parent changes that took place for the duration of the experiment considering two cases, one when only the spatial outlier is used, and one when the temporal-spatial outlier is enabled. The outcome of these experiments is shown in Table I. The results indicate that using the spatial outlier detection scheme has dramatically reduced the likelihood of choosing a malicious parent since the number of times the malicious node was selected as a new parent is reduced from 172 to 70. The addition of the temporal outlier detection further reduces this to only 35 times.

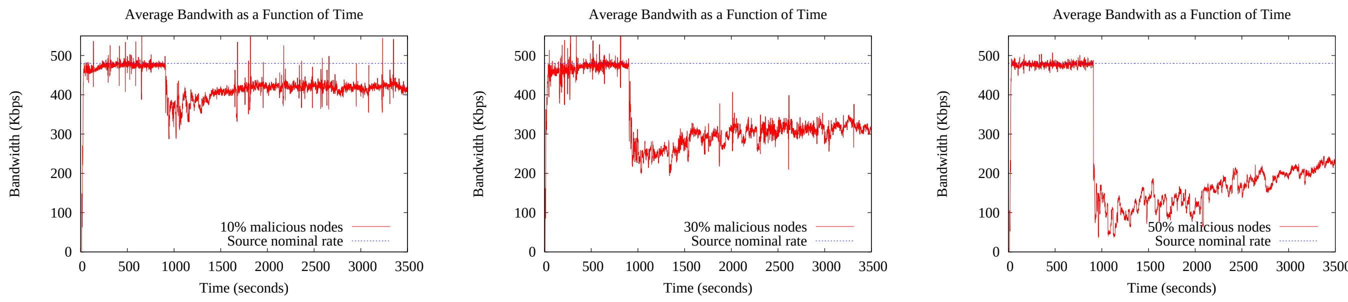


Fig. 4. The average bandwidth over time for an ESM overlay of 100 nodes on PlanetLab for a duration of 60 minutes with different percentages of malicious nodes.

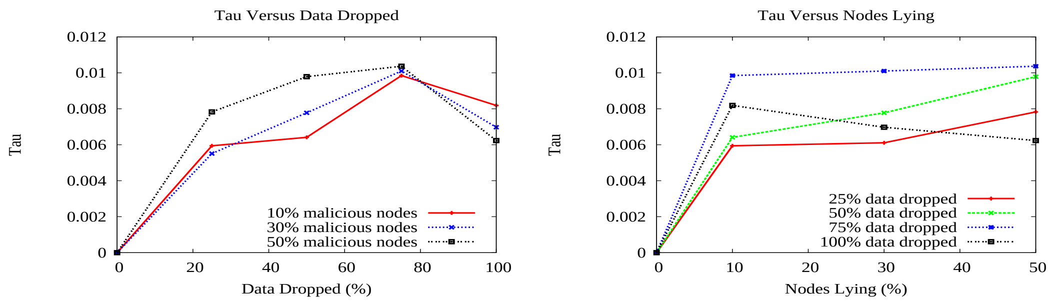


Fig. 5. τ as a function of (a) the percentage of data dropped and (b) the percentage of malicious nodes for an ESM overlay of 100 nodes on PlanetLab for a duration of 60 minutes.

TABLE I
THE EFFECTIVENESS OF OUTLIER DETECTION AT IMPROVING PARENT SELECTION FOR AN ESM OVERLAY OF 100 NODES ON PLANETLAB OVER 60 MINUTES

Experiment	Changes to Malicious Parents	Total Parent Changes
No lying	5	833
Lying	172	1032
Spatial	70	800
Spatial/Temp	35	604

Our method also dramatically improved the stability of the overlay in spite of the presence of the malicious node, as measured by the decrease in total parent changes denoted in third column of Table I. In fact, the number of adaptations is comparable to the number of adaptations that would occur when no malicious nodes are present in the overlay.

F. Coalitions of Attackers and Spatial Outlier Detection

The previous experiment demonstrated the effectiveness of the spatial correlation for detecting outliers produced by a single coalition containing one attacker. We now consider the constrained collusion model presented in Section II in which all faulty nodes are part of the same coalition. A coalition of colluding attackers may attempt to bypass the outlier detection mechanism itself by shifting the centroid so they are not perceived as outliers anymore. As a result, one of the members of the malicious coalition will be selected as the parent.

We considered three colluding cases requiring different degrees of coordination between the attackers. In the first case, referred to as “Optimum BW, Latency”, the malicious nodes can only lie about the latency and bandwidth and not the RTT. The second case, referred to as “Optimum BW, Latency, RTT”, the malicious nodes agree to lie consistently on a set of predefined values: RTT of 0, latency of 0, and bandwidth

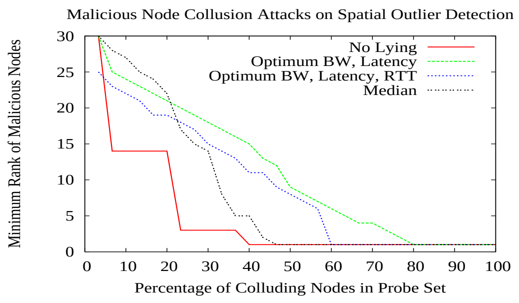


Fig. 6. Number of colluding nodes necessary to influence parent selection. The goal is for a colluding node to be ranked first and chosen as a parent.

of 480 Kbps. Note that in order to influence the RTT, this case requires that one malicious node indeed has an RTT of 0 with the victim and it can intercept all the RTTs of the other nodes in the coalition. The third case, referred to as “Mean”, assumes that the attackers have the ability to share their observed performance and compute and report the average of their real metrics, again only bandwidth and latency. This case requires strong coordination between the attackers, which may not always be possible during a probe cycle without creating inconsistencies in measured probe times. We compare these cases with the normal case, when no nodes are lying.

We summarize our findings for an ESM overlay of 118 nodes on PlanetLab in Fig. 6. This graph depicts the highest rank of a member of the malicious coalition as a function of the percentage of malicious nodes. Note that it took a malicious coalition of 80% of the nodes in a probe set in the first case and 60% of the nodes in a probe set in the second case before a malicious node is chosen as the next parent. This demonstrates the effectiveness of the spatial outlier detection since both the number and type of metrics used by the outlier

TABLE II
SYSTEM SETTINGS FOR THE RESPONSE MECHANISM

Variable	Description	Value
α	Spatial (Horizontal) Outlier Weighting	10
β	Temporal (Vertical) Outlier Weighting	7
γ	Gossip Response Weighting	1
Δ	Local Reporting Threshold	14
Ψ	Global Trust Value Threshold	Variable

detection defense make it difficult for the attackers to maintain consistency. In the "Mean" case, 47% of the nodes needed to be in a coalition before they could deterministically guarantee that a malicious node would be 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. When compared with the normal case in which no node exhibits malicious behavior, the "coalition" would only need to contain 40% of the nodes. Thus, lying about metrics, even with sophisticated coordination techniques, is no longer an effective attack technique. The spatial outlier technique we describe constrains the behavior of attackers and reduces their ability to artificially augment their influence on the system.

G. Effectiveness of Response Mechanisms

Algorithm 3: Procedure to exclude malicious nodes as possible parents. The code is invoked after line 8 in the ESM parent selection pseudo-code presented in Algorithm 2

```
Input: Potential Parent Candidates List ($PCL$)
Output: Updated $PCL$
1  foreach $rnode$ in $PCL$ do
       // Ignore Known Malicious Nodes
       // Resulted from Global Response
2      if (rnode is on BlackList) then
3          remove $rnode$ from $PCL$;
4      else
5          keep $rnode$ in $PCL$;
6      end
       // Detect Malicious Behavior
       // Resulted from Local Response
7      if (outlierDetection(rnode) == false) then
8          keep $rnode$ in $PCL$;
9      else
10         remove $rnode$ from $PCL$;
11     end
12 end
```

To demonstrate the effectiveness of our response mechanisms at mitigating the effects of malicious nodes and sustaining the average bandwidth of the system, we conducted experiments in which a percentage of the nodes were malicious and recorded the average bandwidth for the duration of the experiment. The system was using both spatial and temporal outlier detection to generate the local suspicion values. The additional steps which occur during the the parent selection process to mitigate the effects of malicious nodes are presented in Algorithm 3, which is executed between lines 8 and 9 of the original parent selection algorithm presented in Algorithm 2.

Fig. 7(a) and Fig. 7(b) present results for the local and global response mechanisms when 30% of the nodes are malicious. Each malicious node joins the network and lies about having the best bandwidth (480Kbps), latency (0ms), and no saturation. Once the malicious nodes have had a chance to optimize their position in the overlay, fifteen minutes after they joined the overlay, they start dropping 90% of the data traffic received through the data dissemination tree. We also present as a reference the average bandwidth under non-attack conditions, with the response mechanisms enabled in Fig. 7(c). Fig. 7(a) shows that using only a local response does decrease the effect of the malicious nodes. However, the average bandwidth of the system converges to a value below the one obtained in the absence of malicious nodes and the system takes longer to stabilize. Next, we explored the effect of the global response mechanism on the bandwidth and system stability. Fig. 7(b) demonstrates that the addition of the global response mechanism further decreases the effect of the malicious nodes, bringing the average bandwidth of the system close to the value when no malicious nodes exist in the system. When using only the local response, over one third of the identified malicious nodes were actually false positives. By using the global reputation system, we were able to reduce the number of false positives to zero. We also evaluated the use of only the global response mechanism, which resulted in performance similar to the combination of the local and global response mechanisms but showed greater variation in bandwidth. Due to lack of space, we present the results for the combination of both.

The system settings for the response mechanism can be found in Table II. Intuitively, the settings were designed to place more weight on being consistent with the current state of the system as measured by the spatial outlier while allowing nodes to have small inconsistencies in reported metrics and not be considered malicious. Each value was determined empirically through experimentation. As shown in Fig. 7(b), even a very conservative response mechanism greatly increases the resiliency of the network to a large percentage of malicious attackers. While the conservative approach was only able to black list approximately one third of the malicious nodes (no non-malicious nodes were black listed), the remaining malicious nodes were pushed towards the fringes of the overlay, allowing the system to sustain near-optimal bandwidth. As the response mechanism is tuned to the application and network conditions, it is able to identify and quarantine higher percentages of malicious nodes. It was identified during testing that the local suspects list could be optimized to only include nodes that had been considered as a parent. This allowed each node to track a much smaller set of suspicious nodes and resulted in a homogenous set of suspicion values at the source. We were able to set cutoff thresholds much tighter without black listing poor performing non-malicious nodes, thereby improving the effectiveness of our response mechanism.

Fig. 8 presents the effect of the response mechanisms on the relative strength of the attacks, τ, as defined in Eqn. 6. It confirms the intuition that when the response mechanisms act quickly, the strength of the attack will be diminished as the malicious nodes are more quickly eliminated from the list of potential parents. As expected, the local response mechanism can partially mitigate the effects of the attackers, while the faster convergence of the global mechanism results in a smaller damage on the overall system. The higher the cut-off threshold, the smaller the damage on the system.

Fig. 9 presents the effectiveness of the response mechanisms

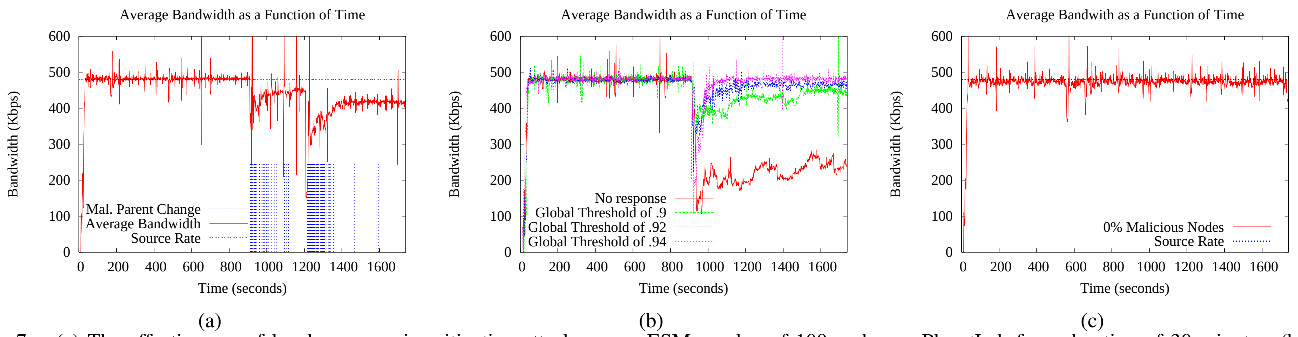


Fig. 7. (a) The effectiveness of local response in mitigating attacks on an ESM overlay of 100 nodes on PlanetLab for a duration of 30 minutes. (b) The effectiveness of global response in mitigating attacks on an ESM overlay of 100 nodes on PlanetLab for a duration of 30 minutes. (c) The average bandwidth over time for an ESM overlay of 100 nodes on PlanetLab for a duration of 30 using both response mechanisms under non-attack conditions.

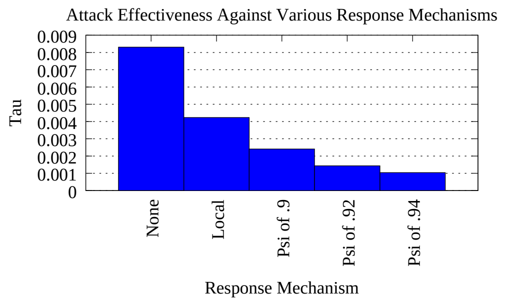


Fig. 8. Attack effectiveness against different response mechanisms on an ESM overlay of 100 nodes. Tau represents the amount of damage an attack created in the system as described in Section V-D

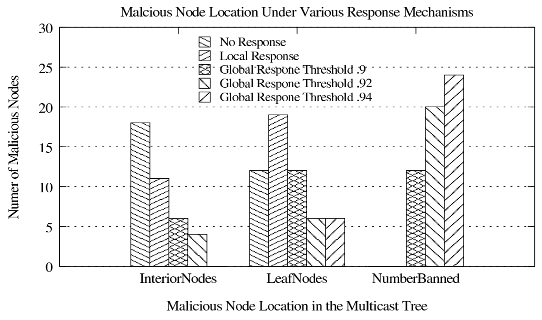


Fig. 9. Malicious node location for an ESM overlay of 100 nodes with 30% malicious nodes on PlanetLab under different response mechanisms

at moving the malicious nodes towards the fringes of the tree to locations of less influence. Without a response mechanism, the majority of malicious nodes (18 out of 30) occupy interior positions in the multicast tree, with the rest being leaves in the tree. The greater the number of interior positions the malicious nodes control, the greater the effect they will have on the system performance since they can affect all nodes downstream of them. By utilizing the local response only, the system is able to push more of the malicious nodes towards the fringe of the network, with only 11 being interior nodes and 19 now being leaf nodes. Note that the local response does not ban nodes from the network since they can only be banned when the global mechanism is activated. With the addition of the global response and the ability to remove malicious nodes from the network, the system is more effective in responding to the malicious nodes. For example, with a global response threshold of .92, only 4 malicious nodes are interior nodes, 6 nodes are leaf nodes, and the remaining 20 are removed from the network entirely. By removing malicious nodes from locations of influence, our solution is able to maintain the performance of the system.

H. Overhead and System Performance

Our outlier detection does not introduce any extra link stress since it uses information that is already being exchanged between nodes. 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 probed nodes, while the storage requirements consist of three additional values in the route table for the peer set maintained by each node. In the case of the temporal outlier detection, the memory usage consists of maintaining the temporal centroid. By incrementally updating the centroid, nodes do not need to maintain the entire history for each probed node. The temporal outlier detection also requires modifying the route table entries to store nine additional values: mean, standard deviation, and count for each of the three metrics.

Our response mechanisms introduce minimal link stress since the reputation information for each individual node is combined with the pre-existing membership protocol. Additional messages are required to disseminate the global black list to member nodes through the multicast tree and suspect information from each member node to the source. On average, the source receives an extra $83(\frac{N}{t})$ Bytes of network traffic per second, where N is the size of the overlay and t is the reporting interval. In our experiments, t was set to 20 seconds, resulting in the source receiving approximately 413 Bytes of suspect data per second. Every node in the tree receives $4B$ Bytes of black list information every t seconds, where B is the number of nodes on the black list. The memory utilization per node is up to $8N$ Bytes for the suspects list and up to $4N$ for the black list. The difference is due to each node storing a local reputation for every other node on the suspects list.

VI. Related Work

Our work focuses on attacks exploiting measurement-based adaptation in overlay networks and our solution uses concepts borrowed from anomaly detection and reputation systems. Below we review work in several areas related to our research.

Attacks exploiting adaptivity. Previous work showed the vulnerability of the TCP adaptation mechanisms, i.e. the congestion control mechanism, to malicious attacks [14]. 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. The attack was generalized in [15], as a form of low-rate ROQ attack targeting point-to-point adaptive control loops that drive resource allocation and affect perceived service of a system (bandwidth, jitter, etc). Our work assumes a stronger adversarial model in an overlay network. The nature of the attacks, application and deployment environment allows us to use a context sensitive observation space and correlated information associated with the same information that drives the adaptation to detect and limit the effect of malicious behavior.

Anomaly detection and Mahalanobis distance. Recently the benefits of the Mahalanobis distance for statistical anomaly detection have been demonstrated in the context of network intrusion detection [31], [39]. In [39] the authors present a comparative study of detection schemes based on data mining techniques for network based intrusion detection. In [31] the authors discuss an unsupervised, payload-based network anomaly detector based on the Mahalanobis distance which was used to detect attacks like worms.

Use of spatial and temporal correlations. Spatial and temporal correlations were previously used in the context of network security. A notable work in this aspect is [30] where the authors use temporal and spatial correlations to trace back attacks and detect attack scenarios, using a large amount of information from intrusion detection systems, firewalls, and different software logs. Unlike the approach in [30], 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.

Correlations have also been used in sensor network and ad-hoc networks for the detection of malicious nodes [40], [41]. Most of this research focused on the evaluation of off-line data developed in a simulator. In our work, the correlation is actually incorporated in-line with the protocol as it tries to adapt. Analysis is performed on the Internet with real data while fusing multiple correlations to improve our predictive abilities. The work in [41] shows how to augment a sensor network with spatio-temporal correlation to detect misinformation being injected into the sensor streams. In our research, we are concerned with an attacker manipulating the control information in order to influence system adaptation.

Malicious behavior in overlay networks. The problem of malicious attackers was previously studied in the context of structured overlay networks. A subset of these types of attacks, referred to as Eclipse attacks [12], [13], was subsequently studied in optimized structured file sharing overlays. The solution enforces degree constraint invariants associated with neighbors, supported by anonymous auditing, and takes advantage of strong organizational neighbor constraints existent in such networks. As unstructured overlay networks do not have such constraints, the proposed solutions are not applicable.

To the best of our knowledge, the problem of malicious insider attacks was not studied in the context of unstructured overlay networks. An attack performed by selfish attackers (i.e. nodes that want to obtain an advantage but do not have destructive goals) was shown through simulations in [42]. Our work is different in the fact that it considers malicious attackers and presents results in the context of a real system in real deployments over the Internet.

Reputation systems. A considerable amount of research has focused on the development of trust and reputation systems for peer-to-peer systems [43], [36] and ad hoc networks [44], which help users make beneficial decisions assuming the existence of detection mechanisms for malicious behavior. It has been shown in [45], [46], [47] that these systems provide an effective way to mitigate the effects of malicious nodes in a decentralized distributed system. In [48], the authors provide techniques for improving the security of reputation systems which could be used in conjunction with our approach. Our work presents concrete solutions for the detection mechanisms that reputation can be built on.

To the best of our knowledge, our work is one of the few implementations of a reputation system in overlay applications that has been tested in a operational system over the Internet and not just simulated. Credence [49] is one other system that we are aware of which uses a reputation system to deal with the file pollution problem in a file-sharing application. Unlike Credence, we focus on multicast applications and consider malicious insiders. Although our work has the advantage of having a point of trust (i.e. the source), it faces the challenge that it can not rely on human interaction to generate reputation. Instead, it achieves its goals in an autonomic fashion.

VII. Conclusions

In this paper we identified insider attacks that exploit measurement-based adaptation mechanisms in multicast overlay networks. We discussed a comprehensive defense framework and presented an in-depth solution to two critical aspects of the problem: preventing poor adaptation decisions and taking corrective measures to isolate and minimize the effect of the identified attackers. We showed the importance of tightly coupling the detection space and the control space. By incorporating context sensitive anomaly detection into the protocol, our detection mechanisms had the semantic understanding to improve the adaptive decision process. We demonstrated the effectiveness of the newly identified attacks and the benefits of using our detection and response mechanisms in the context of ESM, a well-known adaptive multicast overlay network. Our experiments conducted in real-life deployments and emulations, showed that although ESM employs an advanced set of adaptation mechanisms it was unable to mitigate the attacks posed by a malicious adversary. Our experiments demonstrated that our techniques improved the adaptation process and the overall stability of the system while limiting the effect of malicious nodes.

References

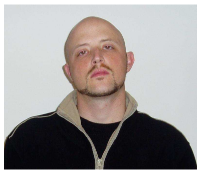
[1] M. Castro, M. Costa, and A. Rowstron, “Should we build Gnutella on a structured overlay?,” in *HotNets*, 2003.

[2] M. Castro, P. Druschel, A. Kermarrec, and A. Rowstron, “SCRIBE: A large-scale and decentralized application-level multicast infrastructure,” *IEEE Journal on Selected Areas in communications (JSAC)*, vol. 20, no. 8, pp. 1489–1499, 2002.

[3] M. Castro, P. Druschel, A. Kermarrec, A. Nandi, A. Rowstron, and A. Singh, “SplitStream: high-bandwidth multicast in cooperative environments,” in *Proc. of SOSP*, 2003.

[4] Y. Chu, S. G. Rao, and H. Zhang, “A case for end system multicast (keynote address),” in *SIGMETRICS*, 2000.

[5] S. Banerjee, B. Bhattacharjee, and C. Kommareddy, "Scalable application layer multicast," in *SIGCOMM*, 2002.
[6] J. Jannotti, D. K. Gifford, K. L. Johnson, M. F. Kaashoek, and J. W. O'Toole, Jr., "Overcast: Reliable multicasting with an overlay network," in *USENIX OSDI*, 2000.
[7] "2005 e-crime watch survey - survey results." http://www.cert.org/archive/pdf/ecrimesurvey05.pdf.
[8] D. S. Wallach, "A survey of peer-to-peer security issues," in *International Symposium on Software Security*, 2002.
[9] W. J. Bolosky, J. R. Douceur, D. Ely, and M. Theimer, "Feasibility of a serverless distributed file system deployed on an existing set of desktop PCs," in *SIGMETRICS*, 2000.
[10] E. Sit and R. Morris, "Security considerations for peer-to-peer distributed hash tables," in *IPTPS*, 2002.
[11] M. Castro, P. Druschel, A. Ganesh, A. Rowstron, and D. S. Wallach, "Secure routing for structured peer-to-peer overlay networks," in *USENIX OSDI*, 2002.
[12] A. Singh, M. Castro, A. Rowstron, and P. Druschel, "Defending against eclipse attacks on overlay networks," in *ACM SIGOPS European Workshop*, 2004.
[13] A. Singh, T.-W. Ngan, P. Druschel, and D. Wallach, "Eclipse attacks on overlay networks: Threats and defenses," in *INFOCOM*, 2006.
[14] A. Kuzmanovic and E. W. Knightly, "Low-rate TCP-targeted DOS attacks: the shrew vs. the mice and elephants," in *SIGCOMM*, 2003.
[15] M. Guirguis, A. Bestavros, and I. Matta, "Exploiting the transients of adaptation for RoQ attacks on internet resources," in *ICNP*, 2004.
[16] "Planetlab." http://www.planet-lab.org/.
[17] "Deter." http://www.isi.edu/deter/.
[18] J. F. Nash, "The Bargain Problem," *Econometrica*, vol. 18, pp. 155–162, 1950.
[19] D. G. Andersen, "Resilient overlay networks," Master's thesis, Department of EECS, MIT, 2001. http://nms.lcs.mit.edu/projects/ron/.
[20] D. Bauer, S. Rooney, P. Scotton, S. Buchegger, and I. Iliadis, "The performance of measurement-based overlay networks," in *QofIS*, p. 2002.
[21] Y. Chu, S. G. Rao, S. Seshan, and H. Zhang, "Enabling conferencing applications on the internet using an overlay multicast architecture," in *ACM SIGCOMM*, 2001.
[22] Y. Rekhter and T. Li, "A Border Gateway Protocol 4 (BGP 4)." Internet Engineering Task Force: RFC 1771, March 1995.
[23] C. Labovitz, A. Ahuja, A. Bose, and F. Jahanian, "Delayed internet routing convergence," *IEEE/ACM Trans. Netw.*, vol. 9, no. 3, pp. 293–306, 2001.
[24] Z. M. Mao, R. Govindan, G. Varghese, and R. H. Katz, "Route flap damping exacerbates internet routing convergence," in *SIGCOMM*, 2002.
[25] M. Seshadri and R. H. Katz, "Dynamics of simultaneous overlay network routing," Tech. Rep. UCB//CSD-03-1291, University of California, Berkeley, November 2003.
[26] C. Tang and C. Ward, "GoCast: Gossip-enhanced overlay multicast for fast and dependable group communication," in *DSN*, 2005.
[27] B. Y. Zhao, L. Huang, J. D. Kubiatowicz, and A. D. Joseph, "Exploiting routing redundancy using a wide-area overlay," Tech. Rep. CSD-02-1215, U. C. Berkeley, Nov 2002.
[28] A. Walters, D. Zage, and C. Nita-Rotaru, "Mitigating attacks against measurement-based adaptation mechanisms in unstructured multicast overlay networks," in *ICNP*, 2006.
[29] A. Walters, "Mitigating attacks against adaptation mechanisms in overlay networks," Master's thesis, Purdue University, 2006. http://projects.cerias.purdue.edu/ds2/papers/aaron_walters_thesis.pdf.
[30] G. Jiang and G. Cybenko, "Temporal and spatial distributed event correlation for network security," in *American Control Conference*, 2004.
[31] K. Wang and S. J. Stolfo, "Anomalous Payload-based Network Intrusion Detection," in *RAID*, 2004.
[32] C. Lu, D. Chen, and Y. Kou, "Multivariate spatial outlier detection," *International Journal on Artificial Intelligence Tools, World Scientific*, vol. 13, pp. 801–812, December 2004.
[33] R. C. Smith and P. Cheeseman, "On the representation and estimation of spatial uncertainty," *IJRR*, vol. 5, no. 4, pp. 56–68, 1986.
[34] M. I. Ribeiro, "Gaussian probability density functions: Properties and error characterization," Tech. Rep. 1049-001, Instituto Superior Tcnico, Lisboa, Portugal, 2004.
[35] D. E. Knuth, *The Art of Computer Programming, 2nd Ed. (Addison-Wesley Series in Computer Science and Information*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1978.
[36] S. Kamvar, M. Schlosser, and H. Garcia-Molina, "The Eigentrust algorithm for reputation management in P2P networks," in *Proc. of WWW*, 2003.
[37] N. Spring, L. Peterson, A. Bavier, and V. Pait, "Using PlanetLab for network research: myths, realities, and best practices," *ACM SIGOPS Operating Systems Review*, vol. 40, no. 1, pp. 17–24, 2006.
[38] Y. Chu, A. Ganjam, T. E. Ng, S. G. Rao, K. Sripanidkulchai, J. Zhan, and H. Zhang, "Early experience with an internet broadcast system based on overlay multicast," in *USENIX Annual Technical Conference, General Track*, 2004.
[39] A. Lazarevic, L. Ertoz, V. Kumar, A. Ozgur, and J. Srivastava, "A comparative study of anomaly detection schemes in network intrusion detection," in *SIAM International Conference on Data Mining*, 2003.
[40] Y. an Huang, W. Fan, W. Lee, and P. S. Yu, "Cross-feature analysis for detecting ad-hoc routing anomalies," in *ICDCS*, 2003.
[41] S. Tanachaiwiwat and A. Helmy, "Correlation analysis for alleviating effects of inserted data in wireless sensor networks," in *MobiQuitous*, 2005.
[42] L. Mathy, N. Blundell, V. Roca, and A. El-Sayed, "Impact of simple cheating in application-level multicast," in *INFOCOM*, 2004.
[43] R. Aringhieri, E. Damiani, S. Di Vimercati, S. Paraboschi, and P. Samarati, "Fuzzy Techniques for Trust and Reputation Management in Anonymous Peer-to-Peer Systems," *Journal Of The Am. Soc. For Information Science And Technology*, vol. 57, no. 4, pp. 528–537, 2006.
[44] S. Buchegger and J. Le Boudec, "A robust reputation system for mobile ad hoc networks," in *Proceedings of P2PEcon*, 2004.
[45] K. Aberer and Z. Despotovic, "Managing trust in a peer-2-peer information system," in *Proc. of IKE*, 2001.
[46] E. Damiani, S. di Vimercati, S. Paraboschi, P. Samarati, and F. Violante, "A reputation–based approach for choosing reliable resources in peer–to–peer networks," in *ACM CCS*, 2002.
[47] L. Xiong and L. Liu, "A reputation-based trust model for peer-to-peer ecommerce communities," in *IEEE CEC*, 2003.
[48] M. Srivatsa, L. Xiong, and L. Liu, "TrustGuard: countering vulnerabilities in reputation management for decentralized overlay networks," in *Proc. of WWW*, 2005.
[49] K. Walsh and E. G. Sirer, "Experience with an Object Reputation System for Peer-to-Peer Filesharing," in *NSDI*, 2006.



AAron Walters is a founding member of 4tphi Research. While a member of CERIAS and the the Dependable and Secure Distributed Systems Laboratory, he earned a MS in Computer Science from Purdue in 2006. He received a BS in Computer Engineering from the University of Notre Dame in 2001. His research interests include distributed systems, anomaly detection, digital forensics, and multi-sensor data fusion.

David Zage (S06, ACM06) is a third year PhD student in the Computer Science Department at Purdue University under the supervision of Professor Cristina Nita-Rotaru. He obtained his Bachelor of Science from Purdue in 2004. He is a member of the Dependable and Secure Distributed Systems Laboratory. His research interests include distributed systems, fault tolerance, and security.

Cristina Nita-Rotaru (S'02, M'03, ACM'03) is an Assistant Professor in the Computer Science department of the Purdue University. She leads the Dependable and Secure Distributed Systems Laboratory. She received the BS and MS degrees in Computer Science from Politechnica University of Bucharest, Romania, in 1995 and 1996, and a PhD degree in Computer Science from The Johns Hopkins University in 2003. She served on the technical program committee of numerous conferences such as INFOCOM, ICDCS, ICNP, MOBIHOC. She received the National Science Foundation CAREER award in 2006. Her research interests include secure distributed systems, network security protocols in wired and wireless networks.