V-COPS: a Vulnerability-based Cooperative Alert Distribution System

Shiping Chen[1,2], Dongyu Liu[3], Songqing Chen[3], and Sushil Jajodia[1]

[1]Center for Secure Information Systems
George Mason University
Fairfax, VA 22030, USA
jajodia@gmu.edu

[2]Sybase Inc.
One Sybase Rd.
Dublin, CA 94568, USA
shiping.chen@sybase.com

[3]Department of Computer Science
George Mason University
Fairfax, VA 22030, USA
{dliu1,sqchen}@cs.gmu.edu

Abstract

The efficiency of promptly releasing security alerts of established analysis centers has been greatly challenged by the continuous emergence of various large scale network attacks, such as worms. With a limited number of sensors deployed over the Internet and a long attack verification period, when the alert is released by analysis centers, the best time to stop the attack may have passed. On the other hand, (1) most of the past large scale attacks targeted known vulnerabilities, and (2) today numerous Internet systems have integrated detection tools, such as virus detection software and intrusion detection systems (IDS), the power of which could be harnessed to defend against large scale attacks.

In this paper, we propose V-COPS – a vulnerability-based cooperative alert distribution system, by leveraging existing independent local attack detection systems. V-COPS is capable of promptly propagating genuine alerts with critical vulnerability information, based on which relevant stakeholders can take preventive actions in time. Extensive analysis and experiments have been performed to study the performance of V-COPS. The preliminary results show V-COPS is effective.

1 Introduction

The Internet has witnessed a rapidly increasing amount of various large scale malicious behaviors, such as attacks, virus, and worms, which cause significant loss to the society. To defend against these malicious behaviors, a few security analysis centers, such as DShield [16] and Symantec's DeepSight [14], have been established. These centers proved effective in tracking important security trends that may allow sites to better tune their security postures [15].

However, the frequent outbreaks of large scale network misbehaviors evidence the inability of existing analysis centers to discover and deal with them in time. Simply relying on analysis centers is no longer sufficient. This could be due to several reasons. First, with a limited number of sensors deployed over the Internet, the systems used by analysis centers may not be able to detect attacks in time, while fast attacks leave little time for human-mediated detection. Second, security alerts released by traditional analysis centers normally go through a rigorous verification procedure and take a much longer time than the time needed for an attack to gain global success over the Internet. Although the alerts analysis centers release are genuine, the best time to stop the attacks may have been missed. Third, the alert distribution by analysis centers is a passive approach, by which the analysis centers hope users can actively browse on-line releases and get information in time. Unfortunately, this is not very effective, as existing experience tells that most people (90%) do not patch their systems even after the patch has been available for a long time [12].

On the other hand, we have observed the following two important facts about the Internet. The first is that most of the past large scale network attacks targeted known vulnerabilities [5]. For example, patches for Slammer [9] and CodeRed [10] have been available for quite a while, but Slammer still infected about 75000 SQL servers, and Code Red I infected about 360,000 hosts with financial losses over $2 billion [10]. Users' not patching the vulnerabilities after they were announced might be due to the unawareness of the vulnerability or other technical reasons such as a reluctance to disturb current running systems. For the latter reasons, Shield [18] was proposed to temporarily apply network filters to protect vulnerable systems.

The second observation is that today on the Internet, most connected systems have integrated attack detection tools. Almost every computer connected to the network has virus detection software installed, and many Intrusion Detection Systems (IDS) have been deployed. For example, Snort [13], one of the popular free IDSes, currently has more than 2 million downloads and more than 100,000 active users [3]. These relatively independent systems generate alerts based on the local information, such as the information from local hosts (e.g., Snort) or the information from local networks (e.g., Bro [11]). Accordingly, they generate lots of alerts promptly with different levels of precisions,

and they even may generate false alerts and reduce the trust of users.

The inability of established analysis centers in responding to large scale fast attacks calls for new alerting systems running in a distributed fashion. Motivated by the demand of an efficient alert distribution system, in this work, we propose V-COPS – a Vulnerability-based CoOPerative alert distribution System. V-COPS simply relies on existing detection systems, aiming to promptly propagate genuine alerts with critical vulnerability information, based on which relevant stakeholders can take actions in time. The design of V-COPS enables the following features: 1) V-COPS quickly and actively distributes alerts with vulnerability information, which can lead to prompt response of alert receivers with patching or applying network filters generated by Shield [18]; 2)V-COPS applies hierarchical alert aggregation during alert distribution to effectively filter out false positive alerts; 3)V-COPS itself is robust against various attacks. Extensive analysis and experiments have been performed to study the performance of V-COPS. The preliminary results show V-COPS is effective.

The rest of the paper is organized as follows. We present the design goals and system model in Section 2. Section 3 describes the design details of V-COPS. We revisit the design goals and conduct security analysis in Section 4 and Section 5, respectively. We present related work in Section 6 and make concluding remarks and introduce future work in Section 7.

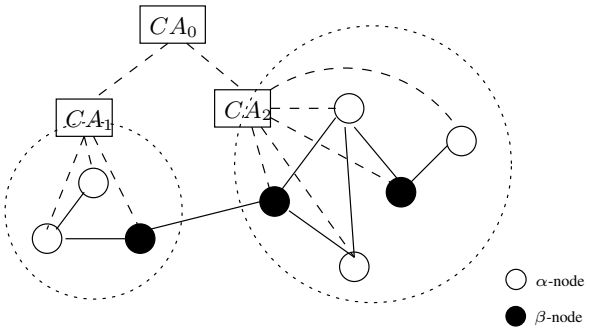


Figure 1. System model.

2 Design Goals and System Model

V-COPS aims to provide a fast and reliable security alert distribution and sharing infrastructure, by which users can react to ongoing network intrusions or worm infections in time, thus to minimize the damage caused by those malicious behaviors. To make it applicable in reality, the system design must achieve the following goals.

- **Fast Propagation and Broad Coverage** The alerts generated by a node should be quickly propagated to as many nodes as possible.
- **Low False Positive Rate** With the high false positive rate of existing local IDSes on which V-COPS relies, V-COPS must be able to filter those false positive alerts out effectively.
- **High Security and Assurance** V-COPS leverages P2P networks for fast alert distribution. On the other hand, attackers can leverage this infrastructure as well. V-COPS must be resilient to those attacks.

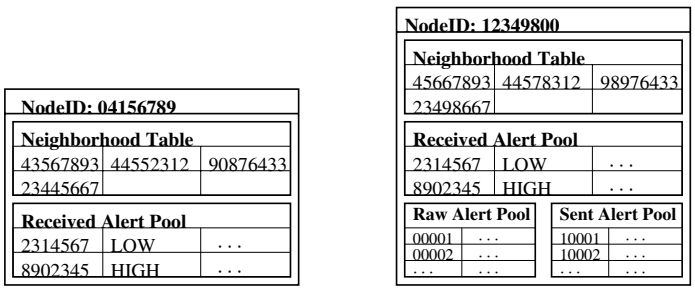
To achieve these goals, we build our cooperative system on top of an unstructured P2P network. We use the following terms in presenting our system model. In V-COPS, a *node* is an Internet host's virtual identity, created by running an instance of the V-COPS software. A *security alert* is a message that contains information about an ongoing network attack or malicious activity. We call an application that generates security alerts an *alert source*.

As shown in Figure 1, there are two types of nodes (α-*node* and β-*node*) in a V-COPS system. If a node has an alert source, it's a β-node, otherwise it is an α-node. According to their definitions, β-nodes are both alert consumers and alert producers, while α-nodes are only alert consumers. β-nodes are also responsible for alert aggregation (more in section 3.2).

Allowing nodes without an alert source to participate in V-COPS has the following advantages. First, their participation increases the coverage of V-COPS dramatically. As a result, it decreases the propagation time of alerts to other nodes, and hence each individual node has sufficient time to react. Second, the more nodes react to the alert, the less nodes that attackers can compromise later. This could dramatically decrease the spreading speed of attacks like fast worms.

Each node in V-COPS has a unique node ID and a public-private key pair. A node ID is a random number of fixed length which is tightly bound to the IP address of the host on which the node resides by a hash function. The public-private key pair is generated by the node itself, and is used to sign messages (alerts for β-nodes) and to allow others to verify the authenticity of the messages (alerts).

Each node maintains a neighborhood table that contains its neighbors' IDs and IP addresses (up to k). k is a parameter which indicates the maximum neighbors a node can connect to simultaneously. Besides a neighborhood table, each α-node maintains a table (referred as to *received alert pool* (RVAP)) to store alerts it receives recently (as shown in Figure 2(a)). In addition to a RVAP, each β-node maintains two other tables (as shown in Figure 2(b)), a *raw alert pool* (RAP) and a *sent alert pool* (SAP). RAP is used to store raw alerts generated by an alert source which resides on the node, and SAP is used to store alerts sent by this node. All tables are bounded by space and time. If a table is full, the older alerts will be replaced.



(a) α-node (b) β-node

Figure 2. V-COPS nodes structure.

We assume there is a traditional hierarchical structured certificate authority (CA) system available in V-COPS. These CAs act as system managers, and are responsible for issuing certificates to nodes, assigning node IDs, and serving as bootstrap index servers in the process of node joining. Each CA manages a portion of V-COPS nodes. Each child CA is certificated by its parent CA, which enables nodes managed by different CAs to verify each other.

As V-COPS runs on a P2P overlay network, the following operations are fundamental to support our design goals.

- **Node Join:** To join V-COPS, a node first contacts a preloaded CA to get a unique node ID, a public-key certificate, and a list of bootstrap nodes. Then the new node contacts the bootstrap nodes one by one to request to be their neighbors following the standard P2P node joining procedure. Once accepted, both the new node and the requestee update their neighborhood tables to include each other's ID and IP address. Note that mutual authentications are performed in the node join procedure.
- **Keep Alive:** Each node periodically sends *keep-alive* messages to all of its neighbors. If during a period of time, a node does not receive a *keep-alive* message from one of its neighbors, it treats that node *dead*, removes the corresponding entry from its neighborhood table, and tries to look for a new neighbor. To look for a new neighbor, a node may ask some of its neighbors to send their neighborhood tables to it, and selects some nodes to establish new neighborhood relationships.
- **Node Leave:** There are two types of node leaves. If a node wants to leave V-COPS permanently, it informs its neighbors to ask them to remove its information from their neighborhood tables. It also informs the CA to remove its information from its bootstrap node repository (if it is in). Otherwise, if it just leaves the V-COPS temporarily (off-line) and will come back again later, it needs to do nothing. If a node leaves the system for a while, its entry in original neighbors' neighborhood table may expire. In this case, it needs to do a new join operation.

3 V-COPS Design

V-COPS mainly consists of three components: (1) a vulnerability based hierarchical alert aggregation method, which can effectively filter out false positive alerts during alert distribution; (2) a probability based verification mechanism, which can filter out illegitimate alerts effectively; and (3) a hybrid distribution scheme, which can propagate alerts quickly to participants. Before describing these three components, we first introduce alert designs in V-COPS.

3.1 Security Alert Design for V-COPS

Security alerts are the information carriers in V-COPS, which describe the properties of ongoing malicious behaviors. In V-COPS, there are three types of alerts. Alerts generated directly by alert sources are *raw* alerts. The formats and contents of raw alerts are application dependent. For example, a Snort alert is different from a Bro alert in many aspects. Strictly speaking, raw alerts are not V-COPS alerts because they only live in the perimeter of a node, and are invisible to other peers.

The alerts generated directly by V-COPS nodes are *basic* alerts. Basic alerts are the major alerts distributed in a V-COPS system. Each of them is a result of *thread* aggregation on multiple raw alerts. Another type of alerts distributed in V-COPS is the *compound* alert. A compound alert is the result of *incident* aggregation on multiple basic alerts. Thread aggregation and incident aggregation will be introduced in Section 3.2. Figure 3 gives a graphic illustration of the relationships among these three types of alerts.

A basic alert has the following fields:

- *Threat level* A number to indicate how severe the attack is. For basic alert, it is *LOW* (0).
- *Alert ID* A randomly generated bit string to uniquely identify an alert.
- *Time stamp* Time when the alert is originally generated.
- *Generator ID* The ID of the node that issues the alert.
- *Targeted vulnerability* A field indicates the host or network vulnerabilities that the intrusions are targeting.
- *Options* An field containing other optional information such as recommended security countermeasures.

A compound alert has a similar format to the basic alert except for the following:

- Threat level is one of the following: *GUARDED*(1), *HIGH*(2), and *SEVERE*(3).

- It has one additional field named *references*, which contains a list of basic alert IDs. Because each compound alert is the aggregation result of a bunch of related basic alerts, those alert IDs provide a way for nodes to trace back to the original basic alerts.

We assume that, in V-COPS, all alert sources are Common Vulnerability and Exposures (CVE) [1] compatible. With continuous efforts of representatives from numerous security-related organizations such as security tool vendors, academic institutions, and government as well as other prominent security experts, CVE has been a widely accepted name standard for vulnerabilities and other security exposures. Before being sent out, each alert (basic or compound) is wrapped into an alert message, which has the following format: $alert|Sig|Cert$, where Sig is the signature generated by the node using its private key on the alert, $Cert$ is the node's certificate, and '|' denotes concatenation. In the rest of this paper, we use alert and alert message interchangeably if the context is clear.

V-COPS alert design has the following features. First, V-COPS alerts are vulnerability oriented. They do not include attacker and victim related information (such as attacker IP, victim IP, and attack class) as common fields. We strongly believe that vulnerability is the precondition of any successful attacks. And it's also the most reliable information that stakeholders can rely on to protect themselves from being attacked using available resources.

Second, instead of a *time-to-live* field, V-COPS uses the time stamp field to restrict the *life time* an alert can live. This design is mainly due to security considerations. A dynamically changing field leaves a security hole to attackers. Here we assume that all V-COPS nodes synchronize their system times with Internet time using network time protocol [2].

Third, different from the security attacks reported by established Internet analysis centers, such as DShield [16] and Symantec's DeepSight [14], the security alerts announced and distributed in V-COPS are not after long time analysis or verification. This increases the false alarm rate. V-COPS uses several other mechanisms to filter out the false positives.

3.2 Vulnerability based Hierarchical Alert Aggregation

A common limitation of existing intrusion detection systems is that they normally issue too many alerts. A lot of those alerts are duplicated in the sense that they are identical except that they are issued at a slightly different time, or issued by detectors installed at different locations. It's desirable to have as few EFFECTIVE alerts as possible when reporting the same ongoing attack. V-COPS uses aggregations to reduce alert duplications.

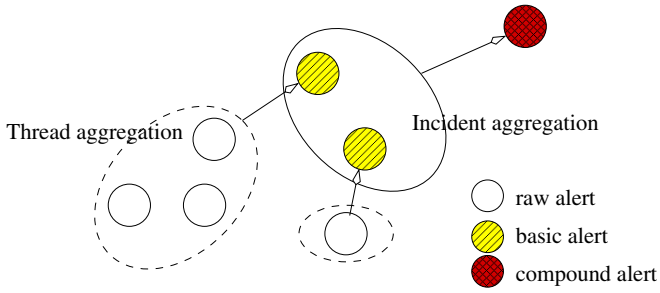


Figure 3. Relationships among different types of V-COPS alerts.

As aforementioned, there are two types of aggregations in V-COPS, *thread* aggregation and *incident* aggregation [1]. Both aggregations are executed on β-nodes. Thread aggregation is to aggregate the raw alerts that are from the same alert source and referring to the same attack, possibly different in time. Incident aggregation aggregates basic alerts into compound alerts. Those basic alerts are from different intrusion detection systems, possibly different in time and location, but referring to the same type of attacks targeting the same vulnerability.

A β-node does thread aggregation as follows. All new raw alerts generated by the alert source that resides on the node are put into the node's RAP. With a pre-determined raw alert process time interval T_{ta}, a β-node clusters all the raw alerts it generated in the past T_{ta} time into different groups based on the threads they belong to. Performing clustering is not straightforward, especially when the raw alerts belonging to one thread have different values in alert fields other than the time stamp. For example, raw alerts issued by a network-based IDS referring to the same attack may have different alert target values (e.g., target IP addresses) when the attack was launched sequentially in the network where the IDS resides. In this case, some similarity functions are needed to correctly cluster the alerts. Valdes and Skinner [17] proposed a probabilistic framework for this purpose. In its current design, V-COPS adopts a simple vulnerability-centric criteria for alert clustering. That is, if two raw alerts refer to the same CVE vulnerability, they are clustered into the same group.

Once raw alerts are clustered, for each cluster, a new basic alert is generated. The new basic alert is assigned a unique randomly generated ID, with the *threat level* set to LOW, the *generator ID* set to the ID of the node itself, and the alert *time stamp* field set to the system time when the alert is generated. The *targeted vulnerability* field is filled with the common vulnerability those raw alerts share.

[1] We borrow the two terms, *thread* and *incident*, from Valdes and Skinner [17].

Other attack related information, such as corresponding recommended actions, can be put into the options field. Once a basic alert is generated, the original raw alerts are discarded.

Incident aggregation is done in a similar way as that of thread aggregation, except that the aggregated objects are basic alerts and the output is compound alerts. β-nodes also do incident aggregation in a batch fashion. That is, every T_{ia} time, a β-node checks its RVAP to perform possible incident aggregation. A certain amount of basic alerts belonging to the same attack incident will be aggregated into a compound alert. Once aggregated, the original basic alerts are discarded, and the compound alerts will be distributed to the network. Again, we use the vulnerability-centric criteria in determining if two basic alerts belong to the same incident.

Each compound alert has a *threat level* value to indicate how severe the attack is. In its current design, V-COPS adopts a three-level indicator system, namely *GUARDED*, *HIGH*, and *SEVERE*, with increasing severity order. The threat level of a compound alert is based on the number of basic alerts from which it is generated. V-COPS uses three threshold values, namely m_1, m_2, and m_3, to decide what threat level a compound alert should be. That is, if less than m_1 alerts are received in the current T_{ia} time interval, no incident aggregation is performed, and all the basic alerts are discarded. If more than m_3 alerts are received, a SEVER level compound alert is generated. When the number of basic alerts falls in between m_1 and m_2, a GUARDED alert is generated, otherwise, a HIGH level alert is generated. m_1, m_2, and m_3 are system wide adjustable parameters. Alerts with different threat levels have different expiration times. The higher the threat level, the longer the expiration time.

V-COPS aggregation designs deserve some extra comments. First, T_{ta} and T_{ia} are system wide parameters. The larger they are, the more alerts are aggregated, therefore the larger the alert number reduction. But, the delay introduced to propagation increases as well. Second, in incident aggregation, we only count basic alerts from distinct nodes, basic alerts coming from a single node are treated as one alert. This design lifts the bar of attackers' launching DOS attacks and sybil attacks against V-COPS itself. Third, the hierarchical aggregation structure (m_1, m_2, and m_3) effectively filters out false positive alerts, and the hierarchical design of expiration times increases the coverage of popular alerts while limits the coverage of rare alerts such as false positive alerts (more details in section 4.2).

3.3 Probability based Alert Verification

Verification is an important mechanism of V-COPS to defend against attacks. Two types of verifications are performed in V-COPS. The first one is to verify the authenticity of an alert itself, applied to both basic alerts and compound alerts. The other is to verify if an incident aggregation is done properly. In other words, to verify if a compound alert does really come from those basic alerts it claims.

To perform the first type of alert verification, a receiver first decomposes the alert message it receives into three parts, namely, the alert itself ($alert$), the signature (Sig), and the certificate ($Cert$). Then it verifies the authenticity of the certificate via traversing the CA hierarchy. This procedure makes sure that the public key contained in the certificate does belong to the alert generator. If the certificate verification succeeds, the node verifies the signature by decrypting the alert using the extracted public key. Since the signature was produced on the alert using the generator's private key, if the alert message is not compromised, this verification should succeed. If any of the above verifications fails, that means that the alert message is a faked one or has been tampered with during its distribution. Therefore, it should be dropped.

To perform the second type of verification, a receiver node first retrieves the references from the compound alert that needs to be verified. Then the node tries to get all or some of those basic alerts to check if the incident aggregation is done following the system rule. Generally, there are two ways to get those related basic alerts. A node first checks its own RVAP to see if it has a copy of the basic alert. If it can't find one, it sends a *query* to one of its neighbors to ask for a copy. The query includes the basic alert ID it is looking for, and is propagated into the system in a random walk fashion. Any node receiving the message with an available copy of the request alert in its RAP or SAP will answer the query and send the copy to the requester.

A V-COPS implementation may adopt different alert verification policies. If network bandwidth is limited, V-COPS may require each node to do a verification when receiving an alert. This policy filters illegitimate (faked or compromised) alerts in their first hop of distribution. However, this may add too much overhead in alert processing, especially when most of the alerts are legitimate. A better policy is that each α-node verifies alerts with some probability, while each β-node verifies alerts with certainty, since β-nodes normally have more computation power than α-nodes, and β-nodes' accepting of illegitimate alerts causes more damage to the system due to their aggregation capability.

3.4 Hybrid Alert Distribution

The ultimate goal of V-COPS is to distribute genuine alerts to as many users as possible and as quickly as possible. With an unstructured P2P network, the easiest way to achieve this goal is flooding. However, flooding can cause too much unnecessary traffic that could prevent alert propagation. Thus, in V-COPS, we propose a hybrid alert propa-

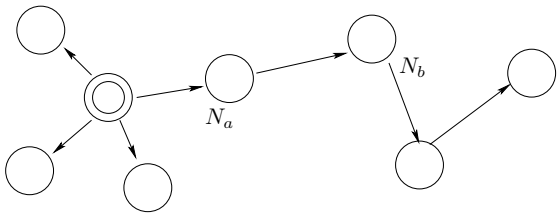


Figure 4. Hybrid alert distribution.

gation algorithm. The principle of this algorithm is based on the following observations. With pure flooding, the coverage expands quickly only during the first several hops. After that, flooding will largely distribute repetitive messages. This has been verified by LightFlood [8]. The random walk algorithm can reduce repetitive messages effectively, but it has a slow coverage rate. To leverage the merits of these two algorithms and overcome their shortcomings, we propose the following hybrid distribution algorithm.

- The alert generator always floods the alert it generates to all of its neighbors.
- All the following propagations are performed in a random walk fashion. That is, each receiver forwards the alert only to (randomly chosen) one of its neighbors.
- The random walk forwarding continues until the alert expires or is dropped by a β-node intentionally.

In alert propagation, validation is performed in each step. Once an alert becomes invalid (i.e., the alert was generated T_{l_i} time ago, here T_{l_i} is the life time of the alerts with threat level i), it is dropped immediately. V-COPS nodes may also verify the authenticity of each forwarded alert according to the system's verification policy. However, in addition to following the above mentioned rules, a β-node takes an extra action towards basic alerts. In particular, when receiving a valid and authentic basic alert, instead of forwarding it out right away, a β-node does a quick look up in its RVAP to see if it has received the alert before. If it did receive the alert before, it simply drops the alert. Otherwise, it puts the alert into its RVAP. Note that instead of forwarding the alert message immediately, a β-node stops the alert propagation temporarily for possible incident aggregation. If the alert is aggregated, the information contained in the basic alert will be propagated in the form of a new compound alert, otherwise, the alert finishes its journey.

Figure 4 gives a graphic illustration of the hybrid alert distribution algorithm. In the figure, when propagating an alert, the alert generator (the double circled node) initially floods the alert. Upon receiving the alert, one of its neighbors, say node N_a, instead of flooding the alert, sends the alert to one of its neighbors (selected randomly), node N_b in this example.

3.5 Alert Reaction/Response

RVAP provides valuable information that users need for making their reaction decisions. The vulnerability field of V-COPS alerts is particularly important. Since many attacks target existing vulnerabilities, once informed by V-COPS, users can quickly identify patches or use network filters provided by Shield [18]. Given a set of alerts, how to react is subject to the receiver's reaction policy. Relatively conservative users may adopt a policy by which they may shut down vulnerable applications temporarily or patch the system once receiving one single GUARDED compound alert. However, moderate-risk takers may defer the immediate reaction to the time when further evidence (such as their own IDS alarms) appears even if they receive multiple SEVERE compound alerts. Part of the reactions can be performed automatically by security tools, such as Shield [18]. Designing appropriate reaction policies is out of the scope of this paper.

4 Design Goals Revisit

As stated in Section 2, V-COPS is designed to achieve three major goals: (1) propagating security alerts into the network as soon and wide as possible; (2) filtering false positives as much as possible; and (3) remaining resilient to attacks targeting V-COPS itself as much as possible. In this section, we analyze the performance of V-COPS in terms of the first two goals, and leave the security analysis to the next section. The notations we use for analysis are summarized in Table 1 .

In addition to analysis using modeling, we also perform simulations to validate the performance of related mechanisms using the following simulation settings. We simulate a V-COPS system with 8000 nodes, 20% of which are β-nodes. The average connection degree for each node is 8. The incident aggregation thresholds (i.e., m_1, m_2, and m_3) are set as 2, 5 and 8. For simplicity, we assume that it takes one time unit (in tens of msecs in practice) to propagate an alert per hop. For the same reason, we simulate the *life time* value by the number of hops that an alert can traverse at most. Specially, the *life time* values for different threat level alerts are set as 5, 10, 20, and 40, respectively.

4.1 Coverage and Traffic Reduction

In this section, we study the performance of V-COPS in alert coverage and traffic reduction. *Alert coverage* measures the performance of V-COPS in alert propagation, and message *duplication rate* measures the effectiveness of V-COPS in traffic reduction. Alert coverage is defined as the average number of nodes (or percentage of nodes) in a V-COPS system that receive a basic alert. A node is in the cov-

n	total number of nodes in V-COPS
α	portion of α-nodes
β	portion of β-nodes
c	compromised portion of V-COPS nodes
c_β	compromised portion of β nodes
d_i	connection degree of node i
$\overline{d}$	average connection degree of nodes
T_{ta}	time interval for thread aggregation
T_{ia}	time interval for incident aggregation
m_i	aggregation threshold for compound alert with threat level i
T_{l_i}	life time for alert with threat level i
r	duplication rate of received alert message
G_β	percentage of β-nodes that generate alerts with the same vulnerability

Table 1. Notations used in analysis

erage of a particular basic alert if it receives a copy of that basic alert, or receives a copy of a compound alert that is an incident aggregation result involving that basic alert. Duplication rate is the average number of copies of an identical alert a node receives in the life time of all alert messages.

We model the alert coverage of V-COPS as follows. Since V-COPS uses a hybrid algorithm to distribute alert messages, if we assume a basic alert is not dropped by a β-node due to incident aggregation, the alert can be received by at most $\overline{d}T_{l_0}$ nodes, where $\overline{d}$ is the average connection degree of a V-COPS node, and T_{l_0} is the *life time* for a basic alert. Consider the effect of the hierarchical aggregation mechanism, a basic alert may be dropped in the middle of its propagation journey. We further assume that the alert has p_1 probability being aggregated into a GUARDED compound alert, p_2 probability being aggregated into a HIGH compound alert, and p_3 probability being aggregated into a SEVERE compound alert. Then we can get the alert coverage as the following formula:

$$\overline{d}T'_{l_0}(1-p_1-p_2-p_3)+p_1\overline{d}T_{l_1}+p_2\overline{d}T_{l_2}+p_3\overline{d}T_{l_3}, \quad (1)$$

where T'_{l_0} is the number of hops a basic alert traversed before it is aggregated, and T_{l_i} is the life time of a compound alert with threat level i. Note that a compound alert is not dropped in its life time.

Formula 1 clearly shows that the higher the probability a basic alert is aggregated into a severer compound alert, the larger the alert coverage. This property meets V-COPS design goal very well because of the following reason. In a cooperative alert distribution system, it is desirable that true positive alerts have larger coverage while false positive ones have smaller coverage. Any large scale rapid malicious behavior would be detected and reported by a large portion of nodes, hence those basic alerts are more likely to be aggregated into compound alerts with higher threat levels. Instead, those alerts alarming local attack events or alarming a large scale attack event falsely would not have a large population, therefore have a small coverage.

We further validate the property using simulation. At the beginning of the simulation, we randomly select a certain percentage of β-nodes (denoted as G_β) to generate basic alerts that carry the same vulnerability but with different alert IDs. Those alerts are then propagated into the network using our hybrid propagation algorithm. When all the alerts in the system die out (due to expiration or β-nodes' intentional dropping), we measure the average coverage of those basic alerts generated at the beginning of the simulation.

Figure 5 shows that when the number of alerts with a same vulnerability is small, the average alert coverage is also small. However, once the number of alerts with a same vulnerability increases, the average alert coverage increases dramatically. For example, when G_β is 1% (i.e., 16 β-nodes generate alerts), the average alert coverage is 28. When G_β increases to 25% (i.e., 400 β-nodes generate alerts), the average alert coverage is increased to 180. This result reflects the amplifying power of V-COPS' aggregation mechanism indicated by equation 1.

Since V-COPS uses a hybrid propagation algorithm instead of a pure flooding algorithm, we expect it has a low alert duplication rate. The aggregation mechanism also further reduces the alert duplication rate. To validate our expectation, we measure the duplication rate with simulations. In these simulations, we use the same settings as for the coverage measurement, and measure the duplication rate with different G_β values. The duplication rate of V-COPS is compared with that of using a pure flooding algorithm.

Figure 6 shows the result. The two curves show the average duplication rate changing trends with different G_β for the flooding algorithm and V-COPS hybrid distribution and aggregation mechanisms. The figure indicates that V-COPS is very effective in network traffic reduction. For example, when G_β is 10%, the duplication rate of the flooding algorithm is as much as 5 times of the duplication rate of V-COPS. Generally, compared to flooding, V-COPS reduces duplication rate by several folds.

4.2 False Positive Alerts

The high false positive rate is one of the major drawbacks of current intrusion detection systems. We expect that a high portion of alerts propagated in V-COPS are false positive alerts. If these alerts are not eliminated, not only precious bandwidth and computation resources would be wasted, but also the reactions of users towards genuine alerts would be disturbed.

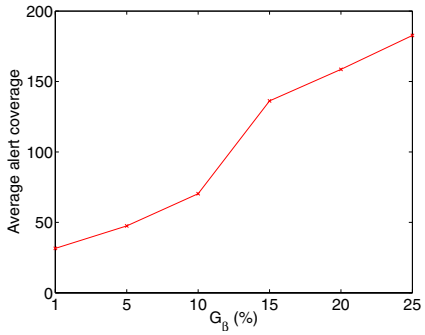


Figure 5. Average alert coverage.

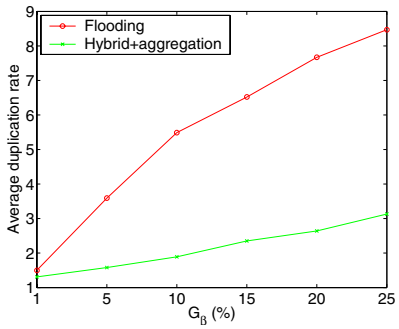


Figure 6. Average message duplication rate.

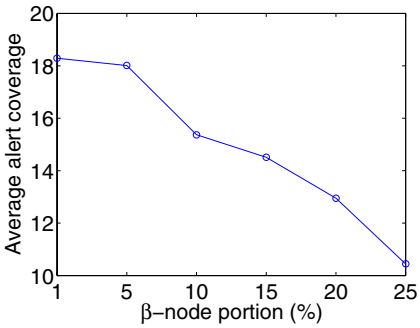


Figure 7. The average coverage of a single false positive alert.

V-COPS relies on two mechanisms to filter out false positive alerts. The first is that each alert's valid time is confined by a system wide parameter T_{l_i} (i is the threat level of an alert). When receiving an alert, a node checks whether the alert is still valid. Once expired, the alert is dropped immediately. That means the distribution coverage of a false positive basic alert is limited to $T_{l_0}\overline{d}$, where $\overline{d}$ is the average connection degree of a β-node, and T_{l_0} is the *life time* value for a basic alert. By adjusting the parameter T_{l_0}, we can adjust the sensitivity level of the system to false positive alert filtering.

Another mechanism V-COPS uses to filter false positive alerts is its incident aggregation design. In an aggregation time frame T_{ia}, if the number of one kind of basic alerts from distinct generators does not exceed the minimum threshold m_1, all those alerts are dropped immediately. Although m_1 is quite small, normally not greater than 5, false positive alerts still can not succeed in passing the aggregation requirement, in most cases. The rationale behind this mechanism is that alerts are application and vendor dependent. For a given event, the possibility of two or three detection applications that are from different vendors issuing the same type of false positive alerts is quite small. m_1 is also an available knob in adjusting the system's false positive alert filtering sensitivity.

We conduct simulations to measure how much coverage a false positive alert can achieve in V-COPS. In these simulations, we use the same setting as for alert coverage measurement, and measure the coverage of a single basic alert under different β values (i.e., the portion of β nodes). Each simulation is run 100 times, and each time a different β-node is used to generate a basic alert. The average value of the 100 runs is reported in Figure 7.

Figure 7 illustrates that in V-COPS a false positive alert has a very small coverage. And the coverage decreases with the increase of the β-node portion. For example, when the β-node portion is 25%, the average alert coverage is as small as 10.5.

5 Security Analysis

In this section, we address threats to V-COPS by considering the most likely forms of attacks that may be attempted. These include malicious alert message dropping, fake alert message injecting, and sybil attack launching. It's definitely not a complete list in any sense, but we believe these three are the major threats to V-COPS considering its design goals.

5.1 Malicious Alert Message Dropping

The essential paradigm of V-COPS is to alert participating nodes when real network attacks or virus infections are ongoing. Once alerted by other nodes, a node may take proper actions to prevent itself from being attacked. From the view of an attacker, the most effective way to maximize its attack achievement is to break the alert message distribution path or at least make the alert propagation slower than its attacking pace. Among all possible approaches, intentionally dropping all passing alert messages is the easiest way for an attacker to implement. In this attack, attackers may compromise a portion of V-COPS nodes, and block all outgoing messages except their own attacking traffic.

By virtue of the fact that V-COPS is built on an unstructured P2P overlay network, it is resilient to the alert dropping attack. In V-COPS, no node is taking a more significant role in alert forwarding than others. Compared with those systems (such as DOMINO [19]) that use a supernode P2P structure as their underlying infrastructure, V-COPS is more resilient to malicious dropping attacks. And because V-COPS uses a hybrid algorithm to distribute alerts (i.e., any alert is propagated in the system in d paths, where

d is the connection degree of the alert generator), malicious dropping can only slow down the propagation coverage partially.

V-COPS can further defend itself from malicious alert dropping attack by using a self-healing mechanism. For example, if a node has not received any alert from one of its neighbors for a specified period of time, it can conclude that the neighbor may have been compromised, and then try to look for a new neighbor. This mechanism is effective even when compromised nodes respond to keep-alive messages as usual.

We validate the capability of V-COPS in defending against malicious message dropping attack through simulations. In these experiments, we use the same experiment settings as before. At the beginning of the experiment, 10% of randomly selected β-nodes generate alerts with the same vulnerability, and propagate them into the network. We assume that a fixed percentage of nodes are compromised and they drop every message they receive intentionally. We compute how many nodes receive at least one of those basic alerts or a compound alert in the experiment.

Figure 8 shows the experimental results. In the figure, each curve represents the alerts propagation trend in a different malicious node portion. We can see that the malicious dropping does not seriously affect alert propagation. Compared with the situation where there is no malicious dropping, when 10% of nodes are malicious nodes and they drop messages intentionally, the alert propagation speed is only delayed by two time units (e.g., it takes 7 instead of 5 time units to propagate the alert to 5000 nodes).

5.2 Fake Alert Injection

Attackers may launch denial of service attacks by injecting a huge amount of fake alerts into V-COPS. Their purpose is to use up V-COPS nodes' precious resources such as limited network bandwidth or computation power. We can further categorize these attacks into two types. The first type of attacks occurs when attackers arbitrarily produce a large amount of alert messages (illegitimate alert messages) using other existing or non-existing β-node IDs, and then inject them into the network through compromised nodes. The other type of attacks occurs when attackers compromise some β-nodes and have them generate and inject a large amount of legitimate but useless alert messages. We call the first type of attacks an illegitimate fake alert injection attack, and the second type an legitimate fake alert injection attack.

The illegitimate alert injection attack is easy to launch, since it does not require compromising a β-node. Fortunately, it is also easy to defend against by the verification mechanism used in V-COPS (see section 3.3 for details). We conduct two sets of simulations to practically measure the effectiveness of V-COPS in filtering illegitimate alerts with different verification policies. At the beginning of experiments, there are 1000 illegitimate alerts in the system. For the first set of simulations, we fix the β-node portion as 20%, and change verification probabilities for an α-node, while in the second set of simulations, we fix the α-node verification probability as 1%, and change the β-node portion.

Figure 9 and 10 show the number of illegitimate alerts alive in the system along time under different verification policies. These two figures show that it takes at most 5 time units to filter out all those 1000 illegitimate alerts (the value 5 is coincident to the life time value of basic alerts), and adjusting the α-node verification probability or the β-node portion, one can effectively adjust the illegitimate alert filtering speed.

The legitimate fake alert injection attack is more harmful to V-COPS than the first type. Since all the fake alerts injected by attackers are legitimate, they can not be filtered out by the verification mechanism. However, V-COPS is resilient to this kind of attack, too, because of the following two reasons. First, The essential objective of V-COPS is to alert nodes that one particular vulnerability is under attack, and urge the nodes to take preventive actions (such as patching). Attackers may not want to deceive V-COPS users to patch existing vulnerabilities, since those vulnerabilities may be utilized by them to gain more benefits.

Second, V-COPS' incident aggregation mechanism mitigates the effectiveness of this type of attacks efficiently. For incident aggregation, if in a particular aggregation time frame T_{ia}, the number of basic alerts from distinct generators does not exceed the threshold m_1, they are dropped immediately. This requirement lifts the bar of a successful attack. We can roughly estimate the success rate of this type of attacks as follows.

Suppose we have n nodes in a V-COPS system, a β portion of which are β nodes, and c_β portion of β-nodes are compromised. For simplicity, we assume that the life time of each basic alert is h in hops, incident aggregation time frame is a in hops, and the average connection degree of nodes is $\overline{d}$. We further assume that the message duplication rate is r. Then we can use the following formula to model how many basic alerts a β-node can receive in time a, which are from distinct compromised generators. We denote this average value as A,

$$A = \frac{c_\beta \cdot \beta \cdot n \cdot h \cdot d}{n} \cdot \frac{a}{h} \cdot \frac{1}{r} = \frac{c_\beta \cdot \beta \cdot d \cdot a}{r}.$$

To avoid being filtered out in indent aggregations, A must be greater than m_1, i.e., $A > m_1$. Then we can compute the minimum portion of compromised β-nodes needed to launch a legitimate fake alert attack as:

$$c_\beta = \frac{m_1 \cdot r}{\beta \cdot d \cdot a}.$$

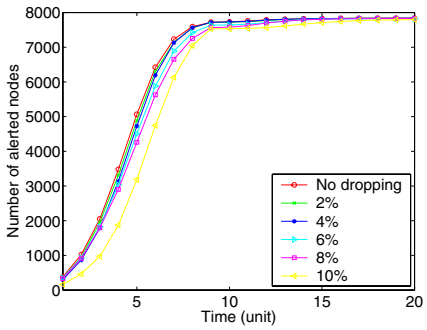


Figure 8. Effect of malicious dropping.

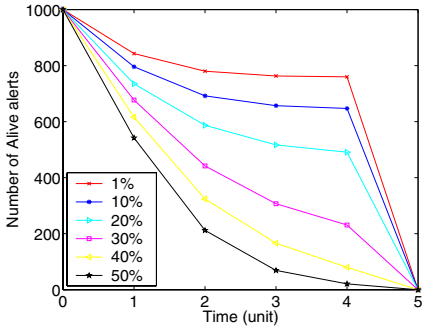


Figure 9. The illegitimate alert filtering trend with different α-node verification probabilities.

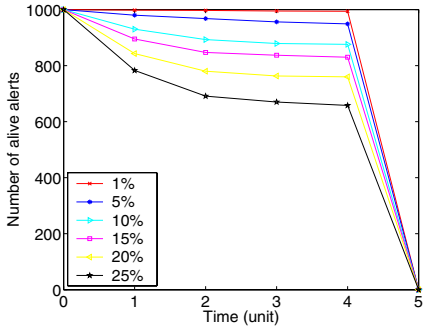


Figure 10. The illegitimate alert filtering trend with different β-node portion.

Considering a normal set of system parameters: $m_1 = 2$, $r = 2$, $\beta = 0.2$, $d = 8$, and $a = 5$, we can get $c_\beta = 0.5$. That means, to successfully launch such a legitimate fake alert attack, attackers need to compromise 50% of the β-nodes, a bar that is not easily approachable. Adjusting the parameter can lift the bar even further.

5.3 Sybil Attack

In a P2P system, it's common that an entity can acquire or forge multiple identities, thereby control a substantial fraction of nodes in the system. With a large amount of identities, an attacker can easily partition the network or isolate a victim node from receiving any information. This attack is referred as sybil attack [7].

By virtue of its design, V-COPS can mitigate sybil attacks substantially. Each V-COPS system has a trusted hierarchically structured certificate authority (CA) system charging of node identity and certificate issuing, and each V-COPS node ID is tightly bound to an IP address by a certificate. This design prohibits an attacker from acquiring a large fraction of node identities in a short time. Therefore, an attacker's attempt of partitioning the V-COPS network or isolating a node is difficult to carry out. V-COPS nodes can intentionally select nodes from different networks as their neighbors to further decrease the isolation threat.

Attackers may also clone a compromised V-COPS node and use cloned nodes to inject a large amount of legitimate fake alert messages into the system, but this type of sybil attack is not a serious threat to V-COPS either. As discussed in section 5.2, alerting nodes to patch vulnerabilities that may be utilized by themselves does not comply with attackers' interest. In addition, those faked messages will be filtered out of the system very quickly because multiple alerts from the same node ID will only be counted once within the V-COPS incident aggregation mechanism.

6 Related Work

Defending against various large scale network based misbehaviors, such as attacks, worms, and virus, has been the focus of lots of security research and practice. Plenty of work has been conducted on centralized correlation, such as DShield [16] and DeepSight [14], and distributed correlation, such as COVERAGE [4], DOMINO [19], and Vigilante [6].

The research closest to V-COPS is DOMINO [19]. But V-COPS differs from DOMINO significantly in several aspects. First, DOMINO itself emphasizes detection functions to generate a blacklist, which may not be effective upon attack dynamics. On the contrary, V-COPS effectively aggregates and distributes alerts with critical vulnerability information so that alert receivers can effectively avoid being attacked. Second, DOMINO heavily relies on active sinks to remove false alerts, while in V-COPS, hierarchical aggregation and probability based verification can effectively filter out false alerts during alert distribution. Third, DOMINO relies on a structured organization, while V-COPS organizes existing systems with an unstructured P2P overlay, which is simpler. This enables any IDS system that follows CVE [1] to work freely within V-COPS.

7 Conclusions and Future Work

To help users defend against large scale fast network malicious behaviors, this paper has proposed a vulnerability-based cooperative alert distribution system, called V-COPS. The essential paradigm of V-COPS is to alert participating nodes of an ongoing attack targeting a particular vulnera-

bility, and to urge users to take corresponding preventive actions.

By leveraging the scalable property of the underlying P2P overlay network, enhanced with specially designed mechanisms, V-COPS achieves the following design goals: (1) it can deliver true positive alerts quickly and scalably to participating nodes without generating significant network traffic; (2) it can filter out false positive alerts automatically, which makes it possible to utilize existing IDSes to precisely alarm large scale ongoing attacks or worm bursts; and (3) it is resilient to various attacks that are targeting itself.

VCOPS is to improve the alert distribution speed and coverage over the existing systems run by security centers to further reduce the damage. The current design of VCOPS itself does not assume the initial detection function, which is the most critical demand to deal with the fast spreading worms over the Internet.

With the prototype system implementation ongoing, we plan to enhance V-COPS in the following ways. First, in the current design, all alerts are treated equally. That is, all basic alerts, no matter it is generated by a personal open source IDS or a professional enterprise IDS, have the same weight in alert aggregation. Generally, alerts issued by an enterprise IDS are more trustworthy than alerts issued by a personal IDS. In the future, we plan to integrate trust level in alert aggregation.

Second, in its current design, V-COPS does not utilize locality information. In practice, locality information is also useful for end users to react to the alert properly. For example, an alert from a nearby node should be treated more urgently than an alert from a node far away. We plan to include locality oriented optimization in our future work.

Acknowledgment

This material is based upon work supported by Homeland Security Advanced Research Projects Agency under the contract FA8750-05-C-0212 administered by the Air Force Research Laboratory/Rome, by Army Research Office under grant W911NF-06-1-0019, by Federal Aviation Administration under the contract DTFAWA-04-P-00278/0001, and by the National Science Foundation under grants CT-0627493, IIS-0242237, IIS-0430402, and CNS-0509061 .We would like to thank William L. Bynum and anonymous reviewers for their helpful comments on this paper.

References

[1] Cve-common vulnerabilities and exposures. http://www.cve.mitre.org/.

[2] Ntp: The network time protocol. http://www.ntp.org/.

[3] Snort:. http://www.sourcefire.com/snort.

[4] K. Anagnostakis, M. Greenwald, S. Ioannids, A. Keromytis, and D. Li. A cooperative immunization system for an untrusting internet. In *Proceedings of the International Conference on Networks*, 2003.

[5] W. Arbaugh, W. Fithen, and J. McHugh. Windows of vulnerability: a case study analysis. In *IEEE Computer*, 2000.

[6] M. Costa, J. Crowcroft, M. Castro, A. Rowstron, L. Zhou, L. Zhang, and P. Barham. Vigilante: End-to-end containment of internet worms. In *Proceedings of SOSP*, Brighton, United Kingdom, October 2005.

[7] J. Douceur. The sybil attack, March 2002.

[8] S. Jiang, L. Guo, and X. Zhang. Lightflood: an efficient flooding scheme for file search in unstructured peer-to-peer systems. In *2003 International Conference on Parallel Processing (ICPP'03)*, Kaohsiung, Taiwan, October 2003.

[9] D. Moore, V. Paxson, S. Savage, C. Shannon, S. Staniford, and N. Weaver. The spread of the sapphire/slammer worm. http://www.caida.org/publications/papers/2003/sapphire /sapphire.html.

[10] D. Moore, C. Shannon, and Jeffery Brown. Code-red: a case study on the spread and victims of an internet worm. In *Proceedings of the second Internet Measurement Workshop*, Novermber 2002.

[11] V. Paxson. Bro: a system for detecting network intruders in real time. In *Computer Networks*, volume 31, December 1999.

[12] E. Rescorla. Security holes... who cares? In *Proceedings of USENIX Security*, August 2003.

[13] M. Roesch. Snort: Lightweight intrusion detection for networks. In *Proceedings of Conference on System Administration*, November 1999.

[14] Symantec. Deepsight threat management system home page. Http://tms.symantec.com.

[15] SymantecReport. Symantec internet security threat report, February 2003.

[16] J. Ullrich. Dshield home page. Http://www.dshield.org.

[17] A. Valdes and K. Skinner. Probabilistic alert correlation. In W. Lee, L. Me, and A. Wespi, editors, *Proceedings of RAID 2001*, volume LNCS 2212, pages 54–68. Springer-Verlag, 2001.

[18] H. Wang, C. Guo, D. Simon, and A. Zugenmaier. Shield: Vulnerability-driven network filters for preventing known vulnerability exploits. In *Proceedings of ACM SIGCOMM'04*. August 2004.

[19] V. Yegneswaran, P. Barford, and S. Jha. Global intrusion detection in the domino overlay system. In *Proceedings of NDSS*, San Diego, CA, February 2004.