Cluster Comput (2010) 13: 67–86
DOI 10.1007/s10586-009-0116-9

Cooperative detection and protection against network attacks using decentralized information sharing

Guangsen Zhang · Manish Parashar

Received: 10 March 2009 / Accepted: 8 December 2009 / Published online: 28 January 2010

Abstract Network attacks, such as distributed denial of service (DDoS) and Internet worms, are highly distributed and well coordinated offensive assaults on services, hosts, and the infrastructure of the Internet, and can have disastrous effects including financial losses and disruption of essential services. Consequently, effective defensive countermeasures against these attacks must provide equally sophisticated and well coordinated mechanisms for monitoring, analysis, and response. In this paper, we investigate techniques for cooperative attack detection and countermeasures using decentralized information sharing. The key underlying idea is the use of epidemic algorithms to share attack information and achieve quasi-global knowledge about attack behaviors. This paper first presents a conceptual model that defines the relationships between the level of knowledge in the distributed system and the accuracy of attack detection. The design of a cooperative attack detection and defense framework is then presented, and its use for detecting and defending against DDoS attacks and Internet worms is described. Simulation results are presented to demonstrate the feasibility and effectiveness of the framework against these attacks.

Keywords Cooperative network attack detection and defense · Decentralized information sharing · Epidemic algorithms · Gossip based communications · Peer-to-peer detection/defense overlay · DDoS · Internet worms

1 Introduction

As the Internet grows in size and complexity, its increased visibility and its diversity have attracted a variety of highly damaging attacks. These attacks can have catastrophic affects, including stolen or corrupted data, huge financial losses and disruption of essential services. For example, large scale epidemics caused by the unleashing of recent Internet worms, such as Code Red [6], have resulted in millions of host computer infections and over 2 billion dollars in financial loss [15]. In recent years, Internet attacks have been appearing with alarmingly regularity. Given their profound impact, protecting the computing infrastructure from such attacks has become a critical issue that needs to be urgently addressed.

Recent years have seen an increasing amount of research in local attack detection and defense mechanisms. Most of them can be classified as either signature based (misuse detection) or abnormal behavior based (anomaly detection) mechanisms. Misuse detection techniques perform pattern matching against a database of known attack signatures. Anomaly detection techniques establish statistical profiles of network traffic and flag any traffic deviating from the profile as anomalous. Although these local detection techniques are widely used to detect attacks, protecting networks from intrusions and attacks remains a significant challenge. A key

The research presented in this paper is supported in part by National Science Foundation via grants numbers IIP 0758566, CCF-0833039, DMS-0835436, CNS 0426354, IIS 0430826, and CNS 0723594, by Department of Energy via the grant number DE-FG02-06ER54857, by The Extreme Scale Systems Center at ORNL and the Department of Defense, and by an IBM Faculty Award, and was conducted as part of the NSF Center for Autonomic Computing at Rutgers University.

G. Zhang · M. Parashar (✉)
The Applied Software Systems Laboratory, Department of Electrical and Computer Engineering, Rutgers, The State University of New Jersey, 94 Brett Road, Piscataway, NJ 08854, USA
e-mail: parashar@rutgers.edu

G. Zhang
e-mail: gszhang@caip.rutgers.edu

reason is current Internet firewall or Intrusion Detection Systems (IDS) are located at the edges of the local network and have to detect attacks using only partial information about the observed attack. Such local detection mechanisms can have a high rate of false positives (legitimate connections that are misclassified as an attack) and can not be used to efficiently and accurately detect distributed or coordinated attacks. Instead, various isolated IDS should cooperate and share local information about network behaviors to defend against attacks. Recently, researchers have proposed data sharing mechanism to improve the accuracy and effectiveness of network detection systems. However, the design of effective cooperative attack detection and defense systems faces significant challenge as they must successfully detect and stop attacks, while at the same time preserve current performance guarantees to legitimate traffic. Key requirements of such a system include (1) *resilience* to failures and temporary instabilities and to the attacks themselves, (2) *decentralization* to provide greater flexibility and eliminate single points of failure, and (3) *low costs* to ensure that resource requirements of the detection and protection system do not degrade the performance of the deploying network.

Currently, most existing distributed intrusion detection systems process data centrally, despite distributed data collection [38]. This limits their scalability and their fault tolerance. Hierarchical designs have been introduced to address these issues [38]. Systems such as Emerald [34] and NetSTAT [45] have a layered structure where attacking events are filtered and preprocessed before they are forwarded to higher levels of the control hierarchy. However, these systems still use dedicated nodes that act as central points of control. As a result, the systems remain vulnerable to faults and have limited scalability. Recently, cooperative mechanisms have been designed to specifically defend against DDoS attacks and Internet worms. Cossack [29] addresses cooperative DDoS detection and Netbait [7] provides a structure overlay for sharing Internet worm information. Although such techniques provide improved DDoS and Internet worm detection capability over local attack detection techniques, they don't meet all the requirements discussed above, for example, they are not resilient to network congestion and their scalability is limited.

This paper presents a decentralized cooperative attack detection and countermeasure infrastructure. The primary objective of this research is to support the integration of existing network attack detection and defense system into a cooperative framework, and to enable them to efficiently share attack information to increase the accuracy and effectiveness of the detection and defense. Attack detection nodes are deployed at strategic points (e.g. hosts, routers, gateways) potentially sensitive to attack behavior. These nodes collaborate and exchange information in a peer-to-peer manner without a centralized coordinator, to achieve a quasi-global view of the overall network behavior and enable early and accurate detection of and reaction to attacks in the network.

Key components of the research presented in this paper are:

- *Conceptual Model:* How well an attack detection framework can detect an attack will depend on the amount of knowledge about the system that can be acquired. A conceptual analysis of the attack detection accuracy that can be achieved through information sharing in real distributed systems where gathering complete knowledge is unfeasible is presented.
- *Gossip Based Communication Mechanism:* Given the large scale of the Internet and critical nature of many of its applications, a resilient and scalable communication mechanism is needed to exchange attack information. A directional gossip mechanisms is designed, which meets this requirement, while reducing the overhead of information sharing.
- *Distributed Coordination Algorithm:* Instead of detecting attacks at central point of control, each detection node detects the network attacks individually. By using a distributed coordination algorithm, the detection nodes in the countermeasure framework can collectively detect and react to network attacks.
- *Detection Overlay:* The distributed decentralized framework presented is built as a peer-to-peer overlay network on top of the Internet and enables isolated local network detection agents at diverse locations to securely share intrusion information. The overlay design facilitates scalable information sharing, manages system heterogeneity participation and is resilient to failure.

The feasibility and effectiveness of the presented decentralized attack detection framework is demonstrated by using it to detect DDoS attacks and Internet worms. Simulation results for these case studies are presented.

The rest of the paper is organized as follows. Section 2 gives an overview of network attacks, discusses related work in DDoS and Internet worm defense, and introduces gossip-based communication mechanisms. Section 3 provides a conceptual model for the proposed distributed decentralized attack detection framework. Section 4 explains the distributed decentralized network attack detection framework in detail. Section 5 describes use of the framework for detecting DDoS attacks. Section 6 describes the use of framework to detect Internet worms. Section 7 concludes this paper.

2 Background and related work

Network attacks are primarily a result of software vulnerability, protocol vulnerability or infrastructure vulnerability. Software vulnerabilities are due to the buggy implementation of the network software (e.g., DNS Bind, HTTP Apache, Telnet, SMTP etc.) [18]. Protocol vulnerability is the exploitation of logical loopholes in protocol state models. Infrastructure vulnerabilities are caused by exploiting the inter-dependencies among network components to attack network resources that provide critical services. This section analyzes requirements and approaches for network attack detection and defense. Specifically, it focuses on two kind of network attacks: Distributed Denial of Service and Internet worms. The requirements and challenges of cooperative network defense are also presented. Finally, the section introduces conceptual and technology building blocks used by the cooperative attack detection and countermeasure infrastructure presented in this paper.

2.1 Distributed denial of service

Distributed denial of service attacks (DDoS) pose a great threat to the Internet. A recent DDoS attack occurred on October 20, 2002 against the 13 root servers that provide the Domain Name System (DNS) service to Internet users around the world. Although the attack only lasted for an hour and the effects were hardly noticeable to the average Internet user, it caused 7 of the 13 root servers to shut down demonstrating the vulnerability of the Internet to DDoS attacks [15]. As one of the most difficult problems in network security, DDoS attacks have become a serious threat to the availability of Internet Services. Distributed denial of service attacks occur when numerous subverted machines (zombies) generate a large volume of coordinated traffic toward a target, overwhelming its resources. DDoS attacks are advanced methods of attacking a network system to make it unavailable to legitimate network users. DDoS attacks are becoming an increasing threat to the Internet due to the easy availability of user friendly attack tools, which help to coordinate and execute a large scale DDoS attack. Even an unsophisticated individual can launch a devastating attack with the help of these tools. Available tools (e.g., Trinoo, TFN, TFN2K, Shaft, and Stacheldraht) have been used in DDoS attacks against well-known commercial web sites, such as Yahoo, Amazon and Ebay [11]. Furthermore, attackers need not fear punishment as it is extremely difficult to trace back the attack and locate the agent machines or the attacker who infected them.

The mechanism of DDoS attacks works as follows: the master (a host running applications that can be used to initiate attacks) sends control packets to previously compromised slaves (a host that runs processes responsible for receiving and carrying out commands issued by the master), instructing them to target a specified victim. The slaves then generate and send high volume streams of messages to the victim using fake or randomized source addresses, so that the victim cannot locate the attackers. There are two main classes of DDoS attacks: bandwidth depletion and resource depletion attacks. A bandwidth depletion attack is designed to flood the victim network with unwanted traffic that prevents legitimate traffic from reaching the victim. Resource depletion attacks are designed to exploit operating system vulnerabilities at the victim. The increase in DDoS attacks in the Internet has resulted in the development and deployment of many DDoS detection and defense mechanisms. These mechanisms can be broadly classified as: Proactive Mechanism, Reactive Mechanism, Tolerance Mechanism and Post Attack Analysis. The classes are briefly introduced below and are summarized in Table 1 (for a more detailed discussion see [52]).

- **Proactive Mechanisms.** The motivation for proactive defense mechanisms is based on the observation that it is hard to detect DDoS attacks. So instead of detecting the attacks using signatures (attack pattern) or anomaly behavior, these approaches try to improve the reliability of the global Internet infrastructure by adding extra functionality to Internet components to prevent attacks and vulnerability exploitation. The primary goal is to make the infrastructure immune to the attacks and to continue to provide service to normal users under extreme conditions.
- **Reactive Mechanisms.** These mechanisms typically deploy third-party Intrusion Detection Systems (IDS) to obtain attack information and take action based on this information. Consequently their usefulness depends on the capability of the IDS systems. Different strategies are used based on the assumptions made by the IDS systems. If the IDS system can detect the DDoS attack packets accurately, filtering mechanism are used, which can filter out the attack stream completely, even at the source network. If the IDS can not detect the attack stream accurately, rate limiting is used. This mechanism imposes a rate limit on the stream that is characterized as malicious by the IDS.
- **Tolerance Mechanisms.** Observing that it is hard to distinguish DDoS attack packets from legitimate ones, researchers have proposed attack tolerance mechanisms. These approaches are based on rate control to enforce fairness in bandwidth allocation and thus minimize the damage caused by DDoS attacks.
- **Post Attack Analysis.** The purpose of post attack analysis is to either look for attack patterns that will be used by an IDS, or to identify attackers using packet tracing. Since the source addresses of flooding packets are faked, various trace back techniques have been proposed to find out the origin of a flooding source. The goal of packet tracing is to trace Internet traffic back to the true source (not spoofed IP address).

The mechanisms described so far can be used to protect Internet from particular DDoS attacks. However, given the dynamic nature of Internet as well as the diversity patterns of DDoS attack traffics, single deployment of these detection systems can not accurately detect general DDoS attacks. To improve the defense efficiency, we need a paradigm shift. In-

Table 1 Taxonomy of DDoS detection and defense

Detection Classification	Related Efforts	Approach Descriptions	Issues
Proactive Mechanism	Secure Overlay Service [8]	Prevents denial-of-service attacks on critical servers by routing requests from previously authenticated clients to the servers via an overlay network	Nodes must be part of the SOS; cannot be used for general DDoS
	Ingress or Egress Filtering [14]	Checks a packet for its source IP address and block packets that come from an address beyond the router's possible ingress address range	Per-flow filtering only
	Router Based Packet Filtering [31]	Employs a number of distributed packet filters to examine whether each received packet comes from a correct link according to inscribed source and destination addresses, and BGP routing information	Legitimate packet may be dropped
Reactive Mechanism	D-WARD [25]	The system is located at the source network router (either LAN or border router), and autonomously detects and suppresses DDoS flows originating at this network	Can have high false detection rates
	History Based IP Filtering [32]	Admits the incoming packets based on a pre-built IP address database, which is built using previous connection history	Legitimate packet may be dropped
	Center Track [42]	Creates an overlay network of IP tunnels by linking all edge routers to central tracking routers, and all suspicious traffic is rerouted from edge routers to these tracking routers	Can have high overheads
	MULTOPS [16]	Builds traffic profile based on the assumption that during normal operations on the Internet, the packet rate of traffic going in one direction is proportional to the packet rate of traffic going in the opposite direction	Can have high false detection rates
Tolerance Mechanism	Pushback [19, 24]	Augments routers with the capability to detect and control flows that create congestion using a Pushback daemon	Requires new router capability
	Max-min Fair Server Centric Throttling [50]	Uses max-min fair server-centric router throttles and involves the server under stress installing rate throttles at a subset of its upstream routers	Legitimate packet may be dropped
Post Attack Analysis	Trace back Related Efforts [4, 9, 30, 39, 41, 49]	Uses router-generated trace back messages to reconstruct the attack paths	Efficient for DoS, infeasible for large scale DDoS attacks

stead of building detection systems that operate in isolation, a distributed framework of detection nodes where heterogeneous systems can plug in and cooperate to achieve a better overall protection is required.

2.2 Internet worms

Internet Worms represent another class of major network attacks against Internet. In addition to their primary objective of replicating and propagating themselves, Internet worms can also achieve DDoS attacks by flooding a selected target. In response to recent Internet worm attacks, a number of defense mechanisms have been proposed. These approaches can be broadly classified based on the methods used to stop worm propagation as prevention, treatment and containment.

Prevention technologies aim at reducing the size of the vulnerable population, thereby limiting the spread of a worm outbreak. In the Internet context, the vulnerability of the population is a function of the software engineering practices that produce security vulnerabilities as well as the social and economic conditions that ensure the homogeneity of the software base. For example, a single vulnerability in a popular software system can translate into millions of vulnerable hosts. While there is an important research agenda, initiated in [40], to increase the security and heterogeneity of software systems on the Internet, widespread software vulnerabilities will persist for the foreseeable future. Therefore, pro-active prevention measures alone are unlikely to be sufficient to counter the worm threat.

Treatment technologies, as exemplified by the disinfection tools found in commercial virus detectors and the system update features in popular operating systems, are an important part of any long-term strategy against Internet pathogens. By deploying such measures on hosts in response to a worm outbreak, it is possible to reduce the vulnerable population (by eliminating the vulnerability exploited by the worm) and reduce the rate of infection (by removing the worm itself from infected hosts). However, for practical reasons, these solutions are unlikely to provide short-term relief during an acute outbreak. The time required to design, develop and test a security update is limited by human time scales, which is usually measured in days and far too slow to have significant impact on an actively spreading Internet worm. Worse, if the installation of such updates is not automated, the response time can be substantially longer. For example, during the Code-Red epidemic, it took sixteen days for most hosts to eliminate the underlying vulnerability and thousands had not patched their systems six weeks later [27]. Finally, creating a central authority for developing, distributing, and automatically installing security updates across hundreds of thousands of organizations will require a level of trust and coordination that does not currently exist [28].

Finally, *containment* technologies, as exemplified by firewalls, content filters, and automated routing blacklists, can be used to block infectious communication between infected and uninfected hosts [7, 51]. In principal, this approach can quickly reduce, or even stop, the spread of infection, thereby mitigating the overall threat and providing additional time for more heavy-weight treatment measures to be developed and deployed. During the Code-Red epidemic, ad-hoc containment mechanisms were the primary means used to protect individual networks (e.g., by blocking inbound access to TCP port 80, or content filtering based on Code-Red specific signatures), or isolating infected hosts (e.g., by blocking the host's outbound access to TCP port 80). These solutions were implemented manually using existing routers, firewalls, and proxy servers. While these limited quarantines did not halt the spread of the worm, they provided limited protection to portions of the Internet.

There are strong reasons to believe that containment is the most viable of these strategies. First, there is hope that containment can be completely automated, since detecting and characterizing a worm, which is required before any filtering or blocking can be deployed, is far easier than analyzing the worm itself for the vulnerability being exploited and creating software to patch the problem. Second, since containment can potentially be deployed in the network it is possible to implement a solution without requiring universal deployment on every Internet host. Current worm containment efforts are summarized in Table 2 (for a more detailed discussion see [52]). We believe that cooperative containment strategies where defense nodes can share containment information can significant increase effectiveness of Internet worm detection and defense strategies.

2.3 Cooperative defense mechanisms

Cooperative defense mechanism enable individual detection nodes to share information about attack behaviors, and can lead to more accurate and effective detection and defense strategies. Such strategies have been investigated by several researchers in recent years. Approaches most related to the research presented in this paper are discussed below.

DIDS [38] is a distributed intrusion detection system consisting of host managers and LAN managers that are responsible for distributed data monitoring and sending notable events to the director. The centralized director then analyzes these events to determine the security of the system as a whole. The centralized director is clearly the bottleneck in this system and limits its scalability. Furthermore, the centralized director is also a single point of failure.

Emerald [34] enables the exchange of security incident related information between different domains or large networks. It uses a layered hierarchical architecture to provide high volume event analysis. Similar with DIDS, it needs a centralized director to make final decision, which can limit its scalability.

Dshield [12] enables firewall users to share intrusion detection information. Users contribute intrusion detection information by running client programs that collect local firewall log files and submit them to the DShield team via email or a web-based interface. Contributed log files are stored in a centralized database which can then be queried to determine if a particular machine has been compromised and to help produce summary reports of various Internet worm epidemics. The system collects information in a centralized database and can not respond to attacks in real time.

NetBait [7] is a distributed system that provides detailed information about network intrusions. It collects data from geographically located machines, which use traditional intrusion detection systems (such as Snort [35]) to discover worm attacks. The goal of NetBait is to provide accurate information to identify infected hosts and expedite the process of worm containment and cleanup.

While the cooperative mechanisms listed above collect network attack information from distributed sites, these systems process information and detect attacks at centralized node. As a result, these systems are not resilient and have limited scalability. The goal of the framework presented in this paper is to address these issues by building on a decentralized and resilient communication mechanism, which is introduced below.

2.4 Gossip based mechanism for decentralized information sharing

Designing an effective decentralized network attack detection framework in large distributed environments requires a scalable and resilient information sharing mechanism. The mechanism should be easy to deploy, robust, and highly resilient to failures. Gossip based mechanisms provide potentially effective solutions that meet these requirements. These

Table 2 Internet worm containment projects

Projects	Approach Descriptions	Issues
Early Bird System at UCSD [37]	Uses a content sifting approach to detect content prevalence and use scaled bitmaps to estimate address dispersion	Supports distribution through the centralized worm detector only
Autograph at Carnegie Mellon University [21]	Automatically generates signatures from worms propagating using TCP transport	Uses local traffic, and as a result, only has a partial observation of worm traffic
Fast Scan Detection at Berkeley ICSI [48]	A scan detection and worm suppression algorithm based on Threshold Random Walk (TRW) for hardware and software implementations	It is a local detection mechanism. It Proposes cooperative detection, however no detailed investigation of the cooperative approach is provided
Network Worm Vaccine at Columbia [36]	Patches vulnerable software using a collection of sensors that detect and capture potential worm infection vectors based on a set of heuristics, to test the resistance of patched applications against these infection vectors	Underlying assumptions limit its usage
Shield project at Microsoft [47]	Installs vulnerability specific solutions and exploits generic network filters at end systems	Can have high false detection rates

mechanisms build upon epidemic algorithms, which mimic the spread of a contagious disease. One may consider dissemination of information in a network to be similar to the spread of a rumor or of an infectious disease in a society, which have been extensively studied by applied mathematicians. Once it has started, an epidemic is hard to eradicate: it only takes a few people to spread a disease, directly or indirectly, to the community at large. An epidemic is also highly resilient in that even if many infected people die before they transmit the contagion or they are immunized, the epidemic continues to propagate through the population. It is possible to adjust the parameters of an epidemic algorithm to achieve high reliability despite process crashes and disconnections, packet losses, and a dynamic network topology.

Epidemic algorithms have been applied to solving several practical problems such as database replication [10], failure detection [44] and resource monitoring [43]. A large body of theoretical work is also available due to the general importance of understanding epidemics and its close relation to random graph theory.

The power of gossip is that information reaches its destination by following a diversity of paths. In effect, every process is a potential source of data for every other process, and over time the number of possible routes by which information from process p might travel to process q increases exponentially. For example, suppose that process p has detected an event of interest, after t time units, $O(2^t)$ processes will know that event [43]. When a system comes under attack, an adversary may manage to corrupt a few messages or disrupt a region of the network. However, if any connectivity remains at all, the gossip exchange of data will eventually prevail, and data stored within the infrastructure will reach all sites in the system. Thus, gossip protocols are relatively tolerant of denial of service attacks.

Based on these characteristics, the gossip based communication is used as the mechanism for information sharing in the decentralized network attack detection and defense framework presented in this paper. Specifically, the framework uses an epidemic algorithm based on the infect forever model [13] where infected individuals remain infectious throughout. A quantity of interest in this model is the number Z_r of individuals infected prior to round r of the spread of the epidemic. Two key measures of the "success" of an epidemic dissemination model are (1) the proportion of infected processes defined as $Y_r = Z_r/n$ (n is the total number of processes) and (2) the latency for the infection to reach other processes defined as number of rounds R necessary to infect the entire system. The analysis presented in [13] shows that:

- Assuming that infectious individuals try to contaminate f other processes in each round, the expected fraction of infected processes after r rounds is:

$$Y_r \approx \frac{1}{1+ne^{-fr}}.$$

  Thus, the ratio of infected individuals to uninfected processes increases exponentially on average, by a factor of e^f in each round.
- The latency, R, for a rumor to reach every process assuming that each infectious process tries to contaminate f other processes in each round, satisfies the relation [33]:

$$R = \log_{f+1}(n) + \frac{1}{f}\log(n) + O(1).$$

  Thus the epidemic spreads quickly, taking at most a logarithmic number of steps to reach every process.

3 A conceptual model for decentralized information sharing

A critical issue in realizing a decentralized cooperative attack detection and defense framework is acquiring sufficient knowledge about the distributed network attack to be able to accurately detect it and effectively defend against it. Therefore, a conceptual model that defines the relationships between the level of knowledge acquired by the nodes in the detection framework and attack detection capability that can be achieved, is helpful and necessary. This section presents such a decentralized information sharing model. The section first defines the knowledge requirement for cooperative attack detection and describes the relationship between the level of knowledge in the system and the corresponding attack detection capability. It then discusses the attack detection capability that can be achieved with knowledge acquired using a gossip based communication mechanism, and analyzes the associated aggregation latencies.

3.1 Knowledge requirement for attack detection

The key requirement for a decentralized cooperative attack detection framework is the capability to acquire sufficient knowledge about distributed attacks. If all the nodes in this distributed framework have common knowledge [17] about the network attack behaviors, which means that every detection node knows the overall network attack behavior, and it also knows that all other nodes have this same knowledge, then network attacks can be perfectly detected. However, achieving common knowledge requires completely synchronized and reliable communication, which is not feasible in a practical distributed system. Rather than achieving perfect common knowledge, this section investigates the level of knowledge that can be practically acquired and the corresponding attack detection capability that can be achieved given these level of knowledge. To provide a basis for this discussion, a more formal definition of a knowledge hierarchy is presented below.

In a distributed decentralized attack detection system, each detection node i will only have a partial view of the system. Assuming that this view is correct and can be trusted, this partial view can be thought of a fact represented as p_i. The overall knowledge in the system can then be defined as $p \equiv \bigwedge_{i \in G} p_i$, where G represent all the detection nodes. Based on the discussions in [17], the different levels of knowledge in the system are as follows:

- *Local Knowledge:* Each detection node i only has its own observation, i.e., p_i. In this case, the knowledge is partial and therefore the detection based on this knowledge will be error prone.
- *Distributed Knowledge:* Each detection node not only has its own observation p_i, it also has some knowledge p_j, where $j \neq i$. However, there is knowledge p_k which is not known to i ($k \in G$ and $k \neq i \neq j$).
- *Quasi-global Knowledge:* Every detection node has knowledge p, but this is the knowledge about the system from t seconds before. If duration of the attacks is longer than t, this Quasi-global knowledge can be used to make decisions to effectively defend against these attacks.
- *Global Knowledge:* Every detection node has knowledge p.
- *Common Knowledge:* Every detection node has knowledge p, and also knows that other nodes have knowledge p, and that the other nodes know that other nodes have knowledge p.

3.2 Levels of knowledge and attack detection capability

In a distributed decentralized attack detection system, initially, each individual detection node only has *local knowledge* about the network behavior. As a result, such systems will allow a large number of distributed network attacks to pass unnoticed. Sharing this local knowledge between detection nodes using only unreliable communication mechanism, the system can achieve *distributed knowledge*. In this case, each detection node gets a partial view of the network behavior and possible distributed network attacks, and as a result, attack detection will have high false rates. Using an asynchronous, resilient communication mechanism to share local knowledge, the system can achieve *quasi-global knowledge*. With this knowledge, every detection node can acquire sufficient information about attacks and as a result, the attacks can be detected effectively. However, due to the time required to acquire this knowledge, there will be a delay between the time of the attack and the time it is detected. Better attack detection capability requiring higher levels of knowledge in the knowledge hierarchy. Acquiring *global knowledge* requires synchronous communications. Such a framework will incur significant overheads but can achieve a better attack detection capabilities. Ideally, if the system can achieve *common knowledge*, then all the network attacks could be perfectly detected. However, this requires synchronous reliable communication which is not feasible in real distributed system [17].

The discussion above shows that cooperative attack detection and defense capability depends on the communication mechanism used to acquire the knowledge about the attacks, and that effective detection and defense still can be achieved with quasi-global knowledge. Further, asynchronous communication mechanisms can be used to achieve this knowledge with low overheads. Therefore, it is feasible to design a distributed decentralized attack detection framework that achieves quasi-global knowledge. The key is to select the appropriate communication mechanism. As both the DDoS attacks and Internet worms only need short time to achieve significant damage to network infrastructure, the delay δ in acquiring quasi-global knowledge should be less than the durations of network attacks. Note that, when the network is under an attack, network bandwidth is a critical resource and the communication mechanism selected should introduce minimum overheads while providing sufficient level of knowledge to support attack detection. Gossip based communication mechanisms are scalable, resilient and have low overheads, and can be potentially used to share attack information. This is explore further in the following sections.

3.3 Latency of network attack information aggregation using gossip

As discussed above, the attack information aggregation latency effects the accuracy and effectiveness of attack detection. If the communication mechanism used to share information takes a long time for every defense node to acquire the required knowledge about the network attack behavior, the attack may have already caused significant damage to the network infrastructure before that nodes can initiate any defense. This section analyzes the aggregation latency of gossip based communication mechanisms. As discussed above, gossip based communication is quite similar to that of broadcasting by means of the push-pull anti-entropy epidemic protocol presented in [10]. The analysis presented is based on a modified and generalized push-pull anti-entropy protocol.

In the push-pull anti-entropy gossip model, each node participating in group communication either periodically send its own information to its neighbors or queries its neighbors to acquire up to date information. Assuming that each network defense node n_i maintains a neighbor list, which is a random and uniform sample of the defense nodes in the system. Further, each node n_i has a numeric attribute a_i, which is the confidence value with which the detection node suspects an attack. The aggregation is then performed over the set of these values. Each node n_i also stores an approximation x_i of the aggregate. The algorithm for the push-pull anti-entropy gossip is presented in Fig. 1. We introduce the following notations [3]:

$$\mu_i = \mu_{\alpha_i} = \frac{1}{N}\sum \alpha_{i,k}$$

$$\delta_i^2 = \delta_{\alpha_i}^2 = \frac{1}{N}\sum (\alpha_{i,k} - \mu_i)^2$$

Here, N is the number of defense nodes in the system that share information, and α_i is the information to be aggregated. Without loss of generality, it can be assumed that the common expected value of the elements of α_0 is zero. The purpose for this assumption is to simplify the expressions.

```
// the active process of the protocol on node ni
**do** forever
wait(getWaitingTime())
nj = selectRandomNeighbor()
// perform elementary aggregation step
send xi to nj
receive xj from nj
xi = aggregate(xi ; xj )
// reply on node nj
receiveApproximation(xi; ni)
send xj to ni
xj = aggregate(xj ; xi)
```

Fig. 1 Push-pull anti-entropy aggregation protocol

In particular, for any vector α, if the elements of α are independent random variables with zero expected value then

$$E(\delta_\alpha^2) = \frac{1}{N}\sum E(\alpha_k^2)$$

Furthermore, the elementary variance reduction step in which both selected elements are replaced by their average does not change the sum of the elements in the vector, so $\mu_i \equiv \mu_0$ for all cycles $i = 1, 2, \ldots$. This property is very important because it guarantees that the algorithm does not introduce any errors into the approximation. This means that from now on we can focus on variance. Clearly, if the expected value of δ_i^2 tends to zero with i tending to infinity then the variance of all vector elements will tend to zero as well, so the correct average μ_i will be approximated locally with arbitrary accuracy by each node. As proved in [20], the expected value of variance reduction during one cycle is given by

$$E(\delta_{i+1}^2) \approx E(2^{-\phi})\frac{1}{N}\sum E(\alpha_{i,k}^2) = E(2^{-\phi})E(\delta_i^2)$$

where

$$E(2^{-\phi}) = \frac{1}{2\sqrt{e}}$$

The formulae presented above indicate that the gossip based information sharing mechanism will converge exponentially toward global knowledge of network attack behavior distributed across the network. In a real deployment of this communication mechanism, several parameters have to be tuned to make the gossip based communication mechanism efficient and have low overheads. If the attacks are aggressive and propagate very quickly, the information dissemination interval should be very short. As a result, network attack information will be aggregated in a short time. However, these communications will occupy a larger portion of network bandwidth than the case where the information dissemination intervals are long. In a real design, the values of such parameters should be selected to balance the speed of convergence of information aggregation and the cost of the associated communication. Sections 5 and 6 discuss the selection of these parameters in the cases of DDoS and Internet worm attacks.

Note that the analysis presented in this section is based on the assumption that the neighbors selected by a defense node when initiating communication is a uniform random sample off all peer defense nodes. A discussion of sensitivity of aggregation to the selection of peers is presented in [3]. It shows that the aggregation scheme have same performance with proper selected parameters.

3.4 Acquiring quasi global knowledge using gossip-based communications

This section shows that quasi global knowledge can be achieved using gossip-based communications in a practical distributed systems. Note that in a distributed system messages sent out using gossip mechanisms are not guaranteed to be received by all the network defense nodes. Let q_i denotes the probability that one defense node does not get the network attack information in round i. Then $q_0 = 1 - 1/N$ (among N nodes, only one node knows the information at round 0), where N is the total number of defense nodes in the system. Here, q_{i+1} is expressed as a function of q_i. Clearly, a defense node will not know the global network attack information in round $i + 1$ if the following is true:

(1) It did not know it in round i.
(2) The neighbor nodes that it chose did not know this information either.

Formally, we can get the following equation [3]:

$$q_{i+1} = q_i q_i \left(1 - \frac{1}{N}\right)^{N(1-q_i)}$$

As $(1 - \frac{1}{N})^{N(1-q_i)} < 1$, from this equation it follows that

$$q_{i+1} < q_i^2 < q_0^{2^{i+1}} = \left(1 - \frac{1}{N}\right)^{2^{i+1}}$$

This result suggests that q_i decreases super-exponentially. As a result, a message will eventually reach all the defense nodes. Further, in each round of gossiping, an individual defense nodes knows that every other defense node either has already received the network attack information or will eventually get this information. Let $\varepsilon(i)$ and p_i be defined such that in round i, all the defense nodes in the system get the network attack information between t_0 and $t_0 + \varepsilon(i)$ time units with the probability p_i. Figure 2 illustrates the speed of convergence of the gossip based aggregation as a function of p_i for different values of N. For $n = 10000$ nodes, it takes around 15 rounds for the mechanism to converge. If the interval between two rounds is 10 seconds, then it will

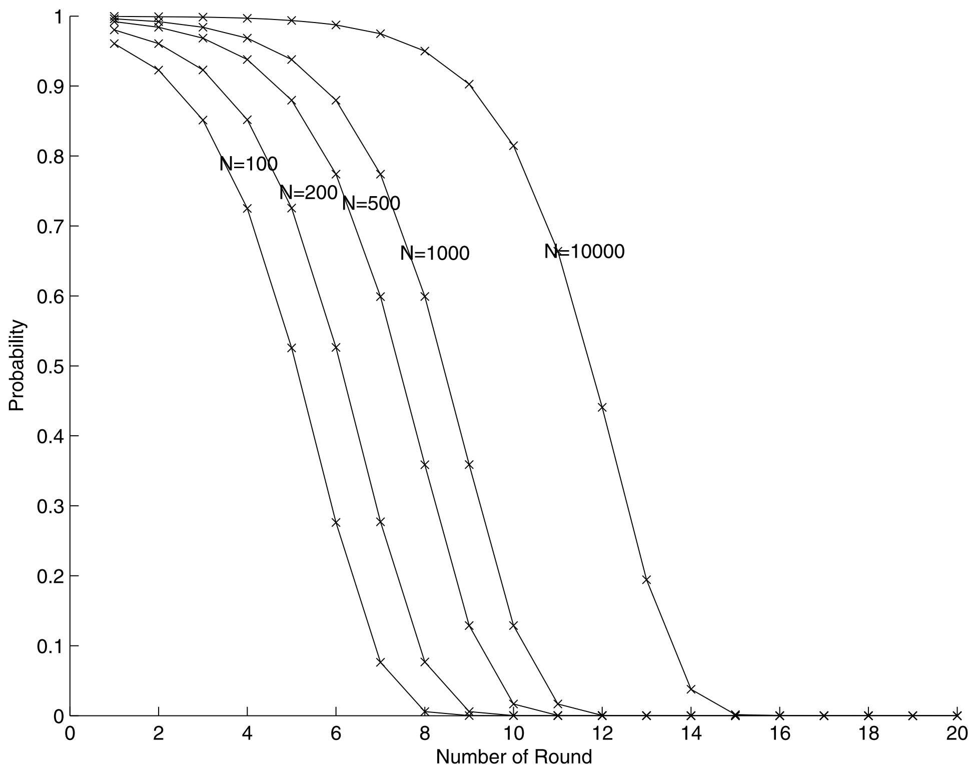


Fig. 2 Convergence speed of aggregations using gossip-based information sharing

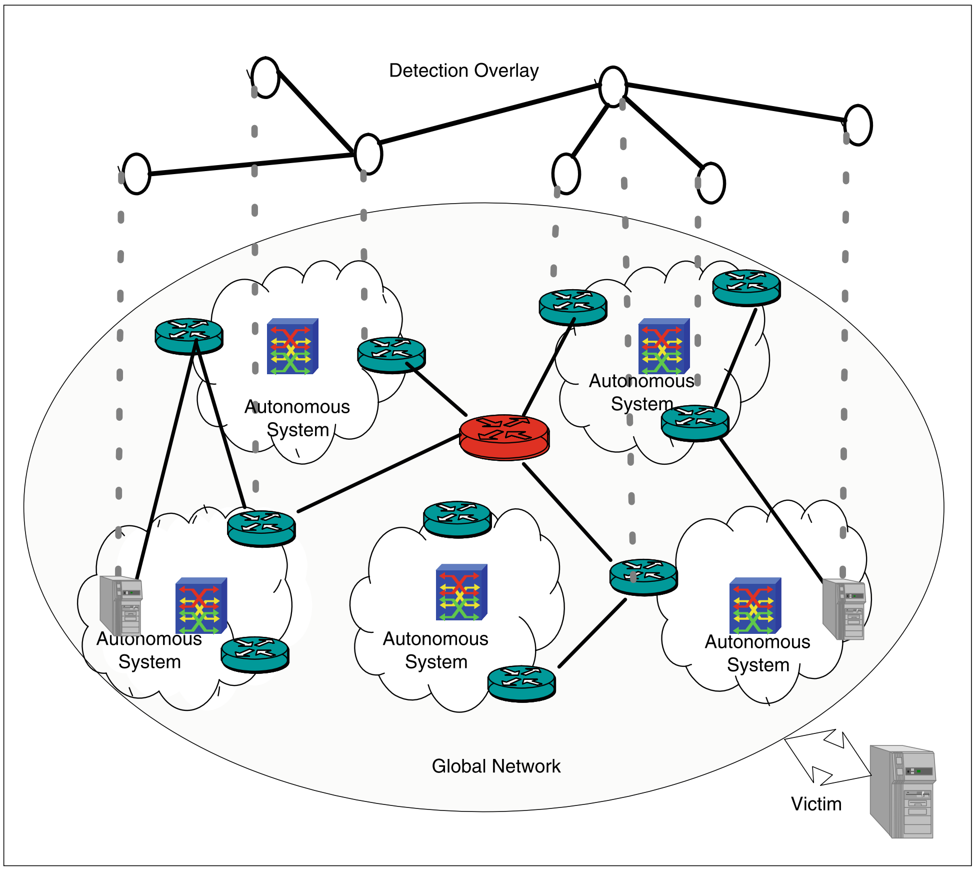
take about 2.5 minutes for all the nodes to get the attack information. The interval between two rounds can be tuned to balance the performance cost and defence efficiency. As presented by David Moore in [28], the duration of most attacks is longer than 5 minutes. As a results, given appropriate parameters, cooperation using gossip based mechanism can be effective against these network attacks.

4 A framework for decentralized cooperative detection and defense against network attacks

The conceptual model and analysis presented in Sect. 3 demonstrated that it is feasible to build a decentralized cooperative defense system, based on gossip-based communication mechanisms, that can detect and defend against network attacks. This section presents the design of such decentralized cooperative detection and defense framework. The framework is composed of local detection nodes that are placed at "strategic" locations in the Internet and that non-intrusively monitor and analyze the passing traffic for possible attacks. Local detection nodes, which can be are routers with attack detection and attack packets filtering functionality, form an secure and resilient, self-organizing unstructured overlay network over the Internet [5]. Attack detection consists of two key stages. In the first stage, each local detection node detects traffic anomalies using various intrusion detection mechanisms. Due to the dynamic and distributed nature of the attacks, detections based on these mechanisms alone will have high false positives. In the second phase, the detection nodes share attack information with other nodes in the overlay network using gossip based communication. The overall architecture of the framework is illustrated in the Fig. 3. The overall operation of the framework is summarized below. The key steps are presented in detail in the following sections.

- *Local Network Attack Detection:* Each detection node monitors the immediate network around it for possible attacks using existing local detection mechanisms. For example, to detect DDoS attacks, each node keeps traffic statistics for high-traffic destinations using sample-and-hold algorithms. If the traffic statistics deviate from the normal profile, the local node will raise an alarm to report attacks. To detect Internet worms, the node can either use a database of worm signatures or monitoring packets toward unused IP addresses to identify malicious packet flows.
- *Information Sharing and Global Detection:* When a local detection node detects a possible network attack, it shares this information with other nodes using a gossip protocol based on epidemic algorithms. The information shared may be the node's confidence that certain target

Fig. 3 Architecture of the decentralized cooperative detection and defense framework



machines are under DDoS attack, or the signatures identified as Internet worm attacks. As discussed in Sect. 3, the gossip based information sharing mechanism will converge exponentially, after which, each node will have sufficient information about the network attacks and will also know that other nodes have this same information, and can make decisions about the network attacks.

- *Cooperative Defense:* Detection nodes will cooperate with each other to defend against confirmed network attacks. When a node confirms the occurrence of a network attack, it will locally deploy countermeasures to prevent the attack from continuing, and will notify other nodes about the attack. These nodes will also perform similar actions. This process will continue until the attack traffic is effectively blocked. The approach can be combined with available mitigating or rate limit technologies to eliminate the attack before it does significant damage.

4.1 Local network attack detection

There are several techniques for local network attack detection, such as misuse detection, statistical anomaly detection, information retrieval, data mining and inductive learning. The internals of an individual local detection node can be fairly complex, but conceptually it can be structured into six pieces, as shown in the Fig. 4. The traffic measurement module is responsible for measuring local traffic. Next, the local detection mechanism will use this data to detect any local anomaly. This local decision will be sent to the cooperative detection engine, which will combine this local decision with the decisions from neighboring nodes using the message dissemination module, to make a global detection decision. Finally, the detection decision module will inform the attack defense module to take action to defend against an attack.

When the local detection node detects a network attack, it will generate an alert message in the form of a tuple (*conf*, *dest*), where the *conf* is the confidence of the detection node on this alert message and the *dest* is the target of the attack. This message will be aggregated using the decentralized attack information sharing mechanism.

4.2 Attack information sharing

A key requirement for network attack detection is low false positive rates, calculated as the percentage of normalcy vari-

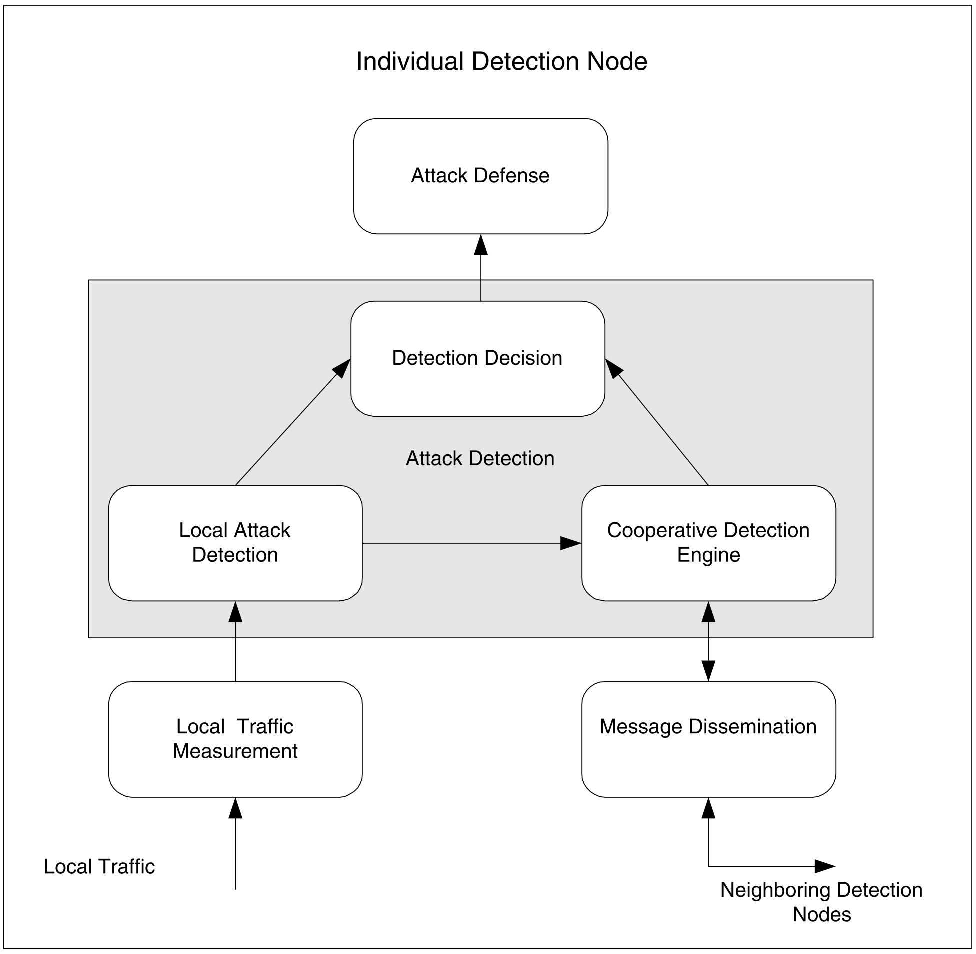

Individual Detection Node
Attack Defense
Detection Decision
Attack Detection
Local Attack Detection
Cooperative Detection Engine
Local Traffic Measurement
Message Dissemination
Local Traffic
Neighboring Detection Nodes

Fig. 4 A conceptual architecture of an individual detection node

```
when  ( node n builds a new  (conf, dest) pair)
{
        while  ( node n believes that not enough of its
        neighbors have received  (const, dest) pair)
        {
                m =a neighbor node of p;
                send  (conf, dest) pair to m;
        }
}
```

Fig. 5 Gossip protocol for cooperation

ations detected as anomalies, and low false negative rate, calculated as the percentage of anomalies detected as normalcies. In the framework presented in this paper there are two factors that will affect system performance: the overhead of the information sharing mechanism, and the level of knowledge on the network attack. Communication bandwidth is often a scarce resource during the network attack, so the attack information sharing should involve only a small number of messages. In particular, any protocol collecting all local data at a single node will create communication bottlenecks or a message implosion at that node. Based on the discussion in Sect. 3, gossip based protocols are resilient and scalable while providing sufficient information for attack detection, and are uses for the information sharing purpose. The structure of the gossip protocol running at each node n is shown in Fig. 5.

Compared to multicast or broadcast protocols, the gossip protocol has a smaller overhead. However, it requires a longer time for each node get the message. A possible variant is directional gossip [22], which reduces the latency and communication overhead of traditional gossip protocols. The cooperative detection and defense framework presented here uses a modified directional gossip strategy, which is effective for defense against DDoS attacks. A gossip strategy for defense against Internet worms will be presented in Sect. 6. Assuming that each node knows its immediate neighbors in the overlay network, the modified directional gossiping protocol is as follows: An individual node sends the $(conf, dest)$ pair to the node on its path to the destination target node with probability 1. It forwards the $(conf, dest)$ pair to all other nodes at random. At anytime t, each node i maintains a list of $(conf_k, dest_k)$ pairs. The algorithm is described below:

(1) Every node in the overlay network runs the aggregation protocol and makes a global decision in the period T_g.
   (a) Let $(conf_{r,k}, dest_{r,k})$ be all pairs sent to node i in round t, where $t \leq N$ (N is total number of rounds in the period T_g, k is the index of node and r is the gossip round).

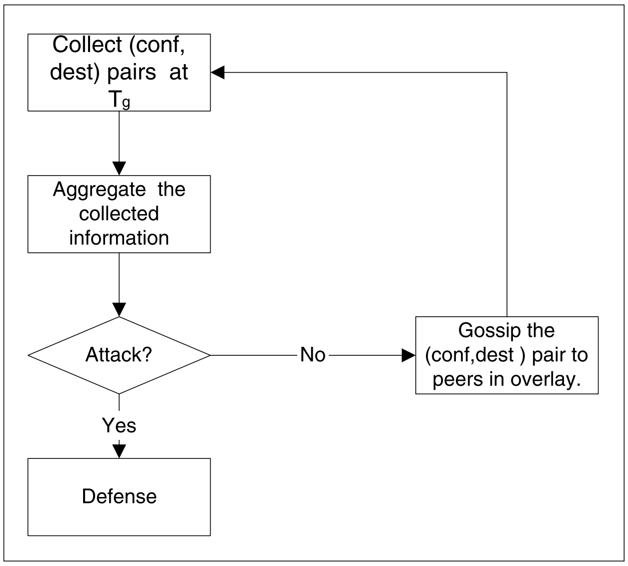

Collect (conf, dest) pairs at T_g
Aggregate the collected information
Attack?
No
Gossip the (conf,dest) pair to peers in overlay.
Yes
Defense

Fig. 6 Flowchart for the modified directional gossip communication algorithm

   (b) Let $d_{t,i} = \frac{\Sigma_r conf_{r,k}}{m}$, where m is the number of messages received and $d_{t,i}$ is the aggregated information.

(2) Query the routing table, find out the next hop to $dest_{t,i}$, send the pair $(d_{t,i}, dest_{t,i})$ to that node with probability 1. Send the pair to other neighbors with probability p.

(3) At round N, Compare $d_{N,i}$ with $Threshold_i$. If $d_{N,i} > Threshold_i$, then $dest_{N,i}$ is under attack. Otherwise, set $d_{N,i} = new$ local detection confidence value. Begin new round of attack information aggregation.

The algorithm is illustrated in the flowchart in Fig. 6.

4.3 Discussion

In addition to improving the accuracy of countermeasure against network attacks, the cooperative detection and defense framework presented in this paper has several advantages including improved detection accuracy, quicker response to attacks, and reduced bandwidth consumption by the attack traffic. Further, since defense is distributed across the detection nodes and reduces the load at each individual node. However, the framework does impose overheads on the network. These overheads include computational overhead imposed by attack detection and attack response enforcement, storage requirement to save logs for attack detection, and communications overhead used to send control messages to distributed locations in a network. An important issues that should be addressed include the establishing of trust between detection nodes.

5 Cooperative DDoS defense

In a distributed denial of service (DDoS) attack, the attack traffic flows across the Internet toward the victim, and as a result, the victim can easily detect the attack by observing its degraded service. However, it is too late to defend against DDoS attacks near the victim as the victim resources would be heavily loaded and would not be able to react to the DDoS attacks. The attacks should ideally be stopped as close to the sources as possible, saving network resources and reducing congestion. However, there are no common characteristics of DDoS streams that can be used to detect and filter them near the source. The cooperative defense framework detects and defends against DDoS attacks in the intermediate network by observing that, in the intermediate network, the aggregated attack flows toward the victim consume more bandwidth than aggregated normal flows to the victim. Since the flows do not cause congestion in the network, and it is hard to detect the DDoS attacks in a single domain, information sharing between distributed domains is used to detect DDoS attack early.

Based on the framework discussed in Sect. 4, we present a DDoS defense case study in this section. First the local detection node monitors all traffic passing through it and maintains statistics for sampled flows grouped by their address prefix. It then uses an existing IDS at each detection node to locally detect attacks. The prototype framework in this paper uses CUSUM [46] to detect abnormal behaviors at a detection node. Let X_n represent one sequence of traffic metrics monitored during the time interval Δ_n. The main idea underlying CUSUM is that, during an attack, for the random sequence X_n, there is a step change in the mean value $E(X_n)$. Let *conf* denote the confidence with which the individual detection node suspects an attack toward the destination *dest*. The cooperative detection framework will generate a $(conf, dest)$ pair and forward it to its neighboring detection nodes in the overlay network using the gossip based communication mechanism. Each detection node then independently aggregates its local detection results with attack alerts received from other detection nodes to make a decision. The attack information aggregation process is discussed in Sect. 4.2. Finally, the traffic (A_i) that is identified as an attack based on the consolidated confidence signatures, will be rate-limited according to the formula below:

$$rate_{out}(A_i) = rate_{in}(A_i) * \lambda(conf_i)$$

where $\lambda(conf_i) < 1$ is a factor defined by the confidence level of the attack signature identified.

5.1 Simulation results: DDoS case study

Experiments using a simulation on the Emulab testbed [2] are used to evaluate the accuracy, effectiveness and overheads of the cooperative defense framework against DDoS attacks. Since it is not easy to distinguish DDoS attack traffic from legitimate traffic, the rate limiting mechanism used by the framework will also block legitimate traffic. The following measurements are used to evaluate the performance of the framework:

- Legitimate traffic drop rate and the malicious traffic drop rate under different patterns of DDoS attacks. Since the algorithm used by the framework dynamically adjusts rate limiting to the suspicious traffic based on the aggregated information, a legitimate user should be adequately served.
- Affect of deployment ratio (the ratio of overlay defense nodes to all the network routers) of defense nodes on the effectiveness of the cooperative defense framework. One advantage of cooperative defense mechanism is that even with partial converge or deployment a synergistic defense effect can be achieved. Since not every router or gateway in the Internet will be a defense node, the cooperative defense mechanism is designed to be effective in partial deployment. This feature is supported by an overlay network topology in which only nodes that have established direct peering relationship are aware of each other. The system provides a significant level of defense for potential targets with only a few defense nodes deployed, and becomes more effective as more defense nodes are added, protecting a larger community.
- Reliability and scalability of the underlying gossip mechanism by analyzing the message dissemination convergence rates and the overhead introduced by the cooperation mechanism.

5.2 Simulation results

The decentralized cooperative defense framework is implemented within Linux routers and tested with live traffic in the Emulab testbed. The Snort [35] intrusion detection system is used to detect attacks at each individual detection node. A simple simulated HTTP client-server application is used in the experiments. The GT-ITM [1] topology generator is used to generate the Internet topology used in the experiments. A sample topology with 100 nodes is shown in Fig. 7.

The DDoS attack is simulated using a specified number of compromised nodes in different sub networks. Detection agents are deployed at selected nodes. In the experiments, 10 attackers are used, each of which sends out 1.3 Mbps UDP traffic to the victim. The legitimate user makes requests with traffic rates chosen randomly and uniformly in the range [2 Kbps, 6 Kbps]. If a request successfully arrived at the server, the server returns the requested document after a random processing delay, chosen according to collected empirical distributions [23].

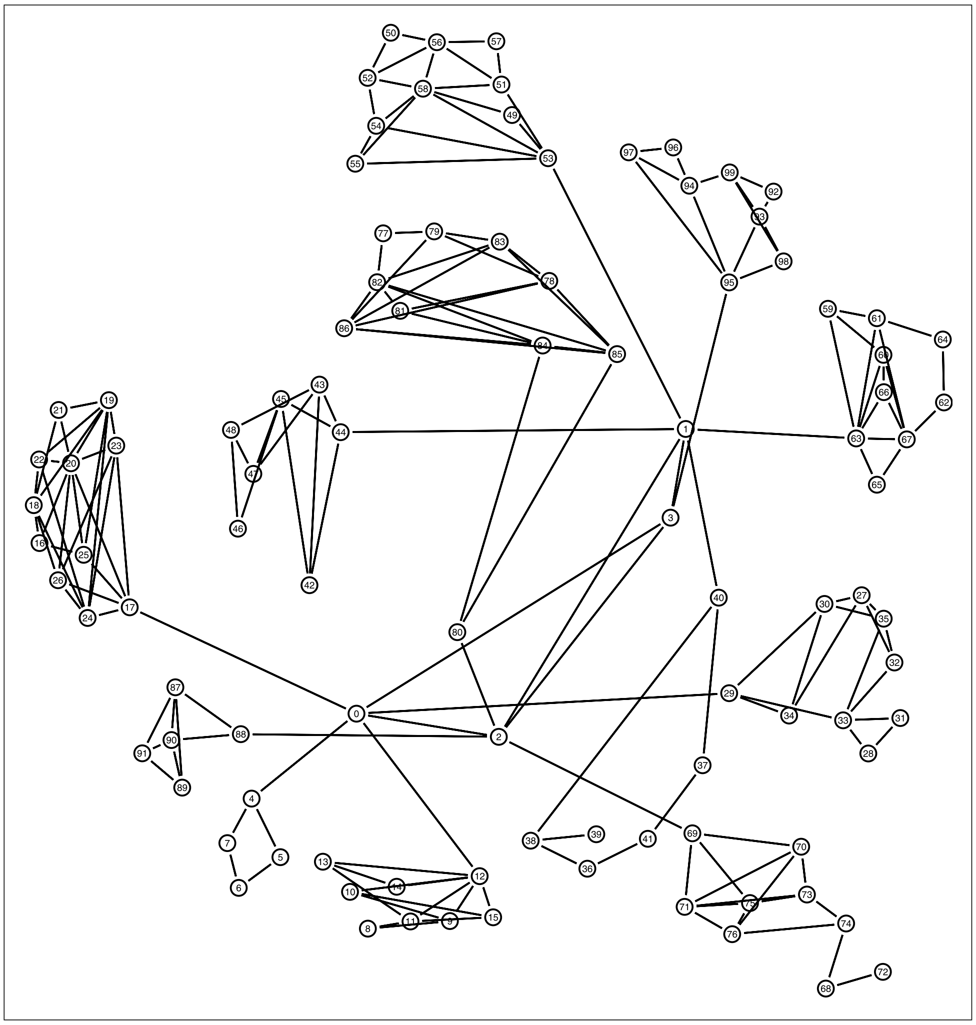


Fig. 7 Network topology simulated in the experiments

In the first set of experiments, "normal use", "under attack without response", and "under attack with cooperative response" scenarios are tested. In each case, the packet rate of a selected client at the HTTP server is measured. Figure 8 plots the result of the experiment. The X axis represents time intervals in seconds; the Y axis represents the number of packets received at the server. The attack starts 50 seconds after the start of legitimate traffic and last for 500 seconds. Compared with the packet rate of normal run, the selected legitimate client's packet rate at the server drop dramatically under attack without response. For the experiment with the cooperative defense mechanism enabled, a gradual increase in the legitimate packet rate is seen. This ramp-up behavior is due to the false detection at local defense node. As a result, some legitimate traffic will be dropped by the rate limiting mechanism as well. As the algorithm converges, each defense node gets more precise information about the global attack information and thus can more accurately rate-limit attack traffic.

The second set of experiments investigate the benefit of an increased deployment of defense nodes. Figure 9 plots the average number of false positive rates, which decrease gradually as more nodes join the peer to peer defense overlay network. By adding sufficient nodes to the defense overlay, attack traffic is dropped efficiently and the amount of the attack packets that reach the victim server decreases. The decrease in the legitimate packet drop rate stabilizes when the deployment ratio is great then 20% in this experiment. This is because, as more detection nodes are added into the cooperative defense overlay, more of them will be on the path from the attack traffic source to the victim destination. As a result, sharing attack information among these nodes will not increase the overall knowledge about the network attack very much.

In the third set of experiments, the parameters of the gossip mechanism are varied to investigate the relationship between the overhead of information sharing and defense efficiency. Let p represent the probability that each detection node in the detection overlay network sends the local attack information to its neighbor nodes. The directional gossip probability p is varied between 0.2, 0.4, 0.6, 0.8, 1.0. The performance of the approach for different gossip probabili-

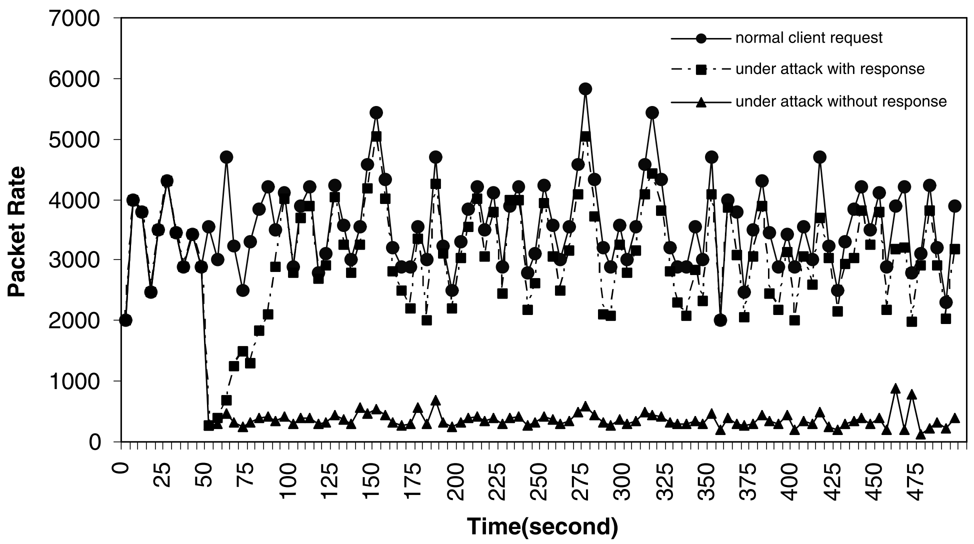


Fig. 8 Packet rates for legitimate traffic under different test conditions

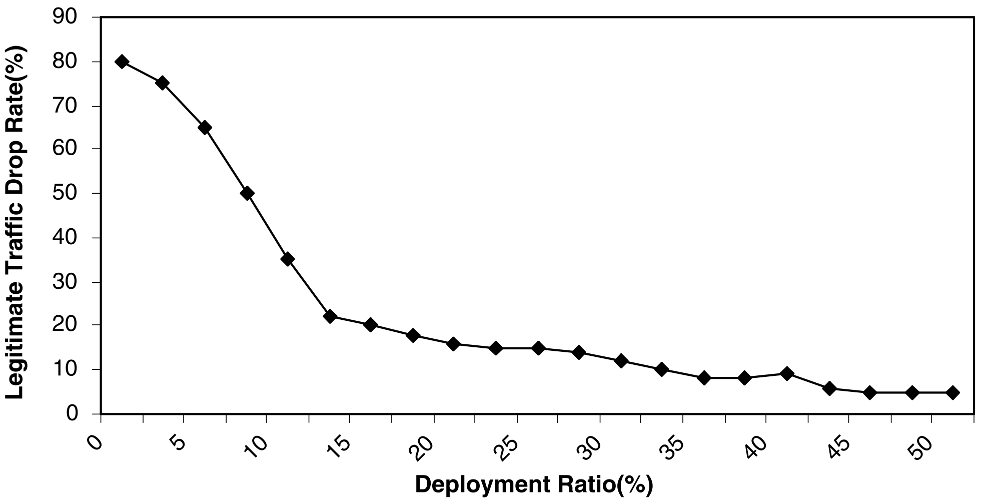


Fig. 9 Reduction in false detection rates with increased deployment

Table 3 Cooperative defense performance for different directional gossip probabilities

Gossip Prob.	False Positive	False Negative
0.2	12.12%	5.2%
0.4	10.03%	4.13%
0.6	8.32%	4.32%
0.8	8.15%	3.56%
1.0	7.67%	3.12%

ties p used are listed in Table 3. The *false positive rate* measures the percentage of legitimate packets dropped by the rate limiting mechanism, and the *false negative rate* measures the percentage of attack traffic that passes the defense node.

As we can see from the simulation results, the framework can detect and defend against DDoS attacks with high accuracy. With $p = 0.4$, a low false positive and low false negative packet drop rate is achieved. The false positive rate is relatively higher than the false negative rate. This is because a high initial drop rate is used when the local defense node detects an attack, and as a result legitimate packets will be dropped in the case of a false detection.

Defenses mitigate the impact of the attack traffic at the victim network but may impose an additional overhead on the networks that implements them. The overhead introduced by distributed cooperative information sharing are also measured in this experiment. Figure 10 plots the per-node overhead with different number of nodes in the system. The number of bytes per second processed by each node for cooperative defense purposes do not increase significantly as more nodes are added to the defense overlay. As a result, the gossip based information sharing mechanism is scalable and can be used in larger networks. When the gossip probability increases, the overhead will increases as well. This parameter can be tuned to adapt to different applications to achieve the best performance.

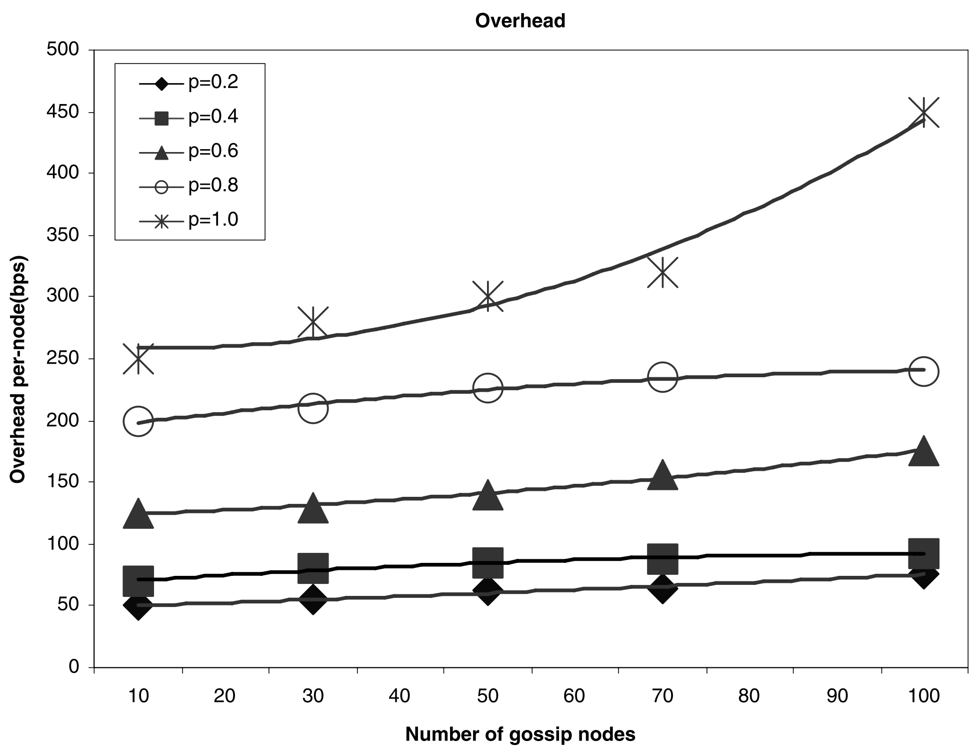


Fig. 10 Overhead of information sharing using gossip

6 Cooperative Internet worm containment

Internet worms are another critical threat to the Internet. They generally use various scanning methods to probe the vulnerabilities of Internet services and propagate themselves rapidly. Unfortunately, many existing defenses which employ local detection to detect infection, hold out little hope in containing novel worms. These mechanisms will have high false alarm rate. A false alarm, whether malicious or unintended, can trigger an effective denial of service attack by the response mechanism itself. In this section, we present a decentralized solution based on the cooperative framework presented in this paper, in which local detection nodes (e.g., firewalls) in various access networks exchange information to collectively defend against worm attacks.

Various worm signature generation mechanisms can be used at the local detection nodes within the cooperative worm containment framework. In this paper, we discuss an implementation based on the Rabin footprint algorithm in Autograph [21]. Each detection node uses this algorithm to compute the content blocks in packet payloads. It then updates the local prevalence $L(i, j)$ for each content block j, where i is the index of a node in the overlay. Once $L(i, j)$ is greater than a local prevalence threshold and its address dispersion is greater than a local address dispersion threshold, the overlay defense nodes begin to filter or rate limit the traffic with this signature.

Local detection nodes use the gossip based aggregation protocol to aggregate the prevalence information within the cooperative defense framework. At any time t, each node i maintains a list of $(content_k, count_k)$ pairs, where $content_k$ represents the signature identified by the signature generation mechanism and $count_k$ represents the prevalence of the signature. Here the k is the index for the content blocks. The gossip protocol used is as follows:

(1) Let $(content_{t-1,k}, count_{t-1,k})$ be all pairs sent to node i in round $t-1$.
(2) Let $d_{t,i} = \Sigma_r count_{t-1,k}$ represent the sum of the prevalence values of the signature $content_k$ received by node i at round t for one particular content block k.
(3) Compare $d_{t,i}$ with $Threshold_i$. If $d_{t,i} > Threshold_i$, then $content_k$ is identified as a worm signature.
(4) Randomly and uniformly choose target $target_t(i)$ from the neighbors of i.
(5) Send the pair $(content_k, \frac{1}{2}d_{t,i})$ to $target_t(i)$ and i (itself).

6.1 Simulation results: Internet worm case study

As a first step, the effectiveness of the cooperative defense framework is evaluated using a Code-Red style worm. While future worms are likely to be more severe, we argue that any containment system must at least mitigate a worm of this magnitude. A simulation, similar to that used by Zou et al. [53], is used to demonstrate the ability of the cooperative defense framework to defend against Internet worms. The Internet topology used is simplified by considering it as a flat network. As discussed in [26], more than 359,000 Code Red infected hosts were observed on July

19th, 2001 by CAIDA. Therefore, 360,000 vulnerable hosts and 1500 cooperating detection nodes connected by a gossip based overlay network are used in the simulations. When the prevalence of the worm signature received by the detection nodes exceeds a certain threshold, the worm signature is confirmed and the detection nodes begin to filter the packets with this signature.

Each vulnerable host stays in one of three states at any time: susceptible, infectious, or removed. A host is in the "removed" state when it is immunized, no matter whether it was previously infected or susceptible. At the beginning of simulation, several hosts are initially selected as infected and the others are all susceptible. Each detection node in the overlay monitor local incoming and outgoing traffic at the edge network, and can be in one of two states: monitoring and alerted. In the monitoring states, it counts the number of worm infection probes observed and shares this information with other detection nodes in the overlay to obtain an aggregated view of the overall worm probe behavior. Using the Rabin footprint algorithm [21] and gossip based aggregation algorithm discussed previously, the worm signatures can be computed, given an enough number of worm probes have been analyzed. When the overlay node has acquired the signatures of the worms, it changes its state from "monitoring" to "alerted". The various thresholds are adjustable parameters in the simulation.

6.1.1 Simulation results

Figure 11 plots the infection progress for a simulated worm attack with and without cooperative defense mechanism enabled. The simulation uses the infection progress model of the Code Red worm presented in [53]. In this experiment, 60% of the vulnerable networks (900 networks out of 1500) have a local detection node as part of the cooperative defense overlay. The signature generation threshold is set to 100 probes.

Figure 12 plots the infection progress for the simulated worm attack with cooperative defense mechanism enabled and using different gossip intervals. In this experiment, once again, 60% of the vulnerable networks (900 networks out of 1500) have a local detection node as part of the cooperative defense overlay and the signature generation threshold is set to 100 probes. Gossip intervals of 0.1, 0.2, 1, 10 minutes are used. When the gossip frequency is increased, the number of infected hosts decrease as expected. Increasing the gossip frequency of the worm information sharing also increases the communication overhead.

The selection of optimal parameters to balance communication costs and defense effectiveness is beyond the scope of this paper. However, it can be seen that the containment of worm propagation is acceptable even for a gossip interval of 10 minutes.

Figure 13 plots the propagation of the simulated Code Red worm attack when independent containment nodes and cooperative overlay defense nodes are deployed. In both cases, the signature generation threshold is set to 100 probes, at which point the packets with that signature are filtered. The plots show that when only independent containment nodes are used, the worm propagation behavior is not constrained and the is comparable to the propagation behavior without any containment. By contrast, when the cooperative defense overlay is used to share worm information, worm propagation is largely constrained.

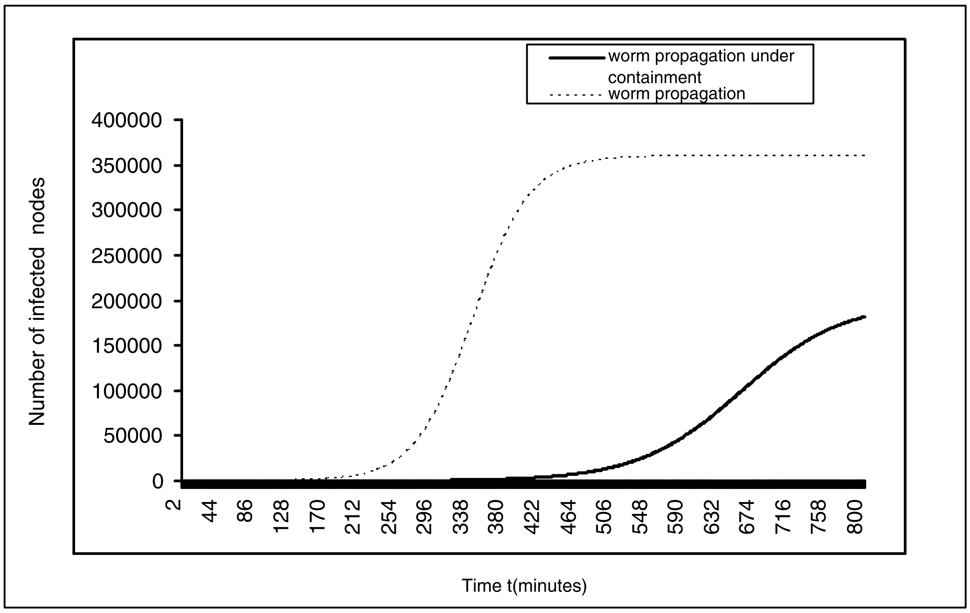


Fig. 11 Propagation of a simulated Code Red worm attack with and without the cooperative defense framework

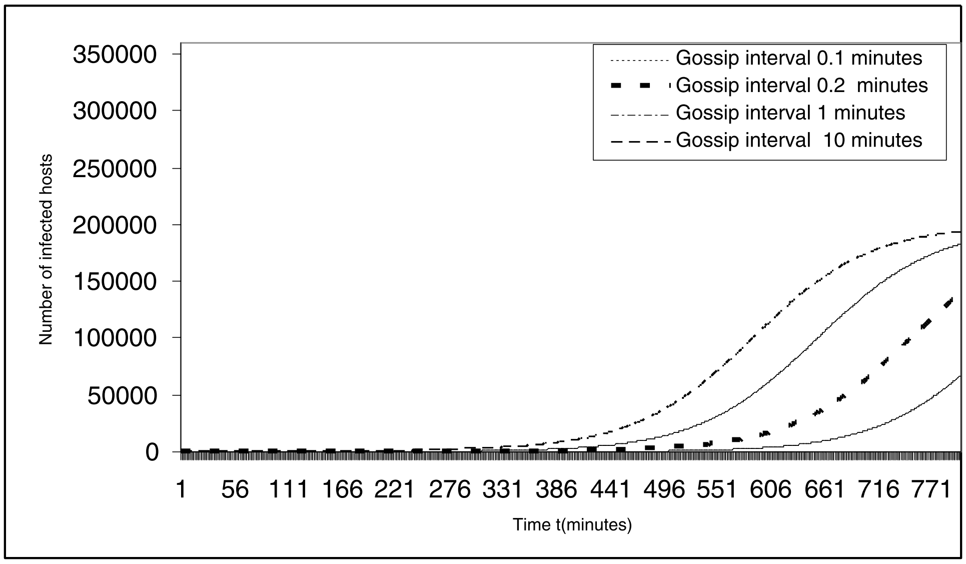


Fig. 12 Effect of gossip interval on the propagation of a simulated Code Red worm attack with cooperative defense enabled

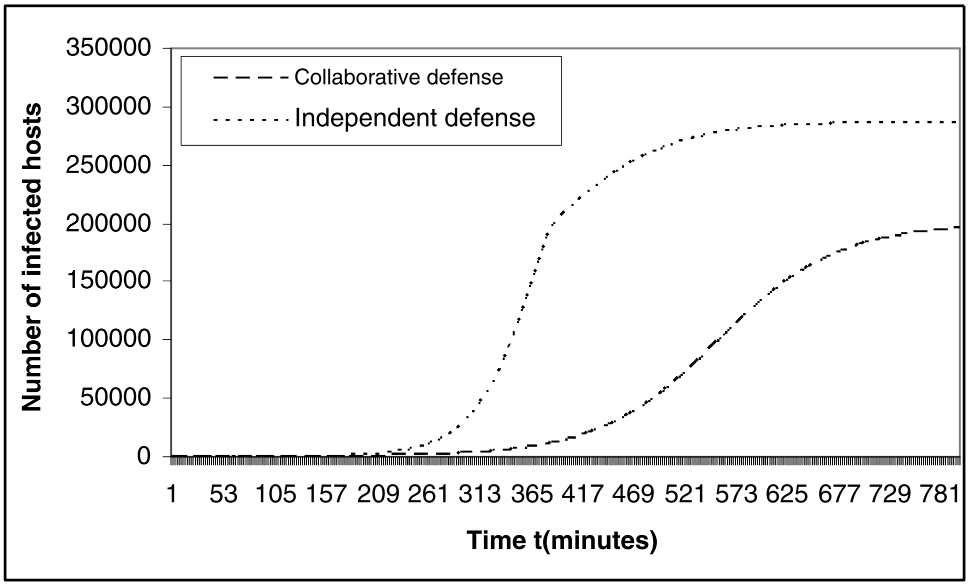


Fig. 13 Propagation of a simulated Code Red worm attack with local defense and with cooperative defense

7 Conclusion

Distributed denial of service and Internet worms are major threats to the global network, which cannot be addressed through isolated actions of sparsely deployed defense nodes. Instead, various defense systems must organize into a framework and inter-operate, exchanging information and services, and acting together, against the threat. In this paper, we designed a decentralized cooperative defense mechanism to protect network infrastructure against network attacks. The framework builds on a self-managing, robust and resilient overlay composed of local detection and defense nodes that are deployed at critical points of the global network infrastructure. These nodes monitor the local network for possible attacks and share attack information using a gossip based communication protocol. Then the aggregated information provides quasi global view of the network attacks, and enables the cooperative framework to more effectively and efficiently defend against network attacks, as long as the length of the attack is greater than the information aggregation latency. This paper also presented a conceptual model that defines the relationships between the level of knowledge in the distributed system and attack detection accuracy. The analysis presented demonstrated the feasibility of gossip based communication mechanisms for cooperative attack detection. A prototype simulation of the framework was used to demonstrate the feasibility of the framework to detect and defend against DDoS attacks and Internet worms, and to evaluate its performance and overheads.

References

1. Modeling topology of internetworks home page. http://www.cc.sgatech.edu/projects/gtitm/
2. Network emulation testbed home. http://www.emulab.net/
3. Gossip-based aggregation in large dynamic networks. (3):219-252, August 2005
4. Adler, M.: Tradeoffs in probabilistic packet marking for IP traceback. In: Proceedings of the Thirty-Fourth Annual ACM Symposium on Theory of Computing, pp. 407–418, Montreal, Quebec, Canada (2002)
5. Andersen, D., Balakrishnan, H., Kaashoek, F., Morris, R.: Resilient overlay networks. In: Proceedings of 18th ACM Symposium on Operating Systems Principles, pp. 131–145, Banff, Canada, October 2001
6. CERT Coordination Center: Cert advisory ca-2001-19 code red worm exploiting buffer overflow in iis indexing service dll (2001). http://www.cert.org/advisories/CA-2001-19.html
7. Chun, B., Lee, J., Weatherspoon, H.: Netbait: a distributed worm detection service (2003)
8. Keromytis, A.D., Misra, V., Rubenstein, D.: Using overlays to improve network security. In: Proceedings of the ITCom Conference, special track on Scalability and Traffic Control in IP Networks, pp. 245–254, Boston, MA, August 2002
9. Dean, D., Franklin, M., Stubblefield, A.: An algebraic approach to IP traceback. Inf. Syst. Secur. **5**(2), 119–137 (2002)
10. Demers, A., Greene, D., Hauser, C., Irish, W., Larson, J.: Epidemic algorithms for replicated database maintenance. In: Proceedings of the Sixth Annual ACM Symposium on Principles of Distributed Computing, pp. 1–12, Vancouver, British Columbia, Canada, August 1987
11. Dittrich, D.: Distributed denial of service (DDoS) attacks/tools (2004). http://staff.washington.edu/dittrich/misc/ddos/
12. DShield.org: Distributed intrusion detection system. November 2000. http://www.dshield.org
13. Eugster, P.T., Guerraoui, R., Kermarrec, A.-M., Massoulieacute, L.: Epidemic information dissemination in distributed systems. IEEE Comput. **37**(50), 60–67 (2004)
14. Ferguson, P., Senie, D.: Network ingress filtering: Defeating denial of service attacks which employIPsource address spoofing (2000)
15. Associated Press for Fox News: Powerful attack cripples Internet (2002)
16. Gil, T.M., Poletto, M.: Multops: a data-structure for bandwidth attack detection. In: Proceedings of 10th Usenix Security Symposium, pp. 23–28, Washington, DC, USA, August 2001
17. Halpern, J.Y., Moses, Y.: Knowledge and common knowledge in a distributed environment. In: Symposium on Principles of Distributed Computing, pp. 50–61 (1984)
18. Hariri, S., Dharmagadda, T., Ramkishore, M., Qu, G., Raghavendra, S.C.: Vulnerability analysis of faults/attacks in network centric systems. In: Proceedings of Parallel and Distributed Computing Systems, pp. 256–261, Reno, Nevada, USA (2003)
19. Ioannidis, J., Bellovin, S.M.: Implementing pushback: Router-based defense against DDoS attacks. In: Proceedings of Network and Distributed System Security Symposium, NDSS '02, pp. 100–108, Reston, VA, USA, February 2002
20. Kempe, D., Dobra, A., Gehrke, J.: Computing aggregate information using gossip. In: Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science, Cambridge, MA, October 2003
21. Kim, H.-A., Karp, B.: Autograph: Toward automated distributed worm signature detection. In: Proceedings of USENIX Security Symposium (2004)
22. Lin, M., Marzullo, K.: Directional gossip: gossip in a wide area network. In: Proceedings of Dependable Computing—Third European Dependable Computing Conference, pp. 364–379, Berlin, Germany (1999)
23. Mah, B.A.: An empirical model of HTTP network traffic. In: Proceedings of the IEEE INFOCOM, pp. 592–600 (1997)
24. Mahajan, R., Bellovin, S., Floyd, S., Ioannidis, J., Paxson, V., Shenker, S.: Aggregate-based congestion control (2003). http://citeseer.nj.nec.com/530614.html
25. Mirkovic, J., Prier, G., Reiher, P.: Attacking DDoS at the source. In: Proceedings of ICNP 2002, pp. 312–321. Paris, France, November 2002
26. Moore, D., Shannon, C., Brown, J.: Code-red: a case study on the spread and victims of an Internet worm. In: Proceedings of the Internet Measurement Workshop (IMW) (2002)
27. Moore, D., Shannon, C., Brown, J.: Code-red: a case study on the spread and victims of an Internet worm. In: ACM/USENIX Internet Measurement Workshop, France, November 2002
28. Moore, D., Voelker, G., Savage, S.: Inferring Internet denial of service activity. In: Proceedings of the USENIX Security Symposium, pp. 9–22, Washington, DC, USA, August 2001
29. Papadopoulos, C., Lindell, R., Mehringer, J., Hussain, A., Govindan, R.: Cossack: Coordinated suppression of simultaneous attacks. In: DARPA Information Survivability Conference and Exposition, vol. 1, pp. 2–13, Washington, DC, April 2003
30. Park, K., Lee, H.: On the effectiveness of probabilistic packet marking for IP traceback under denial of service attack. In: Proceedings of IEEE INFOCOM, pp. 338–347, Anchorage, Alaska, USA (2001)
31. Park, K., Lee, H.: On the effectiveness of route-based packet filtering for distributed DoS attack prevention in power-law internets. In: Proceedings of ACM SIGCOMM, pp. 15–26, San Diego, CA, USA, August 2001
32. Peng, T., Leckie, C., Ramamohanarao, K.: Protection from distributed denial of service attack using history-based IP filtering. In: Proceedings of IEEE International Conference on Communications, vol. 1, pp. 482–486, Anchorage, Alaska, USA, May 2003
33. Pittel, B.: On spreading a rumor. SIAM J. Appl. Math. **47**(1), 213–223 (1987)
34. Porras, P.A., Neumann, P.G.: EMERALD: Event monitoring enabling responses to anomalous live disturbances. In: Proc. 20th NIST-NCSC National Information Systems Security Conference, pp. 353–365 (1997)
35. Roesch, M.: The snort network intrusion detection system (2002). http://www.snort.org
36. Sidiroglou, S., Keromytis, A.D.: Countering network worms through automatic patch generation. In: IEEE Security and Privacy (2005)
37. Singh, S., Estan, C., Varghese, G., Savage, S.: Automated worm fingerprinting. In: Proceedings of the USENIX Symposium on Operating System Design and Implementation, San Francisco, December 2004
38. Snapp, S.R., Brentano, J., Dias, G.V., Goan, T.L., Heberlein, L.T., Ho, C.L., Levitt, K.N., Mukherjee, B., Smaha, S.E., Grance, T., Teal, D.M., Mansur, D.: DIDS (distributed intrusion detection system)—motivation, architecture, and an early prototype. In: Proceedings of the 14th National Computer Security Conference, pp. 167–176, Washington, DC (1991)
39. Snoeren, A.C., Partridge, C., Sanchez, L.A., Jones, C.E., Tchakountio, F., Kent, S.T., Strayer, W.T.: Hash-based IP traceback. In: Proceedings of Sigcomm, pp. 3–14, San Diego, California, United States, August 2001
40. Somayaji, A., Hofmeyr, S., Forrest, S.: Principles of a computer immune system. In: Meeting on New Security Paradigms, Langdale, UK, 23–26 September 1997, pp. 75–82. ACM, New York (1998)

41. Song, D.X., Perrig, A.: Advanced and authenticated marking schemes for IP traceback. In: Proceedings of IEEE Infocomm, vol. 2, pp. 878–886, Anchorage, Alaska, USA (2001)
42. Stone, R.: Centertrack: An IP overlay network for tracking DoS floods. In: Proceedings of the 9th USENIX Security Symposium, pp. 199–212, Denver, CO, August 2000
43. van Renesse, R., Birman, K., Vogels, W.: Astrolabe: A robust and scalable technology for distributed system monitoring, management, and data mining. ACM Trans. Comput. Syst. **21**(2), 164–206 (2003)
44. van Renesse, R., Minsky, Y., Hayden, M.: A gossip-based failure detection service. In: Proceedings of Middleware '98, the IFIP International Conference on Distributed Systems Platforms and Open Distributed Processing, pp. 55–70, England, September 1998
45. Vigna, G., Kemmerer, R.A.: Netstat: A network-based intrusion detection system. J. Comput. Secur. 7(1) (1999)
46. Wang, H., Shin, D.Z.K.G.: Detecting syn flooding attacks. In: Proceedings of IEEE Infocom, pp. 1530–1539
47. Wang, H.J., Guo, C., Simon, D.R., Zugenmaier, A.: Shield: Vulnerability-driven network filters for preventing known vulnerability exploits. In: Proceedings of ACM SIGCOMM, Portland, Oregon, August 2004
48. Weaver, N., Staniford, S., Paxson, V.: Very fast containment of scanning worms. In: Proceedings of the 13th USENIX Security Symposium, San Diego, USA, August 2004
49. Savageand, S., Wetherall, D., Karlin, A., Anderson, T.: Practical network support for IP traceback. In: Proceedings of the ACM SIGCOMM Conference, pp. 295–306, Stockholm, Sweden, August 2000
50. Yau, D.K.Y., Lui, J.C.S., Liang, F.: Defending against distributed denial of service attacks with max-min fair server centric router throttles. In: Proceedings of the Tenth IEEE International Workshop on Quality of Service, pp. 35–44, Miami Beach, FL (2002)
51. Yegneswaran, V., Barford, P., Jha, S.: Global intrusion detection in the DOMINO overlay system. In: The 11th Annual Network and Distributed System Security Symposium (NDSS), February 2004
52. Zhang, G., Parashar, M.: Decentralized information sharing for detection and protection against network attacks (2005). http://www.caip.rutgers.edu/TASSL/Thesis/Guangsen-thesis.pdf
53. Zou, C.C., Gong, W., Towsley, D.: Code red worm propagation modeling and analysis. In: Proceedings of the 9th ACM Conference on Computer and Communications Security, pp. 138–147, November 2002

Guangsen Zhang received his Ph.D. from the Department of Electrical and Computer Engineering at Rutgers University. He received a BE degree in Electronics and Telecommunication from Xi'an Jiaotong University, China and a ME degree in Computer Engineering from Beijing University of Posts and Telecommunication, China. His research interests include autonomic and grid computing, networking and computer security.

Manish Parashar is Professor of Electrical and Computer Engineering at Rutgers University, where he also is Director of the NSF Center for Autonomic Computing and The Applied Software Systems Laboratory (TASSL), and Associate Director of the Rutgers Center for Information Assurance (RUCIA). He also has a visiting position at the eScience Institute at Edinburgh, UK. He received a BE degree from Bombay University, India and MS and Ph.D. degrees from Syracuse University. His research is in the broad area of applied parallel & distributed computing and computational science, and specifically on solving science and engineering problems on very large systems. Manish has received the IBM Faculty Award (2008), Rutgers University Board of Trustees Award for Excellence in Research (2004–2005), the NSF CAREER Award (1999), TICAM, University of Texas at Austin, Distinguished Fellowship (1999–2001), and Enrico Fermi Scholarship, Argonne National Laboratory (1996). He is a senior member of IEEE/IEEE Computer Society and ACM. For more information please visit http://nsfcac.rutgers.edu/people/parashar/.