Interactive Informatics on Internet Infrastructure

F. Zhao, V. R. Vemuri, S. F. Wu
Department of Computer Science
University of California, Davis
{fanzhao, rvemuri, sfwu}@ucdavis.edu

F. Xue, S. J. B. Yoo
Department of Electrical and Computer Engineering
University of California, Davis
{fxue, yoo}@ece.ucdavis.edu

***Abstract*— We present the design and evaluation of I4, a network infrastructure that enables information exchange and collaboration among different domains. I4 can help address problems, such as defending against the unwanted traffic as well as diagnosing the network. We present the Distributed Denial-of-Service (DDoS) attack as an example to demonstrate the advantages of I4. Simulation results show that I4 can significantly reduce the amount of DDoS attack packets and dramatically improve the quality of services received by legitimate users. Our design provides attractive properties, such as incremental deployment as well as incentives for such deployment etc.**

I. Introduction

The current Internet infrastructure follows the end-to-end (*e2e*) principle, which states that functionality should be placed as close to the network edges as possible, keeping the network core focused on the task of routing packets. Thus an ISP domain simply forwards *all* the traffic to its customers[1] with "best efforts" and the customer domains passively receive *all* the traffic arriving at their links.

Indeed, the *e2e* principle simplifies the Internet design and contributes greatly to the success of the Internet witnessed during the last two decades. However, in many scenarios, the domains, either ISPs or customers, would benefit from additional "meta-information" (or "information", in short) related to the current ongoing *e2e* flows. For example, the ISP usually has no knowledge to identify unwanted traffic, such as DDoS attack packets, destined for a different domain; on the contrary, as the actual recipient, the customer domain can identify the offending or unwelcome packets based on its rich capability of intrusion detection or its preferences. If this information, whether certain traffic is wanted or not, is available, the ISP would take actions to eliminate those harmful or useless packets. This not only reduces the traffic load inside the ISP domain, but also prevents the resources in the customer domains from being wasted. Unfortunately the current Internet infrastructure does not provide any means for different domains to exchange such useful information.

This limitation has motivated many proposed solutions[2]. While originally proposed to identify the real path taken by spoofed DDoS attack packets, iTrace/traceback mechanisms [1] [2] [3] [4] [5] enable the ISP domains to propagate the (path) information to victim domains. In ACC/Pushback [11] [12] and Route Throttles [10], the information about the aggregates received and server load is sent back to certain upstream routers where the high-volume aggregates are rate-limited. I3 [19] proposed to decouple the act of sending from the act of receiving; thus the domain needs to insert the binding between its identity and location into the network. SIFF [8] and TVA [17] proposed an end-host capability control mechanism to limit the DDoS attack: the capability is generated by the ISP domains, piggybacked to the recipient and further forwarded to the sender as an explicit authorization. Another example is active network [18] where executable code is sent to the routers in the "black-boxed" Internet core.

All of these pioneering works indicate the need to improve the Internet infrastructure, and each of them has its own merit. Yet we argue that none of them has given a complete tool box for tackling all the problems in today's Internet; moreover, there is still significant space for improvements even with the latest proposals.

In this paper, we propose Internet Information Interaction Infrastructure (I4) to enable the information exchange among domains. It leverages on the most common form of communication, unicast. Our contributions are multifold: First, we present a generic inter-domain information exchange framework within which a large range of challenging problems, such as DDoS attacks, worm, and network failure etc, can be tackled. Second, we present the detailed design of this infrastructure, and demonstrate the effectiveness of information exchange to resist the DDoS attack with extensive experiments and simulations. Third, we describe many accompanying algorithms and ideas, such as weight-based resource allocation and scheduling, and BGP-based key distribution mechanism, as possible improvements over previous proposals.

We design I4 to be practical in the following key aspects. First, the design of I4 strongly incents both ISP domains and customer domains to deploy. Second, I4 supports the incremental deployment that allows the participating domains to have the immediate benefits. Third, we examine many related factors to make the information exchange procedure efficient, robust and secure.

This paper is organized as follows. Section II describes the architecture of I4. Section III describes the detailed information interaction procedure in the case of DDoS attacks. Section IV shows that the useful knowledge can be extracted from information exchanged. Section V studies several procedures to throttle the DDoS attack traffic by applying the extracted knowledge and presents the NS-2 simulation results. In the following sections we summarize the advantages of I4, discuss related issues and present a survey of related works.

[1]Generally speaking, given the path taken by a traffic flow, denoted by $S \rightarrow D_0 \rightarrow \ldots \rightarrow D_n \rightarrow D$, S and D are the customer domains, and $D_0, \ldots, D_n$ are the ISP domains.

[2]Please refer to section VIII for a complete survey on the related works.

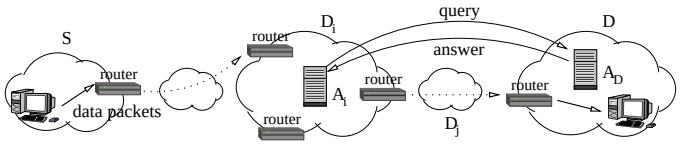


Fig. 1. The I4 agent in D_i, denoted by A_i, observes some packets in the *e2e* communication between S and D. A_i generates a query and sends to the I4 agent, A_D, in D. When this query arrives at D, A_D generates an answer with the help of local Intrusion Detection System (IDS) and other knowledge, and returns back to A_i. Finally A_i will make decisions based on the content of the answer and its local policy.

II. The Architecture of I4

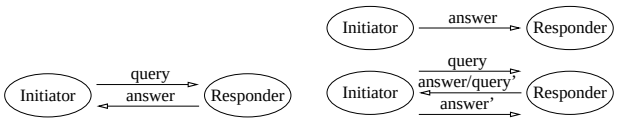
A. Overview

Conceptually, each Autonomous System[3] (AS) in the Internet supporting I4 has one I4 agent responsible for the task of information interaction in this domain (see Fig. 1). Generally speaking, when one agent in the Internet observes a piece of original "information", e.g. one or a sequence of packets, it generates a query regarding this piece of information, and sends this query, usually together with some additional information, to another agent that is responsible for interpreting the query and providing an answer. Finally the agent that sent the query will take actions based on the content of the answer received. This kind of information exchange is termed the "pull" mode. (From the perspective of the initiator, the last event in the information exchange procedure[4] is to pull the information from the responder.)

One of the necessary conditions for information exchange is the availability of the responder's identity or location. However, due to asymmetric routing, the customer domain usually has no idea about which ISP domain currently forwards the traffic coming toward itself[5]. "Pull" mode may be the most suitable way for information exchange in this situation: the ISP domain attaches its location/identity information in the query so that the customer domain knows where to send the answer back.

In other scenarios, the initiator may already know the identity/location of the responder, e.g. via previous information exchange or a service agreement established through some additional channel. If it is the case, the query is not a necessary condition for an answer to be triggered. This kind of information exchange is referred as "push" mode because the initiator pushes the information to the responder. Fig. 2 shows the procedures of both modes. Note that a more complicated "push" mode may require some preliminary communication to fulfill certain prerequisites for the final "push". Fig. 2(b) shows such an example.

To exchange information, query and answer are either piggybacked in the originally observed data packet or conveyed by an out-of-band message. Different choices result in different tradeoffs between overhead and flexibility. Either way, query and answer are usually carried by unicast IP packets that are routed by Internet standard routing protocol, i.e. BGP.



(a) Pull mode (b) Push mode

Fig. 2. Different information exchange modes

B. Intra-AS issues

Due to scalability consideration, the function of the I4 agent may be implemented in as few as just one node inside one AS. In order to exert its capability, the I4 agent has to collaborate with other entities, such as IDS/IPS and routers. The necessary network and intra-domain routing configurations must be set up in advance in order for them to communicate with each other. Moreover, the security association between I4 agent and other entities is established and the time-synchronization is maintained. These requirements are reasonable because they are under the same administration domain.

A router plays an important role in the information exchange. First, it selects some data packets based on certain criteria and forwards them to the I4 agent so that a query can be generated. Second, query and answer are forwarded by routers to the corresponding I4 agents based on their Forwarding Information Bases (FIBs). In addition, during the transmission a router in the I4 domain checks whether an incoming query or answer should be processed in this domain; if so, the router forwards it to the local I4 agent. Third, after an I4 agent receives the answer and decides the actions to take, it communicates with routers where the decisions will take effect.

It is also possible, especially in one large AS, that there are multiple I4 agents set up for the purpose of fault-tolerance, load-balancing, etc. In order to achieve scalability when enabling the inter-agent communication, these I4 agents can be organized in the same way as route-reflection and consolidation in BGP. The details of such kinds of inter-agent communication protocol and hierarchy organization are beyond the scope of this paper.

C. Security

To secure the inter-domain information exchange, an established security association (SA) is usually required. However, this requirement limits the scenarios where the information exchange is feasible because 1) there is no global key management infrastructure, such as PKI, currently; 2) the information exchange is needed even among those previously unacquainted domains. In this subsection, we present some practical security solutions that provide a "weaker" but reasonable security under various information exchange modes.

[3]In this paper, "AS" is used exchangeably with "domain".

[4]Precisely speaking, all information exchange events are correlated more or less in the long term, and all of them constitute one large procedure that lasts through the lifetime of one agent. In this paper, we abstract a set of temporally and content correlated events as one information exchange procedure.

[5]Instead, the ISP domain can implicitly infer how to reach the agent in the customer domain from the destination IP address in the packets forwarded.

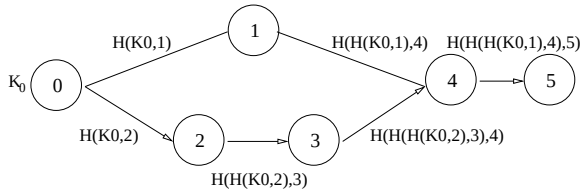


Fig. 3. Multipath based authentication

1) BGP based key distribution mechanism: Fig 3 illustrates our idea using a simple topology where there are six ASes and each number in the circle represents the AS number. Each AS, i, generates a secret key, K_i. When AS *0* announces its network prefixes Pf_0 to its neighbor, e.g. *1*, it includes a key (called "received" key), $Kr^0_{0,1} = H(K_0, 1)$ in the BGP message where H is a secure hash function. Note that $Kr^(0)_{0,1}$ can be securely transferred because the link between 0 and 1 is either a direct physical link or a logical link built upon a pre-established peering relationship. AS 1 stores this received key together with Pf_0 and the corresponding AS path in its routing information table. When AS 1 propagates Pf_0 to its neighbor, e.g. AS 4, it generates another received key, $Kr^0_{1,4} = H(Kr^0_{0,1}, 4) = H(H(K_0, 1), 4)$, for AS 4. The same procedure is repeatedly applied by any intermediate AS when propagating Pf_0. Note that each AS would use its own secret key when announcing its own network prefixes, and use the received key when propagating the received network prefixes. When one AS receives multiple BGP messages regarding Pf_0, it selects one AS path as the best one and keeps the rest as the backup paths. In the following we describe how this mechanism helps the security in various information exchange modes.

2) Security improvement in one-way "push" mode: In this mode, assume the initiator is 4 and the responder is 0. As AS 4 takes the shortest path, $4 \rightarrow 1 \rightarrow 0$, as the best path, it must receive a key $Kr^0_{1,4} = H(H(K_0, 1), 4)$ associated with Pf_0. In its answer to 0, 4 includes the current best AS path and Message Authentication Code (MAC) generated as follows: $MAC = H(Kr^0_{1,4}, \textit{the content of answer} \parallel 4 \rightarrow 1 \rightarrow 0)$. When AS 0 receives this answer, it generates $Kr^0_{1,4}\prime$ from the received AS path and K_0, then reconstructs $MAC\prime$ and compares it with the received one. AS 0 accepts the received answer if they match and otherwise drops it silently.

In this way the responder can assure that the initiator is indeed on the "AS path" provided in the answer. An attacker may try to forge the "path" in order to avoid being identified during the information exchange. However, it cannot derive the correct keys used by other ASes (except the downstream ones) due to the one-way property of hash function. Although the attacker can forge a long "path" seemingly from its downstream domains, it still has to include itself in this "path". Thus this improves the security of one-way "push" mode; otherwise the attacker could send the forged answer from anywhere without being detected.

3) Multi-path based authentication: In terms of security, "pull" mode and three-way "push" mode are better than the one-way "push" mode because the additional exchanges implicitly ensure that the answer is from the one who actually receives the query before. However both are still vulnerable to Man-in-Middle attacks. To address this issue, our solution leverages on the availability of multiple AS paths.

As shown in Fig. 3, AS 4 receives two AS-paths to AS 0 and two received keys, $Kr^0_{1,4}$ and $Kr^0_{3,4}$. Assume that AS 4 is the initiator and AS 0 is the responder. In the "pull" mode, AS 4 generates a query:$\langle 4 \rightarrow 1 \rightarrow 0 \parallel 4 \rightarrow 3 \rightarrow 2 \rightarrow 0 \parallel MAC \parallel \textit{the content of query} \parallel nonce \rangle$ where $MAC = H(Kr^0_{1,4}, Kr^0_{3,4}, 4 \rightarrow 1 \rightarrow 0 \parallel 4 \rightarrow 3 \rightarrow 2 \rightarrow 0 \parallel \textit{the content of query} \parallel nonce)$. usually this query will follow the best path back to AS 0. When AS 0 receives this query, it generates the corresponding received keys and verifies *MAC*. If everything is OK, AS 0 generates an answer as follows:$\langle \textit{the content of answer} MAC \parallel nonce + 1 \rangle$ where $MAC = H(Kr^0_{1,4}, Kr^0_{3,4}, \textit{the content of answer} \parallel nonce + 1)$. When AS 4 receives this answer, it will verify *MAC*. In the three-way "push" mode, AS 4 and AS 0 can even establish a pre-shared key to further protect the confidentiality of information exchanged. The details are skip here.

The availability of multiple AS paths enables AS 4 and AS 0 to prevent the Man-in-Middle attack because the attacker must be able to control both AS paths to succeed. Multiple AS paths are not only available to a multi-homed AS. For example, a single-homed AS, such as AS 5, may receive multiple different AS paths to AS 0 during routing dynamics. AS 5 can verify whether there is any "man" in the middle between AS 0 and AS 4; however it cannot detect whether 4 is such an attacker.

4) Discussion and summary: To manage the keys, the origin AS can indicate the length of the validity period in seconds in the BGP messages, which does not require global time synchronization.

The ways we distribute the "recevied" key is similar with Listen&Whisper [21]. However, it is for different purposes. Listen&Whisper focuses on verifying the integrity of different AS paths while we try to utilize the availability of AS paths for security protection of inter-domain communication. In fact, our scheme can be combined with Listen&Whisper to provide better security.

III. Information Interaction in the case of DDoS Attacks

A. Overview

DDoS attacks are deemed as the first-order threat in the Internet. The infamous attack in February 2000 caused major Internet portals such as Yahoo, eBay and E*Trade to shut down. Despite the lack of media attention after that, DDoS attacks are even more severe and prevalent in the Internet. Today the binary codes or even the complete packages of DDoS attack tools are readily available and do not require sophisticated knowledge to launch. In a previous paper [14], the authors reported a surprisingly huge number of DDoS attacks observed in everyday traffic.

We apply the "pull" information exchange mode in the case of DDoS attack. We call the agent in the ISP domain "query

agent" and the agent in the customer domain (i.e. the target of attackers) "answer agent". As shown in Fig. 1, the information exchange procedure can be formulated as a feedback model: the query is a signal to the customer domain while the answer serves as a feedback to the ISP domain; after several times of exchanges, this feedback mechanism would make the whole system converge to an equilibrium state. In the following we present detailed packet formats and information interaction procedures.

B. Query

1) Query generation: To generate a query, routers in the ISP domain randomly select a data packet with a certain probability, Pr, and forwards this selected packet[6] to the query agent. We propose to piggyback the query in the selected data packet because an out-of-band query message results in more overheads. Although this may cause the fragmentation if Maximum Transmission Unit (MTU) is exceeded, we argue that the adverse impacts are little because: 1) the attacker tends to use smaller packets in order to exhaust the router resources to the maximum level; 2) the attack packet itself is small in many challenging DDoS attacks, such as SYN flooding and setup channel flooding [17]; 3) the query is piggybacked in the first fragmentation; thus once an attack fragmentation is identified, the whole packet does not have to be re-assembled.

We propose a new type of IP protocol, called "query", which is placed in the protocol field in the original IP header. The query starts with a generic header where the next header field indicates the type of next header, either an upper transport protocol or another query. Each payload follows the Type-Length-Value (TLV) format as some payloads are optional or of variable size.

The value of Pr should be carefully chosen in order to strike a balance between the overhead of IP packet processing and the amount of information exchanged. Furthermore, a router could select the data packet from different aggregates[7] [11] with different probabilities, thus it can spend more resources for some aggregates of special interests, e.g. those destined for one preferred customer domain whose intention of reception is distributed proactively or reactively [5].

2) Query payloads: Table I lists the descriptions and the suggested lengths of payloads appearing in the query. Note that in practice, there may be more efficient way to represent these payloads. We briefly discuss below the use of these payloads. (See section IV for more details.)

- "Router ID" and "Interface ID": The administrator can assign unique numbers to routers and their interfaces. These two payloads identify the origin of the information (either a query or answer) received by an I4 agent. Also later a query agent can know where the corresponding knowledge should be distributed.

[6]Additional information, such as the IP address or the identity of the router, the interface where this selected data packet arrives or departs, and the local time, may be forwarded to the query agent as well.

[7]The traffic can be separated into aggregates based on the interfaces that the packets arrive at or depart from, destination IP address or prefix, or the output of some clustering algorithm.

TABLE I

PAYLOADS IN THE QUERY

Payload Type	Length	Description
Query agent	4 bytes	the query agent's IP address at which the answer will be received
Router ID	2 bytes	to identify the router that selects this data packet
Interface ID	1 bytes	to identify the interface where the selected data packet arrives or departs
Timestamp	4 bytes	the local time when the router selects this data packet
Sequence No.	4 bytes	a counter maintained by the query agent
Cookie	16 bytes	a random number generated for stateless verification

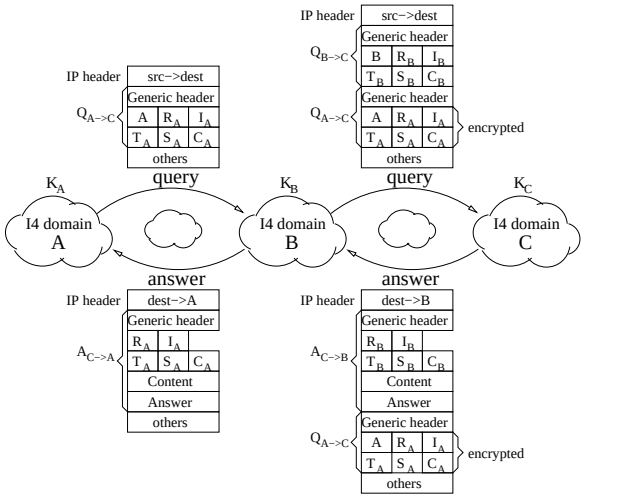


Fig. 4. Information exchange procedure: R_A, I_A, T_A, S_A and C_A denote "Router ID", "Interface ID", "Timestamp", "Sequence number" and "Cookie" generated by A respectively. The payloads generated by B are denoted in the same way. Note that C copies R_B, I_B, T_B, S_B and C_B from the received query $Q_{B\to C}$ into the answer message $A_{C\to B}$.

- "Timestamp": It allows a query agent to learn the temporal property of the information received. This payload together with "Router id" and "Interface id" is provided by routers to the query agent.
- "Sequence number": To resist the replay attack, this payload contains the current value of a counter incremented by one when a query is sent.
- "Cookie": It allows the query agent to statelessly verify that the received answer is indeed in response to a query generated by itself earlier. This payload contains the output of the following formula:

$$H(K, \textit{Query agent} \parallel destIP \parallel \textit{other payloads}) \quad (1)$$

where K is the secret key generated by the query agent and H is a secure hash function.

In the case of a DDoS attack, the query is like a question to the answer agent: Is this selected data packet good or bad?

3) Query transmission: As the destination IP address is not changed, the query is forwarded to the same destination domain as the original data packet selected. Fig. 4 shows the procedure of query transmission. A generates a query, $Q_{A\to C}$, to C. If there is an I4 domain, say, B on the route from A to C, it treats this received query just like a *selected* data packet: B

TABLE II

ADDITIONAL PAYLOADS IN THE ANSWER MESSAGE

Payload Type	Length	Description
Content	variable	IP header and the partial upper layer payloads in the selected data packet
Answer	1 byte	the evaluation result of the selected data packet
Signature	variable	the signature of good or bad traffic
Duration	4 bytes	the validity period of the supplied signature

TABLE III

POSSIBLE VALUES AND MEANINGS OF THE ANSWER PAYLOAD

Value	Description about the selected data packet
$(00)_{16} \leq y \leq (64)_{16}$	the prob. as a bad packet is $y\%$
$(32)_{16}$	unknown, 50% as a bad packet
$(00)_{16}$	a good packet, 0% as a bad packet
$(64)_{16}$	a bad packet, 100% as a bad packet

first encrypts $Q_{A\to C}$ (except the generic header) with its secret key, K_B, and then inserts its own query $Q_{B\to C}$ in between the IP header and $Q_{A\to C}$. Note that the cookie payload in $Q_{B\to C}$ also covers the encrypted portion of $Q_{A\to C}$. Finally this query will arrive at the destination domain C.

C. Answer

1) Answer message generation: As the actual recipient, the customer domain is the most appropriate one to answer whether this selected data packet is good or bad. In addition, with the help of local IDS/IPS, it is indeed capable to provide an accurate answer[8].

We use an out-of-band message to carry the answer because some *e2e* communication is unidirectional. Similarly, we design a new type of IP protocol, called "answer"[9]. In the answer message, the source IP address is the destination IP address in the received query, and the destination IP address is the value of "Query agent" payload, i.e. the IP address of the query agent. The answer agent has various strategies to respond to queries; for example, it may cluster the answers to a set of queries in one answer message in order to reduce the overhead.

2) Answer message payloads: All the payloads (except the first "Query agent" payload in cleartext, but including the following encrypted portions) in the received query should be copied into the generated answer message. In addition, as shown in Table II the following four new payloads may appear in the answer message.

- "Content": This allows the query agent to correlate the received answer with the originally selected data packet.
- "Answer": This contains the evaluation result, namely, the probability that the selected data packet is bad. Table III shows the possible values in this payload. The probability as a good packet can be calculated easily.
- "Signature": When combining with "Answer" payload, this optional payload indicates the signature of the good or bad traffic represented by the header and/or the partial data payload, thus the query agent could install some filters based on the received signature in the appropriate routers.
- "Duration": This indicates the validity period of a provided signature. Note that the query agent may independently set up the lifetime for the received signature rather than based on this payload.

3) Answer message transmission: Fig. 4 shows the procedure of answer message transmission. When B receives an answer message from C, it first checks if this is a replayed message by examining the "Sequence number" payload, S_B, just like the anti-replay window mechanism in IPSec. Then B reconstructs the cookie based on Equation 1 with IP addresses and related payloads as inputs. Note that the order of the source IP address and the destination IP address in the received answer message should be reversed when calculating the cookie. B accepts this answer message if the output matches with the "Cookie" payload, C_B, received or simply discards it otherwise. After the validation, B may update its knowledge and take further actions based on the received information. Furthermore, B decrypts the first encrypted query, $Q_{A\to C}$ in this example and constructs an answer message for A based on, e.g. the "Content" payload and the "answer" payload, generated by C. When A receives $A_{C\to A}$ from B, it follows the same procedure to verify the received answer message and takes the information into consideration if succeed.

D. Discussion

Our information exchange protocol is efficient and lightweight because it does not maintain the connection-oriented states like in TCP. "Cookie" payload enables the query agent to statelessly verify the received answer message. It is computationally impossible for an attacker to forge a valid cookie without the knowledge of the secret key, K. Although it is still vulnerable to Man-in-Middle attack, it does not introduce any new threat. See section VII for more discussion on security issues.

Query and answer may be lost or reordered during the transmission. With the anti-replay sliding window and the stateless verification, our protocol is robust against packet reordering. Moreover, "Timestamp" payload allows the query agent to apply the received information properly, especially when an answer experiences the long transmission delay. Finally as we will show in section IV, adaptive sampling can tolerate I4 packet loss.

IV. KNOWLEDGE EXACTION FROM INFORMATION INTERACTION

From the information exchanged, the customer domain and the ISP domain can extract the knowledge about aggregates, as we shall show below. An aggregate can be denoted by $\langle I, R_{id}, I_{id}, C\rangle$ if it is destined for a customer domain C and arrives at an interface I_{id} of one particular router R_{id} in the ISP domain I. Each element in this vector can be either 0 or represent a particular domain, router or interface. We use Fig. 4 to illustrate our method. We use the following

[8]Indeed, the partial path information carried in the query may help identify the attack packet.

[9]The numeric values for the types of "query" and "answer" protocols will be assigned by IANA.

notations: T is the length of the time period during which a percentage is measured by counting the received answer messages; N_A is the number of answer messages regarding this aggregate received during T; N_A^b is the number of answer messages with negative evaluation results regarding this aggregate received during T; Pc is the percentage of bad packets of one aggregate.

A. Knowledge

1) The arrival rate of traffic: Every domain can estimate the arrival rate of aggregates arriving at their local links. We adopt the following formula from [11]:

$$R_{new} = (1 - e^{-\frac{t}{k}})R_{current} + e^{-\frac{t}{k}}R_{old} \quad (2)$$

where $R_{current} = \frac{l}{t}$, t is the inter-packet interval, k is a constant, e.g. $k = 2$ and l is the average length of data packets.

Moreover, with "Query agent", "Router ID" and "Interface ID" payloads in the received query, the customer domain C can estimate the arrival rate of aggregates forwarded by one remote ISP domain, B, because each query is a randomly selected sample of the traffic. For example, if within T seconds the number of queries that C receives from B is $N_{Q,B}$ and the probability of query generation is Pr, the rate of the aggregate $\langle B, 0, 0, C\rangle$ is $\frac{N_{Q,B}}{Pr*T}$ packets per second.

2) The percentage of bad packets: C can further estimate the percentage of bad packets, denoted by Pc, of the aggregates forwarded by B. For example, assume within T seconds the number of queries from B received by C is $N_Q - 1$ and C generates the answers $\{A_0, \ldots, A_{N_Q-1}\}$. Recall that A_i, $0 \leq i \leq N_Q - 1$, contains the probability that a selected data packet is bad. Then Pc of the aggregate $\langle B, 0, 0, C\rangle$ during this time period is estimated as $\frac{\Sigma A_i}{N_Q}$ where $0 \leq i \leq N_Q - 1$. With additional "Router ID" and "Interface ID", C can estimate Pc of even "smaller" aggregates.

Similarly, B can estimate Pc of the aggregate $\langle B, R_{id}, I_{id}, C\rangle$ based on the answer messages received from C. Assume that within T seconds, the answer messages with the router id, R_{id}, and the interface id, I_{id}, received from C are $\{A_0, \ldots, A_{N_A-1}\}$. Then Pc of the aggregate $\langle B, R_{id}, I_{id}, C\rangle$ during this time period is $\frac{N_A^b}{N_A} = \frac{\Sigma A_i}{N_A}$, where $0 \leq i \leq N_A - 1$.

Note that we assume that each data packet in aggregates is discrete when calculating Pc. In fact, if one selected data packet belongs to one session, we can label all the packets in this session based on the answer regarding this selected data packet.

B. More about percentage estimation

A smaller query generation probability can reduce the processing overhead and traffic load, however it results in inaccurate percentage estimation. The previous subsection presents a basic approach to estimate the percentage of bad packets. Fig. 5 gives a simple result of implementing a "sample and hold" strategy which is nothing but a zero-order interpolation. From this figure we can see that when the probability of query generation becomes smaller, the error in estimation becomes bigger. However even when $Pr = 0.005$, the estimation is still close to the real data. We will apply these parameters in the simulation of rate-limiting DDoS attack traffic in section V.

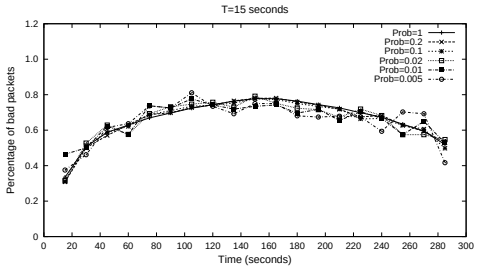


Fig. 5. Percentage of bad packets when T=15

In the future, we propose to study this issue further in depth. For example, two questions that need to be answered are: 1) how to adapt to the traffic dynamics and adjust the corresponding parameters to provide an accurate estimation? 2) how to apply the estimated percentage to the current incoming traffic and promptly adapt to any rapid change?

To address the first question, we intend to apply an adaptive sampling method where the sampling frequency depends on the dynamic properties of the variables being sampled. To address the second question, we can analyze the trend from the last m measured percentages, then estimate Pc in the near future, for example, by Linear Mean Square Estimation (LMSE) and ARMA. It is also useful to evaluate these approaches with real DDoS attack traces from real networks.

V. Examples of Collaborative Defense against DDoS Attacks

I4 is capable to address any kind of unwanted traffic. If a "signature" is available, such as existing TCP sessions, non-spoofing packet flooding and worm traffic, a filter based on the answer message can be installed in the upstream domain. With the knowledge described in section IV, I4 is even more powerful in that it can address more challenging attacks, such as spoofing attack[10] and initial requests flooding [17]. In this section, we assume a general form of the DDoS attack where a "signature" is not available.

Our observation is that during a DDoS attack, bad traffic contends limited resources, such as the packet scheduling, the link bandwidth etc, with the good traffic. However, currently the router cannot distinguish the "good" from the "bad". I4 precisely addresses this limitation. In the following we show how knowledge learned from information exchanged would help mitigate the effects of DDoS attacks.

A. Differentiated server load balancing mechanism

Reference [10] proposed that during the DDoS attack a server (the victim) indicates the load it desires to specific upstream routers, then routers drop the excess traffic to the server. For example, assume that there are n upstream

[10]Due to the lack of IP address accountability, an attacker can easily conceal his/her location(s). Moreover a header or content based filter may cause the bilateral damage.

domains, $\{D_0, D_1, ..., D_{n-1}\}$, forwarding the traffic to one victim domain, V. V splits its total server load, S, into $\{S_0, S_1, ..., S_{n-1}\}$ and then indicates this information to the corresponding upstream domains, D_i. However in [10] V does not split its server load optimally. As we described above, with the information exchange the victim domain can now estimate the volume of traffic forwarded by each upstream domain D_i and which upstream domain forwards the "better" traffic in terms of the percentage of bad packets within. Thus V can assign the larger workloads to those D_i forwarding a lower percentage of bad packets, which makes the victim domain not only receive the appropriate amount of traffic without exceeding its capacity, but also serve more "good" packets from the legitimate users.

B. Weighted queue scheduling mechanism

In this section we propose a weighted queue scheduling mechanism that schedules packet forwarding from one incoming queue to one outgoing queue based on the weight assigned to the incoming queue.

1) Description: Given a router with n queues (Usually each queue has the same characteristics, such as bandwidth and delay.), $\{Q_0, Q_1, ..., Q_{n-1}\}$, let the percentage of bad packets in each queue Q_i be p_i. Each queue Q_i is assigned a weight, $w_i = f(1-p_i)$, where $f()$ is an ascending function or simply $f(x) = x$.

In the classical "Round Robin" scheduling mechanism, each Q_i is scheduled with the same weight and then the percentage of bad packets forwarded by this router is equal to $Y = \frac{\Sigma p_i}{n}$. In this proposed mechanism, the ratio between the number of packets forwarded from Q_i and the total number of packets forwarded by the router is equal to $\frac{w_i}{\Sigma w_i}$ and the percentage of bad packets forwarded by the router is equal to $X = \frac{\Sigma p_i * w_i}{\Sigma w_i}$ where $i = 0, 1, ..., n-1$. It can be easily proven that $X \geq Y$ and $X = Y$ if and only if $p_0 = p_1 = ... = p_k$. Thus the weighted queue scheduling mechanism is better than or as good as the "Round Robin" mechanism in terms of the overall percentage of bad packets forwarded.

By assigning a higher weight to the queue containing a lower percentage of bad packets, the router in the ISP domain spends more resources in forwarding the packets from these "good" queues. Thus the queue with the higher percentage of bad packets tends to become full and eventually more bad packets are dropped.

2) Discussion: The proposed mechanism is based on the preferential scheduling of the shared resources among interfaces. Although the modern "carrier-class" router starts to have more and more parallelism built in, there might still exist many central resources shared among linecards. Moreover there are still a lot of legacy routers, for example, without dedicated CPU or memory for each linecard, or without full-mesh cross-bar. As it is these slower routers that are more likely congested during the DDoS attack, this proposed mechanism could significantly improve the performance of good sessions if implemented in these bottlenecks.

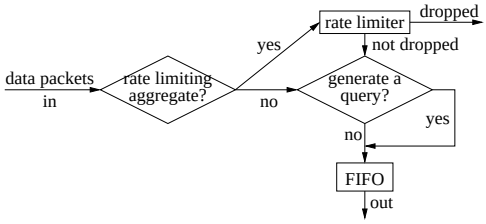


Fig. 6. Rate limiting procedure

C. Weighted aggregate-scheduling mechanism

We propose another scheduling mechanism based on the weight of aggregates inside each queue.

1) Overview: Assume that there are n aggregates in one unidirectional queue, Q, $\{A_0, A_1, ..., A_{n-1}\}$. The arrival rate of A_i is R_i pkt/sec or B_i Mb/sec, thus the total arrival rate of all aggregates is ΣB_i Mb/sec. Also we assume that the probability to generate an I4 query from the packets arriving at Q is P and the bandwidth of Q is B Mb/sec.

When the queue is congested, the router starts to rate-limit the incoming traffic in this queue. The total of excess traffic to be dropped is $\Sigma B_i - c * B$ where c is a constant factor. Fig. 6 shows the procedure of processing an incoming packet in I4 queue during the congestion. The router checks whether this packet belongs to an aggregate to be rate-limited. If yes, the packet is forwarded to a rate-limiter module that determines whether this packet should be dropped. If not, the packet is forwarded to a query generation module that generates a query based on this packet with the probability P.

2) Rate-limiting algorithms: Algorithm 1 shows the pseudo code of greedy rate-limiting algorithm implemented in our simulation.

Algorithm 1 Greedy rate-limiting algorithm

Sort the aggregates in the descending order of the percentage of bad packets, for exmaple, $\{A_{i_0}, A_{i_1}, \ldots, A_{i_n}\}$.
$j \Leftarrow 0$, $E \Leftarrow \Sigma_{k=0}^{n} R_{i_k} - B$ where R_{i_k} is the arrival rate of A_{i_k} and B is the link bandwidth.
Given an incoming packet, *pkt*, for each aggregate, A_{i_j},
  if *pkt* $\in A_{i_j}$
    pkt is dropped with the probability, min$\{E/R_{i_j}, 1\}$, and exit the loop.
  else if $R_{i_j} > E$, then *pkt* is forwarded and exit the loop.
  else $E \Leftarrow E - A_{i_j}$, $j \Leftarrow j + 1$

Besides, other rate-limiting algorithms, such as token-bucket rate-limiting algorithm [11] and weighted rate-limiting algorithm as described in section V-B, are also possible. Compared with ACC/Pushback [11], our proposal has more advantages: 1) the high-bandwidth aggregate may not be attack traffic always; 2) with the information learned from the real recipient, the router has a better way to aggregate the flows together and drops more from bad aggregates.

3) Aggregation: To make the rate-limiting more effective, it is better to consider the aggregates with the similar percentage of bad packets together. This may need to combine or separate aggregates dynamically. Moreover, it may be more cost-effective to consider the aggregation of small aggregates as

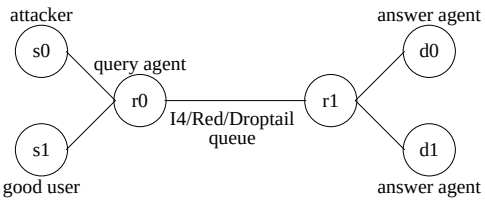


Fig. 7. The simulation topology

TABLE IV

THE SETTING OF BACKGROUND TRAFFIC

src	dst	bad traffic volume (Mbps)	total traffic volume (Mbps)	start (Seconds)	end (Seconds)
s_0	d_0	0.1	0.1	5.0	310.0
s_0	d_1	0.3	0.3	5.0	310.0
s_1	d_0	0	0.3	0	310.0
s_1	d_1	0	0.1	0	310.0

a whole. An alternative is to ignore the small aggregate for now and consider it later at some downstream domains when it has converged to a big enough aggregate. We plan to study more about dynamic aggregation and the impacts of these two different strategies on small aggregates in the future.

4) Simulation: In the topology shown in Fig. 7, s_0 is a DoS attacker and s_1 is a good user. In order to simplify the problem, we assume d_0 and d_1 have a way to identify the attack packets, such as based on the source IP address. We attach a query agent to r_0, and answer agents to d_0 and d_1. The traffic arriving at the queue $\langle r_0, r_1 \rangle$ is separated into aggregates based on the destination IP address, d_0 and d_1. In our NS-2 simulation, the bandwidth of $\langle r_0, r_1 \rangle$ is 0.81Mbps, the probability to generate a query is 0.005 and the time period to estimate the percentage of bad packets is 15 seconds.

Table IV shows the background UDP traffic in the simulation. Besides, we also set up eight TCP(FTP) sessions between s_1 and d_0. Different from the UDP traffic, these good TCP sessions start at 0.0 and end at 305.0.

We run the simulation when the type of $\langle r_0, r_1 \rangle$ is I4 or Droptail or RED (Random Early Dropping) during the DoS attack. Fig. 9 shows the total throughput of eight TCP sessions to d_0 averaged every 25 seconds with each type of queue. Fig. 8 shows the percentage of bad packets forwarded by each type of queue. The simulation results demonstrate a seven fold improvement in TCP throughput and a four-fold reduction in the percentage of bad packets.

VI. ADVANTAGES OF I4

A. Incentive of support

With I4, the participating domains could enjoy valuable information that complements their local knowledge and provides a more comprehensive view of the Internet activities. This collaboration mode has proven to be more effective than doing-it-alone mode. For example, in the DoS attack, not only the customer domain can avoid the saturation of its link, but also the ISP domain can reduce its network traffic load and serve its customer better. We believe that there are mutual

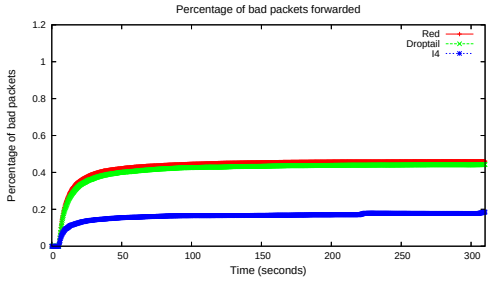


Fig. 8. Percentage of bad packets forwarded by various types of queues

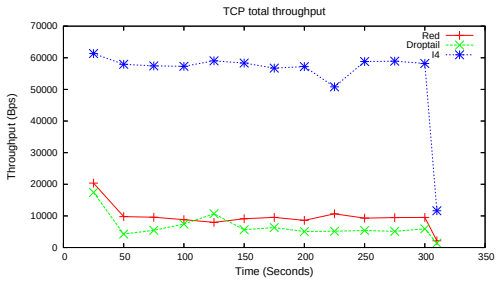


Fig. 9. Throughput of TCP sessions with RED, Droptail and I4 queuing

benefits, thus strong incentives, for domains to collaborate together by deploying I4.

B. Incremental deployment

I4 can be deployed incrementally. Even when only a small number of ISPs deploy I4 in the current power-law Internet, a large portion of unwanted traffic could be filtered out. Reference [9] shows that 50 ASes with the highest node degrees could cover approximately 90% of all the paths and thus are able to examine most of the Internet traffic. Deploying I4 in these big ASes can also provide scalability because an end domain does not have to communicate with many other domains. In summary, both the ISPs and the customer domains can enjoy immediate benefits even with a small number of deployments, which in return attracts more and more domains to participate.

C. Efficiency and Scalability

The information exchange procedure is stateless, efficient and robust against the state or resource exhaustion attack and the unexpected situations, such as packet loss, reordering etc. Considering the burdensome recovery costs, the design of I4 makes a good choice in the tradeoff between reliability and efficiency. By designating one or several agents responsible for the task of information interaction of the whole domain, the number of nodes to be upgraded in the Internet remains minimal; thus this kind of hierarchy organization provides scalability[11]. Moreover, the routers can aggregate the flows, e.g. based on the network prefix, thus the amount of information may scale well even with large number of network flows.

[11]If each router performs the functionality of I4 agent, the communication delay can be reduced. So there is a tradeoff between scalability and performance.

D. Universal

I4 is a universal architecture that can help tackle a large range of problems. For example, during worm outbreak, the capability of intrusion and anomaly detection in one customer domain can discover the worm signature. With I4, this knowledge can be further distributed to other domains; thus the spread of worm can be stopped much faster than before. Also the victim domain could generate the signature of DDoS attack traffic if possible and distribute this knowledge to its upstream domain to throttle the attack. Even when the signature is not available, as we show before, with the information of just one packet accumulated together, i.e. whether this packet received by the victim is "good" or "bad", the ISP domain can preferentially drop attack packets and thus save more good packets.

I4 can also help with the network diagnosis. Today only a few *e2e* measurement tools, such as traceroute and ping, and perhaps BGP information are available to help detect and diagnose the network problem, which provides limited and sometimes confusing information. With I4, the ISP domain can provide the information, such as the link condition, the statistic of flows, the root-cause of network failure, to the customer domain so that the customer domain can recover from the diaster or leverage this information to improve the *e2e* performance, such as by multi-path routing, source routing etc.

VII. Discussion

A. Security analysis

The attacker may try to evade, disable or even attack I4. In the following, we discuss related threats and countermeasures.

To eavesdrop, modify, intercept and even drop the I4 packet, the attacker must attach to the same routing paths taken by these packets. This prerequisite raises the bar to launch this kind of attack and also limits the scope of potential attackers to be at certain locations. The communicating peers can establish the security association (SA) to protect the confidentiality and integrity of information exchanged based on distributed keys. Note that even with a SA a Man-in-Middle can still drop the I4 packet. Other methods, such as "WATCHER" [20], can be used to detect disruptive routers.

The attacker may try to flood the customer domain with forged queries. If the attacker is inside an I4 domain, the security association between the border router and the I4 agent can detect the forged I4 packet. If an attacker is inside a regular domain or an I4 domain is compromised, the query flood can be injected into the Internet. This issue can be addressed by rate-limiting the queries at border routers. For example, for a certain aggregate, if the ratio between the number of queries and the number of data packets is significantly larger than a threshold (e.g. the query generation probability), the extra queries will be dropped. Although the query generated by a good domain can be dropped, the forged query actually makes the system more effective because it exactly catches an attack packet. Also the impacts on the percentage estimation should be little as the algorithm works well with a small number of queries. Another approach is to identify the domains where excess queries come from and then inform other benign I4 domains to block I4 packets from those domains. This would further motivate the local investigation.

The attackers may try to overwhelm the ISP domains by flooding with answer messages. As the forged answer messages do not contain the valid cookies generated by a sequence of I4 domains, they will be dropped when arriving at the first I4 domain, which significantly saves the network bandwidth.

A greedy/malicious customer domain may try to achieve more benefits by providing wrong answers. For example, it may identify an "attack" packet as "good" intentionally so that the probability for its aggregate to be dropped is smaller. However it ends up with receiving more unwanted packets, which wastes its resources and adversely affects the performance of its legitimate users. Furthermore, in order to incent an honest answer, the ISP domain can provide differentiated services to aggregates. For example, the aggregate containing a higher percentage of bad packets is assigned a larger probability of query generation. Thus the percentage estimation is more accurate and the changes can be detected more quickly. Note that the total number of queries generated is still kept the same. Thus the incentive to provide an honest answer is increased. In the future we plan to apply game theory to analyze the interaction of different answer strategies.

Last but not least, our information exchange procedure provides anonymity and recoverable privacy, as the identity of original I4 domain could be encrypted during the transmission. This would further increase the incentive of participation, especially when the ISP domain may concern that the information provided becomes the evidence against itself later.

B. Availability of I4 under stress

The availability of I4 service is important to resist unwanted traffic in the Internet. If there is more unwanted traffic, the I4 agent more likely generates an effective query, which in return helps remove the unwanted traffic. In other words, I4 is self-reinforcing and self-protecting. Also a step-by-step approach is possible: Firstly, the victim domain informs the upstream domain of the acceptable amount of traffic when a severe DDoS attack is detected, as in [10]. After the upstream domains stop forwarding the excess traffic, the victim domain provides more answer messages to help the upstream I4 domain drop more "bad" packets. We plan to conduct further experiments in the test bed to evaluate these ideas.

VIII. Related Works

Our work leverages on many previous works in the literature. Due to the limitation of space, we focus on the DDoS related works.

Early works in this field primarily target at the spoofing DDoS attack. Ingress filtering [13] prevents this attack by checking whether the source IP address falls into the network prefix of an edge domain. However there is no strong

incentive of deployment because 1) the effectiveness of such mechanisms depends on universal deployment; 2) the attacker is able to evade this mechanisms with just minor efforts. iTrace/traceback [1] [2] [3] [4] [5] is proposed to traceback the true origin of spoofed packets by providing additional (path) information to the victim. Despite a significant step, it fails to consider the incentives of deployment: the information provided to victims cannot help stop the unwanted traffic remotely injected into the Internet. Given that the legal actions may take a long time to start, ISPs and victims do not see the immediate benefits to justify the cost of deploying iTrace/traceback. References [7] and [16] proposed to help filter the spoofed packet based on either the embedded path informaiton or "hop count" at the edge of domains. However, the attack traffic cannot be dropped early.

ACC/Pushback [11] proposed to rate-limit the high-volume aggregates during link congestion and to further push such information back to upstream routers. In [10], a server under stress installs router throttles in the upstream routers so that excess traffic is dropped before arriving. However neither can distinguish the legitimate traffic from the attack traffic. I4 can be combined with them to drop more attack packets.

SIFF [8] and TVA [17] proposed the concept of "capability" that allows an end host to selectively drop the unwanted packets. Our proposal can provide the same functions as them. The main differences are as follows: 1) SIFF and TVA implicitly assigns a lower priority to a current flow by not renewing the previous capability. Instead, we explicitly send the feedback in a statistically generated answer message so that the information of individual packets can be accumulated into the knowledge of aggregates. 2) With undirectional traffic and short flows, both SIFF and TVA are less efficient; moreover TVA has to adjust the bandwidth for the initial requests based on different types of traffic. Our mechanism utilizes "trend" in the traffic and the extracted knowledge to preferentially drop the attack packets in any kind of traffic. 3) TVA used "fair-queuing" to further prevent the flood of initial requests. Our mechanism assigns the different priorities to different initial request aggregates, so that more requests from attackers are dropped. In fact, our proposal can be combined with TVA to achieve both fairness and prioritization. For example, we can reserve some bandwidth for each flow and allocate the rest based on priorities.

Furthermore, there are a lot of works on analyzing and detecting the DDoS traffic based on statistical methods, such as [6] [15]. Reference [14] reports the DoS attack prevalence and dynamics in the Internet. These works greatly help us understand the DDoS attack.

IX. Conclusions

We have presented the design of I4, a network infrastructure of information interaction in the Internet. We demonstrate the advantages of I4 in the case of DDoS attacks. With I4, the customer domain expresses its preferences about the current flows to the ISP domain so that the unwanted traffic can be dropped early. We develop algorithms to tackle the practical challenges related to the information exchange and knowledge learning. Our simulation results and theoretical analyses show that the performance can be greatly improved with the information exchanged. Compared with previous works, our proposal can handle many different types of unwanted traffic. The design of I4 also makes it easy to bear to practice.

X. Acknowledge

This work is supported in part by NSF award # 0520333 and AFOSR (Grant FA9550-04-1-0159).

References

[1] S. M. Bellovin, "ICMP Traceback Messages", Internet Draft, March 2000.
[2] S. Savage, D. Wetherall, A. Karlin, and T. Anderson, "Practical Network Support for IP Traceback", *Proceedings of ACM SIGCOMM*, August 2000.
[3] A. C. Snoeren, C. Partridge, L. A. Sanchez, C. E. Jones, F. Tchakountio, S. T. Kent, and W. T. Strayer, "Hash-Based IP Traceback", *Proceedings of ACM SIGCOMM*, August 2001.
[4] D. Song, and Adrian Perrig, "Advanced and Authenticated Marking Schemes for IP Traceback", *Proceedings of IEEE INFOCOM*, April 2001.
[5] A. Mankin, D. Massey, C.-L. Wu, S. F. Wu, and L. Zhang, "On Design and Evaluation of Intention-Driven ICMP Traceback", *Proceedings of IEEE International Conference on Computer Communications and Networks*, Octobor 2001.
[6] A. Hussain, J. Heidemann, and C. Papadopoulos, "A Framework for Classifying Denial of Service Attacks", *Proceddings of ACM SIGCOMM*, August 2003.
[7] A. Yaar, A. Perrig, and D. Song, "Pi: A Path Identification Mechanism to Defend against DDoS Attacks", *Proceedings of IEEE Symposium on Security and Privacy*, May 2003.
[8] A. Yaar, A. Perrig, and D. Song, "SIFF: A Stateless Internet Flow Filter to Mitigate DDoS Flooding Attacks", *Proceedings of IEEE Symposium on Security and Privacy*, May 2004.
[9] Y. Xie, V. Sekar, D. Maltz, M. Reiter, and H. Zhang, "Worm Origin Identification Using Random Moonwalks", *Proceedings of IEEE Symposium on Security and Privacy*, May 2005.
[10] D. Yau, J. Lui, F. Liang, and Y. Yam, "Defending against distributed denial-of-service attacks with max-min fair server-centric router throttles", IEEE/ACM Transactions on Networking, Volume 13, Issue 1 (February 2005), Pages: 29 - 42, Year of Publication: 2005, ISSN:1063-6692.
[11] R. Mahajan, S. M. Bellovin, S. Floyd, J. Ioannidis, V. Paxson, and S. Shenker, "Controlling High Bandwidth Aggregates in the Network" (Extended Version), July 2001.
[12] J. Ioannidis, and S. M. Bellovin, "Implementing Pushback: Router-Based Defense Against DDoS Attacks", *Proceedings of Network and Distributed System Security Symposium*, February 2002.
[13] P. Ferguson, and D. Senie, "Network Ingress Filtering: Defeating Denial of Service Attacks Which Employ IP Source Address Spoofing", RFC 2267, January 1998.
[14] D. Moore, G. M. Voelker, and S. Savage, "Inferring Internet Denial-of-Service Activity", *Proceedings of USENIX Security Symposium*, August 2001.
[15] H. Wang, D. Zhang, and K. G. Shin, "Detecting SYN Flooding Attacks", *Proceedings of IEEE INFOCOM*, 2002.
[16] C. Jin, H. Wang, and K. G. Shin, "Hop-Count Filtering: An Effective Defense Against Spoofed Traffic", *Proceedings of ACM CCS*, October 2003.
[17] X. Yang, D. Wetherall, and T. Anderson, "A DoS-limiting Network Architecture", *Proceedings of ACM SIGCOMM*, August 2005.
[18] `http://nms.lcs.mit.edu/activeware/`
[19] `http://i3.cs.berkeley.edu/`
[20] K. A. Bradley, S. Cheung, N. Puketza, B. Mukherjee, and R. A. Olsson, "Detecting disruptive routers: A distributed network monitoring approach", *Proceedings of IEEE Symposium on Security and Privacy*, May 1998.
[21] L. Subramanian, V. Roth, I. Stoica, S. Shenker, and R. H. Katz. "Listen and Whisper: Security Mechanisms for BGP", *Proceedings of NSDI*, March, 2004.