An Autonomic Approach to Denial of Service Defence*

Erol Gelenbe, Michael Gellman, and George Loukas
Department of Electrical & Electronic Engineering
Imperial College, London SW7 2BT
{e.gelenbe,m.gellman,georgios.loukas}@imperial.ac.uk

Abstract

Denial of service attacks, viruses and worms are common tools for malicious adversarial behaviour in networks. In this paper we propose the use of our autonomic routing protocol, the Cognitive Packet Network (CPN), as a means to defend nodes from Distributed Denial of Service Attacks (DDoS), where one or more attackers generate flooding traffic from multiple sources towards selected nodes or IP addresses. We use both analytical and simulation modelling, and experiments on our CPN testbed, to evaluate the advantages and disadvantages of our approach in the presence of imperfect detection of DDoS attacks, and of false alarms.

1 Introduction

A network security attack may be launched at any time by a teenager, an insider, a criminal, an industrial spy, or even a foreign government [1]. In February, 2000 a teenager caused billions of dollars in damage to some of the leading Internet organisations, including Yahoo.com, eBay.com, Amazon.com, Buy.com and CNN.com. Many of these are *Denial of Service (DoS)* attacks, which are sometimes combined with worms and viruses, whose main purpose is to render a network's resources unavailable to legitimate users. Today such attacks are typically distributed (DDoS), where the attacker uses a large number of compromised computers to attack one or more targets simultaneously, with the problem worsening when "reflection" is used [2].

Complicating defence is the difficulty involved in deducing the true location of the source injecting traffic into the network because of source address spoofing. In his 1985 paper on TCP/IP weaknesses [3], Morris writes: "The weakness in this scheme [the Internet Protocol] is that the source host itself fills in the IP source host id, and there is no provision in TCP/IP to discover the true origin of a packet". The difficulties involved in tracing and shutting off the large numbers of DoS flows participating in attacks make them both dangerous, and complicated to defend against.

Due to inadequacies in the original IP protocol, additional measures have been suggested after the fact to combat the problem of source address spoofing. Ingress Filtering [4] configures routers to drop arriving packets that have illegitimate source addresses (i.e., IP addresses outside an "acceptable" range). It has also been suggested that the real IP address can be inferred with a technique called IP traceback [5], which uses probabilistic packet marking to allow the victim to identify the network path traversed by attack traffic without requiring interactive operational support from Internet Service Providers (ISPs). Another scheme is hop-count filtering [6], which exploits the fact that, although the attacker can forge any field in the IP header, he/she cannot falsify the number of hops a packet needs to reach its destination starting from its source address. This hop-count information can be used in attack detection and defence. Aggregate Congestion Control with Pushback [7] attempts to consider *aggregates* of traffic flows for purposes of classification and defence, using pushback messages to signal upstream nodes to

Another area of DoS mitigation is the use of overlay networks to control different flows of traffic. Secure Overlay Services (SOS) [8], which is geared toward supporting Emergency Services or similar types of communication, reduces the probability of successful attacks by (i) performing filtering near protected network edges, pushing the attack point perimeter into the core of the network, where high-speed routers can handle the volume of the attack traffic, and (ii) introducing randomness and anonymity into the architecture, thus making it difficult for an attacker to target nodes along the path to a specific SOS-protected destination. Another recent approach is a distributed framework called DEFCOM [9], which enables the exchange of information and services between defence nodes. These approaches provide ways to mitigate DoS attacks, yet an obvious shortcoming is the fact that a compromised overlay node could damage the operation of a large part of the net-

*This work was supported by the UK Engineering and Physical Sciences Research Council under Grant GR/S52360/01.

Proceedings of the Sixth IEEE International Symposium on a World of Wireless Mobile and Multimedia Networks (WoWMoM'05)
0-7695-2342-0/05 $20.00 © 2005 IEEE

work.

1.1 CPN-based DDoS Defence

We consider a DDoS defence scheme based on the following principles. An attacked node may be informed by a local or distributed detection scheme about the ongoing assault. All nodes upstream from that node, up to the source(s) of the attack, will be informed and will take agreed upon precautions. However, the detection scheme is imperfect so that both false alarms and detection failures are possible. Such imperfections concern both the attack as a whole, and also the identification of specific packets which may or may not play a role in the assault. Thus the probability of correct detection is less than one, and the probability of a false alarm is larger than zero. The reaction of the targeted node, and of the nodes that are upstream from it towards the source(s) of the attack(s), is to drop packets which are thought to be part of that attack.

The Cognitive Packet Network (CPN) [10, 13] is a Quality of Service (QoS)-driven routing protocol in which each flow specifies the QoS metric (e.g. delay, loss, jitter, or other composite metrics) that it wishes to optimise. Payload in CPN is carried by dumb packets (DPs), while smart packets (SPs) and acknowledgement packets (ACKs) gather and carry control information which is used for decision making.

In CPN, each flow specifies its QoS requirements in the form of a QoS "goal". SPs associated with each flow constantly explore the network, and obtain routing decisions from network routers based on observed relevant QoS information. SPs store the identities of the nodes they visit, and collect local measurements such as times and loss rates. At each CPN node, the SP uses a local reinforcement learning algorithm based on measurements collected by previous SPs and ACKs, to elicit a decision from the node as to the next hop to travel to. When an SP reaches the destination node of the flow, an ACK packet is generated and returned to the source according to the opposite (destination to source) path traversed by the SP, but from which all node repetitions have been removed by using a right-to-left deletion algorithm to delete the sub-paths between identical nodes. When the ACK reaches the source, the forward route, which is the reverse of the route that it used, is stored for subsequent payload or dumb packets (DPs) which will be source-routed to the destination.

Our CPN-based DDoS defence technique exploits the ability of CPN to trace traffic going both down- and upstream thanks to SPs and ACK packets, so as to facilitate the stifling of malicious traffic. When a CPN node detects a DoS attack, it will use the ACKs to ask all intermediate nodes upstream to drop packets of the incoming flow. Detection is achieved by allowing any node to determine for itself two parameters governing bandwidth allocation: the maximum that it is able to receive (B_{TOT}), and the maximum that it is willing to allocate to any particular flow that traverses it (B_{Client}); both are dynamic parameters that may change over time as a function of the conditions at the node, and on the identity and QoS needs of the flows, and they may also vary during the life of a particular flow or connection. This idea can be extended to allowing a node to specify different bandwidth restrictions for flows of different QoS classes.

When a CPN router receives an SP or DP from a flow that it has not already seen before (e.g. with a new source-destination pair, accompanied possibly by a new QoS class), it will send a specific Flow-ACK packet back to the source along the reverse path, and inform the source of its (B_{Client}) allocation. This may occur periodically for each ongoing flow. The node will monitor all of the flows that traverse it, and drop some or all of the packets of any flow that exceeds this allocation. When the allocation is exceeded, the node informs (using ACKs) upstream nodes that packets of this flow should be dropped. Other possible actions could include diverting the flow into a "honeypot", or into a special overlay network used for protection, or it may simply alert a network administrator.

To illustrate this approach, we performed a DDoS attack in our CPN testbed shown in Figure 1 (top left). A 220KB/s MPEG1 video was streamed over UDP from node 3 to node 30. The video in the unattacked network is clear, as shown in Figure 1 (top right). Then, a saturating DDoS attack is launched from nodes 1 and 2 against node 30. This attack corrupts the video stream, making it unintelligible (Figure 1, bottom left). Then we enable our defence algorithm, alleviating the impact of the attack, resulting in a clear video stream shown in Figure 1 (bottom right). These results are quite promising; they show that given a system which is (even imperfectly) able to distinguish attack traffic from valid traffic, a sensitive real-time data stream can be protected.

In the next section we discuss a mathematical model, measurements on the small testbed of Figure 2, and simulations of the testbed, under different conditions of detection. The results obtained show strong agreement between all three methods: see Figure 3 for Webserver 0 without (top) and with (bottom) attack. Further results are given in the next section.

2 Performance evaluation

In this section we evaluate the performance of our defence mechanism experimentally, as well as via simulations and an analytical model. All three techniques are applied to the small network shown in Figure 2. Nodes 3, 4 and 5 attack Webserver 0 with flows of 2500 $packets/sec(pps)$

Proceedings of the Sixth IEEE International Symposium on a World of Wireless Mobile and Multimedia Networks (WoWMoM'05)
0-7695-2342-0/05 $20.00 © 2005 IEEE

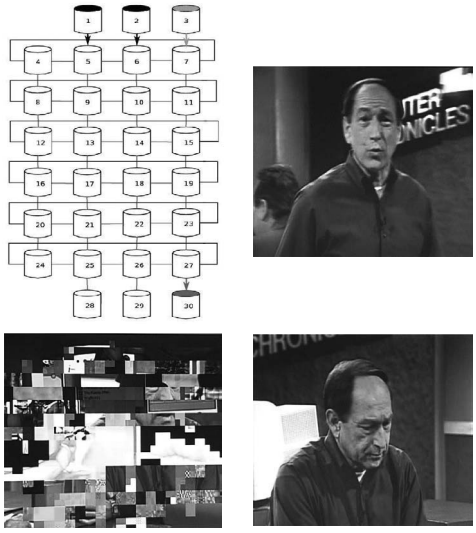

Figure 1. Experimental evaluation of our defence scheme. The top-left figure shows the CPN testbed used to conduct the experiments. Top-right shows a frame of video in the unattacked network. Bottom-left demonstrates the corruption in the video stream due to the attack. Bottom-right shows the restored video sequence after defence is enabled.

each. Both webservers (nodes 0 and 13) receive requests from all valid clients. We evaluate the impact of the attack and the defence mechanism by considering the rate of "valid" packets which make it safely to their destination nodes (*goodput*), at each node under varying load levels and different detection probabilities.

We first describe the analytical model we have constructed for a general packet network consisting of N nodes. At any node i, the arriving traffic is the aggregate of several "normal" (benign) flows, and possibly of several DoS flows, where $\mathbf{n} = (n_1, n_2, \ ... \ , n_j, \ ... \ , n_{L(\mathbf{n})})$ and $\mathbf{d} = (d_1, d_2, \ ... \ , d_j, \ ... \ , d_{L(\mathbf{d})})$ are the paths in a normal and a DoS flow, respectively. $L(\mathbf{n})$ is the path length of flow $\mathbf{n}$, and j is used to denote the position of a generic node inside the path. The total traffic rate λ_i arriving externally to node i is composed of two parts:

$$\lambda_i = \sum_{\mathbf{n}} \lambda^n_{i,\mathbf{n}} + \sum_{\mathbf{d}} \lambda^d_{i,\mathbf{d}}, \qquad (1)$$

where $\lambda^n_{i,\mathbf{n}}$ is the "normal" or benign incoming traffic rate which belongs to normal flow n, and $\lambda^d_{i,\mathbf{d}}$ is the arrival rate of DoS packets belonging to flow $\mathbf{d}$.

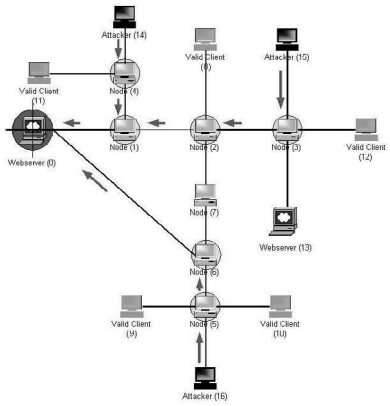

Figure 2. A DDoS attack takes place against Webserver 0; nodes 1, 2, 3, 4, 5, and 6 participate in the defence

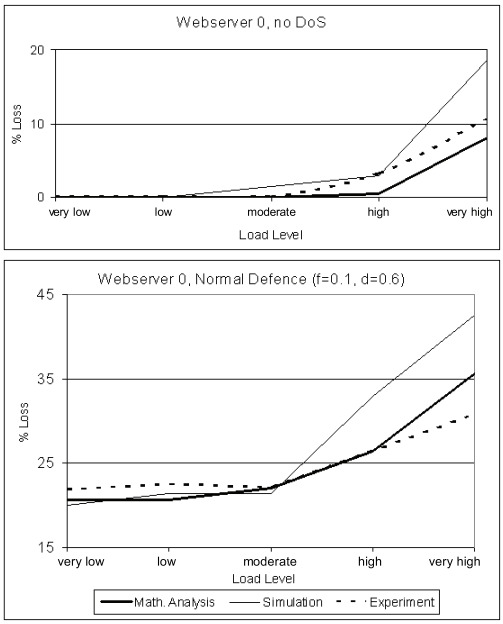

Figure 3. Packet loss at node 0: analysis, simulation and measurement without (top) and with (bottom) DDoS attack for different load levels

Node i will have the capability to recognise the DoS traffic, though only in an imperfect manner; some DoS traffic will be mistakenly taken to be normal traffic while the remaining DoS traffic will be correctly recognised by the node for what it is. Similarly, some normal traffic will be

Proceedings of the Sixth IEEE International Symposium on a World of Wireless Mobile and Multimedia Networks (WoWMoM'05)
0-7695-2342-0/05 $20.00 © 2005 **IEEE**

mistakenly thought to be DoS traffic. Any traffic that node i takes to be DoS traffic is dropped at the entrance to the node. Thus, a fraction $f_{i,\mathbf{n}}$ of normal traffic (the probability of false alarms) and a fraction of DoS traffic $d_{i,\mathbf{d}}$ (the probability of correct detection) will be dropped as it arrives to the node. If the node's DoS detection mechanism were perfect we would have $f_{i,\mathbf{n}} = 0$ and $d_{i,\mathbf{d}} = 1$. Once a packet is admitted into a node, it is queued and then forwarded based on its destination address. We model each node by a single server queue with service time s_i representing both the time it takes to process the packet in the node and the actual transmission time. The traffic intensity parameter ρ_i is then:

$$\rho_i = s_i(\sum_{\mathbf{n}} I^n_{i,\mathbf{n}}(1 - f_{i,\mathbf{n}}) + \sum_{\mathbf{d}} I^d_{i,\mathbf{d}}(1 - d_{i,\mathbf{d}})), \quad (2)$$

where for node i, $I^n_{i,\mathbf{n}}$ is the arriving traffic rate of the normal flow $\mathbf{n}$, and $I^d_{i,\mathbf{d}}$ is the arriving traffic rate of a DoS flow $\mathbf{d}$.

Since DoS attacks will tend to overwhelm the node's packet processing and transmission capability, packets will be lost by the node with probability L_i. We could use different formulas to relate traffic intensity to the buffer overflow probability, based on large deviation calculations or based on empirical observations. However, for the sake of simplicity, we use loss probability expressions for a finite capacity queueing model [11].

Since any traffic that is correctly or mistakenly thought to be DoS traffic is dropped at the input of the node, and since the traffic which effectively enters a node has been filtered in this manner, the traffic equations for the system become:

$$I^n_{n_j,\mathbf{n}} = \lambda^n_{n_1,\mathbf{n}} \prod_{l=0}^{j-1}((1 - L_{n_l})(1 - f_{n_l,\mathbf{n}}))$$

$$I^d_{d_j,\mathbf{d}} = \lambda^d_{d_1,\mathbf{d}} \prod_{l=0}^{j-1}((1 - L_{d_l})(1 - d_{d_l,\mathbf{d}})), \quad (3)$$

where we set $L_{n_0} = L_{d_0} = f_{n_0,\mathbf{n}} = d_{d_0,\mathbf{d}} = 0$. These equations express the fact that, at any node, an incoming packet may be dropped due to correct or mistaken identification as a DoS packet, or due to buffer overflow because the node is overloaded, while all packets which enter the buffer queue and are not dropped are eventually routed to the next node on their path or absorbed at the current node if it is itself the destination node. Equations (3) show the dependence of the traffic rates to the buffer overflow or loss probabilities, while ρ_i and consequently the buffer overflow probabilities L_i in turn depend on the traffic rates. The solution of (3) is obtained numerically via a non-linear iteration.

To evaluate the effectiveness of our scheme, we measure the goodput at each node. This both establishes the effectiveness of our DDoS protection scheme, and also of how successful or unsuccessful the DDoS attack has been. The goodput $G(i)$ at each node is:

$$G(i) = \sum_{\mathbf{n}} I^n_{i,\mathbf{n}}(1 - L_i)(1 - f_{i,\mathbf{n}}) \quad (4)$$

Let us illustrate the use of this model to evaluate the impact of an attack in the network of Figure 2 when a DDoS attack takes place against Webserver 0. The results are summarised in Figure 4. They show that if we do not apply a defence, a moderate attack could cause the network's performance to degrade drastically. For example, at high load the victim (0) webserver's goodput is less than 22%, compared to 99% without the attack. If we apply a simplistic defence in which half of the packets which are destined to 0 are dropped, we see that the overall goodput is improved, although the "victim" webserver suffers. Results are presented for a small probability of false alarm $f = 0.1$ and for two different levels of attack detection $d = 0.6$ and $d = 0.9$. We observe a significant improvement of goodput for both webservers at all load levels, especially with $d = 0.9$.

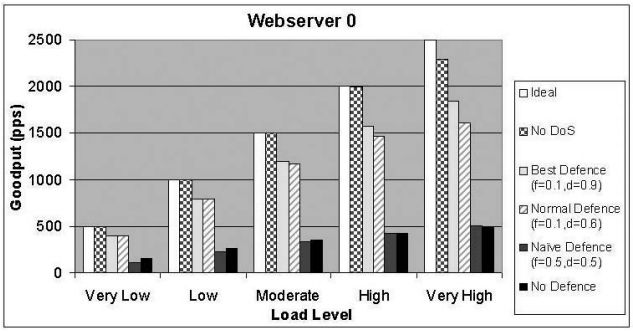

Figure 4. Analytical results for the impact of DDoS on Webserver 0

We carried out simulations for the same network configuration with the *ns-2* tool [12] employing simple FIFO scheduling, and drop-on-overflow buffer management at the nodes. We first left the network undefended and then applied a simple defence mechanism based on packet drops. We measured the percentage of legitimate packets which are lost at each node. The results are shown in Figure 5, where the x-axis is the network's load level and the y-axis shows the loss percentage of legitimate packets at each node for each load level. Then we considered the network's performance under attack when it is protected. Nodes 1, 2, 3, 4, 5 and 6 are made aware of the DDoS attack against webserver 0 and apply the defence mechanism, while node 7 is not in any of the DoS paths and will not participate in the defence.

The results both with and without defence in Figure 5 show significant improvement for each node at each load

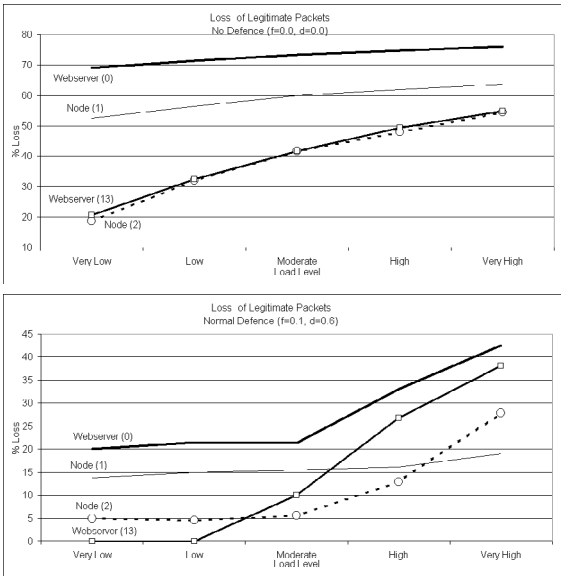

Figure 5. Simulation results for the effect of DDoS on selected nodes' goodput

level. Additionally, they highlight the usefulness of our approach in choosing a detection strategy. They show that if webserver 0 is a decoy and the critical webserver that we want to protect is 13, then at lower load levels a *naïve* defence, in which all nodes drop 50% of transiting packets heading for webserver 0, will suffice. However, if our goal is to minimise the impact of the DDoS attack on webserver 0 in a network with a high amount of legitimate traffic, then according to these results, we need an accurate detection scheme (e.g. $f = 0.1,\ d = 0.9$).

3 Conclusions and Further Work

In this paper, we have introduced a DDoS defence scheme based on our autonomic routing protocol (CPN), evaluating it on a testbed, via mathematical modelling and with a discrete event simulation. Both the mathematical analysis and simulations have measured the useful traffic rate that the network provides in the presence of an attack for various levels of attack detection accuracy. Our testbed experiments have demonstrated the ability of our mechanisms to protect real-time traffic from the effects of flooding.

Some of the avenues for further experimentation that we are exploring are the use of our algorithms in larger-scale systems, and hardware implementation. Experimenting with larger, more heterogeneous networks allows us to examine how CPN responds to a much more dynamic environment, while designing a CPN-based approach at the hardware level will allow us to take advantage of the performance gains available due to parallelisation of packet processing. Both of them hold promise for evaluating and improving our DDoS defence method.

References

[1] CERT Coordination Center. CERT Overview - Incident and Vulnerability Trends. http://www.cert.org/present/cert-overview-trends, May 15, 2003.

[2] V. Paxson. An Analysis of Using Reflectors for Distributed Denial-of-Service Attacks. *ACM Computer Communications Review 31(3)*, Jul. 2001.

[3] R.T. Morris. A Weakness in the 4.2BSD Unix TCP/IP Software. Technical Report Computer Science #117, AT&T Bell Labs, Feb. 1985.

[4] P. Ferguson and D. Senie. Network Ingress Filtering: Defeating Denial of Service Attacks which Employ IP Source Address Spoofing. Tech. Rep. RFC 2267, Jan. 1998.

[5] S. Savage, D. Wetherall, A. Karlin, and T. Anderson. Practical Network Support for IP Traceback. *Proc. ACM SIGCOMM*, pp. 295-306, Stockholm, Sweden, Aug. 2000.

[6] S. Jing, H. Wang, and K. Shin. Hop-Count Filtering An Effective Defense Against Spoofed Traffic. *Proc. ACM Conference on Computer and Communications Security*, pp. 30-41, ISBN 1-58113-738-9, Washington DC, Oct. 2003.

[7] R. Mahajan, S. Bellovin, S. Floyd, J. Ioannidis, V. Paxson, and S. Shenker. Controlling High Bandwidth Aggregates in the Network. *SIGCOMM Computer Communications Review*, 32(3):62–73, July 2002.

[8] A. Keromytis, V. Misra, and D. Rubenstein. SOS: Secure Overlay Services. *Proc. ACM SIGCOMM*, pp. 61-72, ISBN 1-58113-570-X, Pittsburgh, PA, Aug. 2002.

[9] J. Mirkovic, P. Reiher, and M. Robinson. Forming Alliance for DDoS Defense. *New Security Paradigms Workshop*, Centro Stefano Francini, Ascona, Switzerland, Aug. 2003.

[10] E. Gelenbe, R. Lent, and Z. Xu. Cognitive Packet Networks: QoS and Performance. *Proc. IEEE MASCOTS Conference*, ISBN 0-7695-0728-X, pp. 3-12, Fort Worth, TX, Oct. 2002.

[11] E. Gelenbe and G. Pujolle. *Introduction to Queueing Networks*. 2nd Ed., Wiley, London and New York, 1999.

[12] The Network Simulator NS-2, http://www.isi.edu/nsnam/ns.

[13] E. Gelenbe, M. Gellman, R. Lent, P. Liu, Pu Su. Autonomous Smart Routing for Network QoS. Proc. First International Conference on Autonomic Computing (IEEE Computer Society), ISBN 0-7695-2114-2, pp. 232-239, May 17-18, 2004, New York.