How to Build a Dam: Fighting Application-Level DoS Attacks

Gal Badishi*
EE Department, Technion

Amir Herzberg
CS Department, Bar-Ilan University

Idit Keidar
EE Department, Technion

Background and Goals

The Internet has become a fertile ground for hostile activity. One of the simplest yet most effective attacks that can be launched over the network is a denial of service (DoS) attack. The most trivial form of a DoS attack floods the target with packets, causing congestion and consuming all available network resources. In order to employ massive attack power, the attacker usually launches a distributed denial of service (DDoS) attack, in which several subordinate hosts flood the target in concert.

Traditionally, DoS attacks were performed at the network level. As a result, various network-level solutions have emerged. One type of readily available and cheap solution uses an existing firewall or router to perform rate-limiting of traffic and filtering of packets according to header fields like address and port number. Since most reasonable networks already contain the necessary hardware regardless of DoS attacks, this solution is very appealing. However, these solutions have limited effectiveness. Spoofing of headers that match the filtering criteria can be easily performed, and although rate-limiting stops networks from being overwhelmed, it indiscriminately discards messages.

Another type of solution uses expensive devices proposed by commercial companies. These solutions rely on additional, costly hardware and software that perform complex computations to track incoming packets and decide on an action to take. Indeed, such solutions can identify and isolate the DoS attack much better than the simple solutions presented above. However, complex and expensive systems are not suitable for most organizations.

As network-level DoS defenses are becoming more readily available, we can identify a shift of trends in the attackers' strategy. Since applications tend to perform much more computations per packet than a network-level mechanism does, less traffic is needed to cause the application to exhaust all CPU resources and fail to handle valid requests. Hence, an easy application-level DoS attack is in effect.

Another testament for this important problem was given in 1999 by The Committee on Information Systems and Trustworthiness. The committee declared that defending against DoS attacks is very important, but there are no systematic design methods nor mechanisms for doing so. Moreover, the committee found that ad-hoc solutions for DoS, as used in time-sharing systems, are inadequate for use in networks. The main problem lies in the fact that no framework for reasoning about the effectiveness of any mechanism nor for comparing various solutions exists.

Leveraging Net-Level Mechanisms for Quantifiably Superior App-Level DoS-Resistance

Our work thus far makes the first step in giving answers to the serious issues posed above. We believe that the proper way to tackle the problem is through a **dual-layer approach**. On the one hand, we want to exploit cheap and simple measures at the network layer. On the other hand, we would like to leverage these network mechanisms at the application layer so as to achieve effective DoS resistance. This higher layer allows for more complex algorithms as it has to deal with significantly less packets than the network layer does, and may have closer interaction with the application. An exemplar algorithm may use random ports for communication, thus reducing the probability of spoofing to the bare minimum. We feel that such a dual approach allows for a cheap, simple, effective, and perhaps most importantly, **generic defense against DoS attacks** for entire classes of applications. Such generic mechanisms, e.g., a DoS-resistant two-party communication channel [1], or a DoS-resistant multicast protocol [2], can also be used as building blocks in devising even more elaborate solutions for intricate classes of applications.

As shown in the previous section, there is a strong need for a **formal framework** for understanding and analyzing the effects of proposed solutions to the application-level DoS problem. The main challenge in attempting to formalize DoS-resistance for the first time is coming up with appropriate models for the attacker and the environment, as well as for functionality that can be provided by network-layer mechanisms such as firewalls.

Consider for example timing assumptions. When we design a service for the Internet, it is attractive to model the environment as asynchronous, so as to allow for unpredictable

*Supported by the Israeli Ministry of Science.

delays. Conversely, we observe that an attacker that controls timing can cause DoS without sending *any* bogus messages, simply by delaying application messages long enough for an unbounded number of them to reach their target simultaneously. While DoS due to timing delays is a realistic concern, such extreme timing delays and pile-up effects are not the main concern one usually worries about in the context of DoS attacks. The more realistically harmful DoS attacks generally arise from heavy traffic of bogus requests generated by the attacker. Modeling the attacker as having complete control over timing would not allow for any meaningful discussion of such realistic attacks. Instead, we should define a model where the severity of the impact of the attack on the application is directly related to the amount of bogus messages an attacker can send. At the same time, the attacker should be allowed some realistic control over timing. Similar considerations arise when defining the attacker's ability to snoop on protocol messages and to promptly react to them. We present some model variants in [1].

We briefly sketch our proposed scheme for mitigating DoS attacks on end hosts. Our network communication is based on ports. We denote the maximum possible reception rate of the receiver by R messages per time unit. Similarly, the maximum sending capacity of the adversary is denoted by C. The ratio $\frac{C}{R}$ tells us how strong the adversary is compared to the target, e.g., the ratio $\frac{C}{R}$ can be the number of hosts under the attacker's control.

The first level of defense is implemented within a router or a similar device. We assume that there is a separate buffer with distinct resources for each port, The application can determine the number of messages to read from each buffer every time unit, as long as the total number of messages read is at most R. The messages that are delivered to the application are chosen randomly from the buffers, and the rest of the messages are discarded. We note that although this description assumes unbounded buffers, it is very easy to implement the same service using fixed-size buffers.

Using this model for the network layer, we have been able to prove some general theorems. We say that a *blind attack* is in progress, if the attacker does not know the port it needs to attack. Otherwise, we call the attack a *directed attack*. We show that a blind attack causes insignificant damage with relation to a directed attack [1], where damage is the probability of a valid message that reaches the router/firewall to be forwarded for further processing, rather than discarded.

Clearly, what is left to be done is to make sure the attacker has negligible chances of finding the port the communication is taking place on. We can now utilize the network-level mechanisms to design higher-level protocols that ensure resistance to DoS attacks. Some application-level protocols we have developed for this cause use random ports, or use pseudorandom port hopping to evade the attacker [2, 1].

Future Research Directions

The two main ideas presented above, the dual-layer approach and the formal framework, define a very large design space. We believe that this important topic allows for much research. We now list some possible research directions.

Evidently, choosing the class of applications to protect may influence the protection scheme. Thus far we have looked at multicast and two-party communication. Other application classes we are currently pursuing, such as client-server communication, may require adaptation of our proposed schemes, and clearly call for a separate analysis.

Naturally, one cannot determine the efficacy of a protocol with no proper adversary model. An extremely important research direction in this context is therefore examining a range of attacker/environment models, perhaps defining a taxonomy of them, and studying the possibilities and limits of DoS-resistance in these models. Good models allow us to analyze attacks as they happen "in the wild", and to find the adversary's best strategy, or an upper bound thereof.

Once deciding on a specific protocol to protect a class of applications, many other considerations come into mind. Should redundant ports be used? What are the tradeoffs between redundancy and over-provisioning on the one hand and success rate on the other? How far is the chosen protocol from optimality and what is the proper metric?

Additionally, it is worthwhile to consider how cryptographic solutions to DoS problems, such as "proof of work" solutions, come into play in these settings. Using cryptographic primitives in DoS-mitigating protocols may raise other questions. E.g., the two-party port-hopping communication protocols presented in [1] use a shared secret to generate the pseudorandom hopping sequence. Devising an automated, DoS-resistant initialization process that shares the secret is an important direction for future research.

As for the lower levels, it is interesting to study various other formal models for the network layer. It is important to link the theoretical research to actuality by evaluating the models' effectiveness, feasibility and utility. For example, one can examine current router implementations and determine how they relate to proposed models, and how effective they are as an infrastructure for dealing with application-level DoS attacks.

References

[1] G. Badishi, A. Herzberg, and I. Keidar. Keeping DoS attackers in the dark with pseudo-random port-hopping. Submitted for publication.

[2] G. Badishi, I. Keidar, and A. Sasson. Exposing and eliminating vulnerabilities to denial of service attacks in secure gossip-based multicast. In *The International Conference on Dependable Systems and Networks (DSN)*, pages 223–232, June-July 2004.