Network Capabilities: The Good, the Bad and the Ugly

Katerina Argyraki David R. Cheriton
Distributed Systems Group
Stanford University
{*argyraki, cheriton*}*@dsg.stanford.edu*

Abstract

Network capabilities have been recently proposed as the remedy to denial of service (DoS); the main idea is to establish priority channels that carry authorized traffic, shielding this traffic from unauthorized traffic floods. It has been claimed that capability-based solutions can prevent DoS, while requiring insignificant mechanism or state in the network, unlike any other DoS solution.

In this paper, we argue that capabilities are neither sufficient nor necessary to combat DoS. We point out that they are susceptible to "denial-of-capability" attacks, i.e., DoS against the capability distribution mechanism itself. To protect capabilities from these attacks, extra anti-DoS mechanism is required in the network; once this extra mechanism has been deployed, it can be used to protect from DoS not only capability requests, but all traffic, obviating the need for capabilities.

1 Introduction

The connection-less nature of the network layer has played a key role in Internet success, allowing for a simple and flexible network core. At the same time, it has always raised a lot of debate; every time the need for end-to-end guarantees comes up, researchers propose to extend the Internet architecture with some form of network-layer connections that provide the desired guarantees for priority traffic.

Denial of service (DoS) is the latest cause for debate. The fact that any Internet host can send IP packets to any other host without first obtaining the receiver's permission (i.e., without having established a network-layer connection) enables malicious/compromised nodes to flood a receiver with unwanted traffic. As a result, the receiver's link becomes congested, legitimate TCP connections back off, and the receiver fails to serve its legitimate traffic. This calls for a solution in which the receiver indicates to the network which traffic is legitimate and which is unwanted and then causes the former to be treated with priority.

The most likely and, at the same time, most vulnerable DoS targets are public-access servers, like auction sites or search engines. Such servers make attractive DoS extortion victims, because their viability relies strongly on their ability to offer continuous quality of service. Unfortunately, these servers are also the hardest to defend against DoS, because they typically communicate with thousands or millions of unknown clients, widely distributed across the Internet; identifying and blocking attack traffic without affecting the service provided to legitimate clients has proven to be a real challenge.

There are essentially two approaches to the problem: the "datagram" and the "connection-oriented" approach. The former relies purely on datagram service: the receiver by default allows access to all senders and explicitly denies (or limits) access to unwanted ones, by propagating filtering rules into the network [8, 4, 6]. The connection-oriented approach is its dual: the receiver by default denies (or limits) access to all senders and explicitly allows access to legitimate ones, by establishing network-layer connections with them; in order for a sender to be indicated as legitimate, it must first connect to the receiver and obtain authorization to send. Capabilities [3, 12, 13] are an efficient implementation of the connection-oriented approach.

Capabilities address the problem essentially by separating traffic into two categories: "good" traffic, i.e., traffic that belongs to established network-layer connections and is treated with priority, and "bad" traffic, i.e., traffic that failed to obtain authorization and receives best-effort treatment. The problem is that there is some traffic that falls in neither category: connection-setup requests.

In this paper, we argue that capabilities are neither sufficient nor necessary to combat DoS. First, we show that legitimate connection-setup requests are vulnerable to DoS, and the only way to protect them is with a datagram solution. Then we argue that such a datagram solution can protect not only connection setup, but all traffic in general. In short, once we have deployed sufficient DoS counter-measures to make capabilities effective, the need for capabilities is obviated.

2 Capabilities: Network-layer Connections Revisited

In this section, we give an overview of capabilities. We start by describing how capabilities work (§2.1), then describe their key benefits (§2.2) and their key weakness (§2.3): vulnerability of connection-setup traffic to DoS. We conclude that this vulnerability can only be addressed with a datagram solution.

2.1 A Green Zone for Legitimate Traffic

Capability-based communication occurs in two stages: capability setup and data transmission. Both stages involve the sender, the receiver and a set of verification nodes (e.g., upgraded routers) located on the path between the two. The setup stage can be naturally piggybacked on TCP connection setup:

1. The sender sends a capability request to the receiver.
2. Each verification node on the path stamps the forwarded request with a special mark; all these marks together constitute the "capability".
3. The receiver returns the capability to the sender.

In the data transmission stage, the sender includes the capability in all packets sent to the receiver. Each verification node then verifies the part of the capability it created; packets with a valid capability are forwarded with priority over unauthorized traffic.

Capabilities are essentially tickets with an expiration date, which enable the receiver to use its downstream bandwidth according to its own custom policies. For example, a well-known legitimate client may be given a life-long ticket that entitles it to send an unlimited number of bytes. A new, unknown client may be given a ticket to send a limited number of bytes within the next minute; if the client abuses its ticket and sends unwanted traffic, the client is classified as an attack source and is given no more tickets.

An attack source cannot send authorized traffic, but it may still attempt to flood a receiver's link with spurious capability requests. To prevent such an attack from interfering with authorized traffic, a receiver partitions its downstream bandwidth: the biggest share is dedicated to established connections and a small share (e.g., 5%) is dedicated to capability requests.

2.2 Stateless Filtering

What makes capabilities special is that they do not add per-connection state to the network. Thanks to intelligent marking, verification points do not need to know anything about the receiver or the sender of a packet in order to verify the included capability. In other words, the network can filter attack traffic without keeping any end-to-end filtering state.

Stateless filtering has two important results. First, it obviates the need for traditional packet filters, which are an expensive resource today. Second, it obviates the need for any special inter-ISP relations, such as bilateral filtering agreements, because no filtering state is explicitly exchanged between ISPs. To use capabilities, all an ISP has to do is upgrade a subset of its routers to perform the appropriate marking and verification. This is in contrast to datagram solutions, in which explicit filtering requests are propagated across ISPs and satisfied with traditional, stateful packet filters.

To summarize, capabilities are inexpensive and easy to deploy and they shield legitimate communications against DoS. Even if a receiver is flooded with unwanted traffic, its communication to legitimate clients is unaffected — at least to those legitimate clients that have already obtained a capability.

2.3 Denial of Capability

One question remains: what happens to a legitimate client that has not obtained a capability before DoS starts?

Consider a public web site connected through a 100 Mbps link; 5% of the downstream bandwidth is dedicated to capability requests and the rest to established connections. Now suppose this site is under DoS attack from $20,000$ attack sources that generate 2.5 Gbps of capability requests — the attack parameters correspond to a real, recently published DoS case [2]. As a result, the portion of the victim's bandwidth that is dedicated to capability requests is exhausted, and most capability requests are dropped. Legitimate clients that had obtained a capability before the attack started have no problem connecting to the web site. However, a new legitimate client has little chance of receiving reasonable service. We call this a "denial-of-capability" (DoC) attack, i.e., denial of service against the capability distribution mechanism itself.

Assume, for simplicity, that all capability requests are 64 bytes long. Then the web site can accept roughly $10,000$ capability requests per second, while the attack sources generate about 5 million capability requests per second. Suppose a new legitimate client retransmits its capability request every second until it gets through. The probability of this new client accessing the web site within 20 seconds is approximately 0.04.

A legitimate client must only get a single capability request through to the receiver in order to establish a connection. This means that the probability of a legitimate

client being unable to connect decreases exponentially with the number of capability requests it sends. This is definitely a good property; the question is whether it is good enough for online businesses, where response time is a critical factor for success. Continuing with the same example, if the legitimate client restransmits every second, the expected time to connection establishment is higher than 8 minutes — by which time the client has most probably given up.[1]

Capability requests constitute, by definition, datagram traffic. Hence, to protect them against DoS, we can only use datagram solutions. One suggested approach is to perform Internet-wide fair queuing of capability requests: configure a subset of Internet routers to fair-queue capability requests per incoming network interface; as a result, no interface gets to forward more than its fair share of requests. It has been argued that, if widely deployed throughout the Internet and, in particular, close to the edges, this form of policing can automatically rate-limit floods of capability requests [3, 13]. Should fair-queuing prove ineffective, a receiver can explicitly deny (or limit) access to unwanted capability requests, by propagating filtering rules into the network [13].

In summary, capabilities remove the need to protect Internet datagrams in general, but introduce a new need: to protect connection-setup datagrams against denial of capability.

3 Setup Traffic is Not Different

To protect capability distribution against DoC, we need an effective and practical datagram solution. Once such a solution exists, it can be used to protect not only capabilities, but all Internet datagrams. One could claim that this argument is incorrect, because setup requests appear to be easier to regulate than general datagrams. In this section, we argue that connection-setup traffic is in no fundamental way different from general traffic — at least not with respect to its susceptibility to DoS and the challenges involved in protecting it.

Of course, legitimate connection-setup requests *are* different from general datagrams in at least two ways: they are smaller and fewer (since each sender must get only one setup request to each receiver in order to establish communication). One might expect these two features to render setup requests easier to regulate than general datagrams.

[1]There is the option of decreasing retransmission timeouts especially for connection-setup packets. However, it is not clear how this would impact legitimate traffic and the receiver's ability to differentiate between good and bad clients — a careful study is needed to show the effects that faster setup-request retransmission would have on Internet stability.

Fair-queuing per incoming interface has been suggested as a practical means for limiting floods of connection-setup requests, although it is inappropriate for limiting general datagrams. General datagrams do not follow any global pattern; what constitutes a perfectly acceptable datagram rate for receiver A may be a fatal DoS attack for receiver B. Hence, restricting all interfaces to equal datagram rates is an arbitrary choice with unpredictable effects on legitimate traffic. Unlike general datagrams, fair-queuing connection-setup requests seems to make sense at first sight: because they are relatively small, flooding a site with setup requests requires lots of them and, since they are relatively few under normal conditions, detecting a setup-request flood should be easy.

In practice, this argument does not hold. Consider our earlier example, where $20,000$ attack sources flood a receiver with capability requests, while the receiver has dedicated 5 Mbps to connection setup. To generate 50 Mbps of 64-byte capability requests (ten times the amount the receiver can handle), each attack source would have to send less than 5 capability requests per second. An attack consisting of such low-rate flows makes fair-queuing ineffective — unless, of course, there is a fair-queuer in front of every attack source, which raises serious deployment issues. Even worse, it has been argued that we are not far from witnessing attacks coming from hundreds of thousands or even millions of attack sources [10]. In this case, preventing DoC by fair-queuing setup requests is impossible not only practically but also theoretically, because attack sources are sending at the same rate as legitimate clients.

Since the network cannot distinguish good from bad capability requests, the only solution is to have the receiver identify bad senders and ask from the network to block capability requests from these senders to the receiver. However, both in terms of mechanism and required state, this is equivalent to asking from the network to block *all* traffic from these senders to the receiver.

In summary, the resources required to protect capability requests against denial of capability would actually be enough to protect all Internet datagrams against denial of service.

4 The Datagram Approach

In this section we present the basic challenges and features of a datagram solution to DoS. We are not proposing a new, radical solution in this paper, but rather argue that the basic components of the right solution have already been identified by the research community.

Unforgeable path information inside each packet: To prevent denial of service, a receiver must police incoming datagrams, to ensure that no entity is granted an unreasonable share of the receiver's resources. To do this, the receiver must first be able to classify incoming datagrams based on source. Such classification is often impossible, because the Internet allows hosts to send IP packets using fake ("spoofed") source IP addresses. Hence, the first basic feature of a datagram solution is to include unforgeable path information inside each packet.

An effective, well-studied way to do this is *packet marking*, where a subset of upgraded Internet routers stamp forwarded packets with a mark; the receiver of a packet then uses these marks to determine the packet's path. Packet marking first appeared in probabilistic IP traceback schemes [9, 5]; Yaar et al. proposed the first deterministic packet marking scheme [11].

Propagation of filtering rules into the network: Once the victim classifies a sender as an attack source, it must cause all datagrams from that sender to be blocked *before* they enter the bottleneck link to the victim. Thus, blocking must occur at a router located in the victim's ISP at the very latest, otherwise the victim's bandwidth is exhausted and legitimate clients are denied access. Hence, the second basic feature of a datagram solution to DoS is to enable a receiver to propagate filtering rules into the network. This approach was first employed by Mahajan et al. in Pushback [8].

The challenging part is that propagating filtering rules may push significant end-to-end filtering state into the network. In order to block datagrams from n unwanted senders, a receiver needs to propagate n filtering rules, each rule describing the traffic from one unwanted sender to the receiver. The number of filtering rules can be reduced through aggregation, but then legitimate datagrams are sacrificed (in the extreme case, one aggregate rule blocks all datagrams to the receiver indiscriminately). So, a DoS victim receiving datagrams from tens of thousands of attack sources would need to propagate an equal number of filtering rules.

No ISP today supports tens of thousands of filtering rules per client; an ISP router may support that number of filters overall for all its ports. This limitation comes from cost and space, because filtering rules are typically stored in expensive, power-hungry TCAM chips [1]. Hence, it is quite likely that the victim's ISP alone cannot block all attack sources.

Scalable and secure distribution of filtering state: If the victim's ISP cannot block all attack sources, the natural solution is to push filtering state upstream, sharing the load with other providers. However, this poses scalability challenges — a tier 1 ISP providing connectivity to 100 DoS victims would need to install millions of filtering rules. The alternative is to move this responsibility from the Internet core to the edges, by pushing filtering state close to the attack sources. However, doing so poses security risks — why would a provider install a filtering rule requested by some unknown victim at the other end of the Internet? Hence, the third and most challenging feature of a datagram solution to DoS is to enable scalable and, at the same time, secure propagation of filtering state across different ISPs.

In [4] we presented AITF, which securely pushes filtering state as close as possible to the attack sources, relying on simple 3-way handshakes between the victim's network and the networks hosting attack sources. Greenhalgh et al. have presented a mechanism that pushes filtering state upstream from the victim, relying on specially equipped filtering entities located on the borders between ISPs [6].

There may be multiple datagram solutions that provide these three features; in this paper, we do not argue for which solution is better, but rather that one of them is necessary to protect capabilities against DoC. Once this datagram solution is in place, the role of capabilities in defending against DoS becomes questionable.

5 Capabilities as an Optimization

A datagram solution incurs a detection delay in blocking a DoS attack; in the time it takes the victim to detect an attack and propagate the right filtering rules, already-connected clients may experience a glitch. This can be a problem for certain servers — while unpleasant to be unable to connect to eBay, it is monetarily much worse if eBay clients that are already logged in and about to bid lose their connectivity. One could argue that, if deployed in combination with a datagram solution, capabilities can solve this problem, because they guarantee that already-connected clients are completely unaffected by DoS attacks.

Capability-based solutions, however, also suffer from the detection delay problem. Suppose an attack source requests a capability; the victim cannot predict that this is an attack source, so it authorizes the source to send a certain number of bytes; the source behaves like a legitimate client for a certain amount of time, tricks the receiver into authorizing it to send more traffic, and then sends unwanted traffic. If tens of thousands of attack sources do this at the same time, connected clients still experience a disruption until the victim detects all attack sources and stops handing them capabilities.

The value of capabilities lies in the power they give receivers to control the number of bytes sent by each source within a certain amount of time. To make this power useful, there would have to be an algorithm for setting

the "right" initial byte and time limits. But, as described above, whatever values are chosen, a malicious sender can gain the receiver's trust and then abuse it. In fact, when multiple attack sources are able to coordinate, they can inflict significant damage without sending any more traffic than a legitimate source [7]. Hence, as attack population sizes increase, it becomes less relevant to limit the number of bytes sent by each source and more important to (i) monitor traffic patterns and (ii) rapidly and explicitly deny further access to misbehaving sources.

6 Conclusion

Network capabilities suffer from the same problem that all network-layer connections have in common: vulnerability in the connection-setup mechanism. We have argued that the capability-setup mechanism is necessarily based on unauthorized datagram packets, and this datagram traffic is not intrinsically more constrained than normal Internet datagram traffic, especially in attack scenarios. Thus, a datagram solution to DoS is required for a capability-based scheme to be complete. But once a datagram solution is in place, we see no clear benefit to deploying a capability-based scheme in addition to it.

There are parallels between this weakness of network capabilities and that observed in more conventional capability-based operating systems and in security infrastructures in general. For instance, in a public-key infrastructure, the greatest challenge is key management and distribution, not data encryption and transmission. The cost and complexity of securing this control-channel portion often overwhelms the benefits of these schemes when examined closely.

Perhaps our arguments stand in contrast to the decades-long success of the telephone network. However, the telephone network relies on an administratively secured and hierarchically-structured control channel, which would suffer from the same weaknesses and attacks, if it was opened up to the degree of flexibility expected in the Internet. With no expectation of the Internet morphing into some variant of the rigidly structured telephone system, we see no solution to the DoC problem that does not obviate the need for capabilities as a means to defend against DoS.

References

[1] Access list configuration in Cisco's Gigabit Ethernet Interface. `http://www.cisco.com/en/US/products/hw/switches/ps5304/prod_configuration_guides_list.html`.

[2] DDoS against Gambling Site. `http://www.csoonline.com/read/050105/extortion.html`, May 2005.

[3] T. Anderson, T. Roscoe, and D. Wetherall. Preventing Internet Denial-of-Service with Capabilities. In *ACM HotNets*, November 2003.

[4] K. Argyraki and D. R. Cheriton. Active Internet Traffic Filtering: Real-time Response to Denial-of-Service Attacks. In *USENIX Annual Technical Conference*, April 2005.

[5] D. Dean, M. Franklin, and A. Stubblefield. An Algebraic Approach to IP Traceback. In *NDSS*, February 2001.

[6] A. Greenhalgh, M. Handley, and F. Huici. Using Routing and Tunneling to Combat DoS Attacks. In *USENIX SRUTI*, July 2005.

[7] A. Kuzmanovic and E. W. Knightly. Low-Rate TCP-Targeted Denial of Service Attacks (The Shrew vs. the Mice and Elephants). In *ACM SIGCOMM*, August 2003.

[8] R. Mahajan, S. Bellovin, S. Floyd, J. Ioannidis, V. Paxson, and S. Schenker. Controlling High Bandwidth Aggregates in the Network. *ACM CCR*, 32(3):62–73, July 2002.

[9] S. Savage, D. Wetherall, A. Karlin, and T. Anderson. Practical Network Support for IP Traceback. In *ACM SIGCOMM*, August 2000.

[10] S. Staniford, V. Paxson, and N. Weaver. How to 0wn the Internet in Your Spare Time. In *USENIX Security*, August 2002.

[11] A. Yaar, A. Perrig, and D. Song. Pi: A Path Identification Mechanism to Defend against DDoS Attacks. In *IEEE Security and Privacy*, May 2003.

[12] A. Yaar, A. Perrig, and D. Song. SIFF: A Stateless Internet Flow Filter to Mitigate DDoS Flooding Attacks. In *IEEE Security and Privacy*, May 2004.

[13] X. Yang, D. Wetherall, and T. Anderson. A DoS-limiting Architecture. In *ACM SIGCOMM*, August 2005.