2013/832

ISSN 1797-3457 (verkkojulkaisu)
ISBN (PDF) 978-951-25-2522-5

TIIVISTELMÄRAPORTTI (SUMMARY REPORT)

Reliability of computer networks during denial-of-service attacks

Tuomas Aura and Aapo Kalliola
Aalto University, Department of Computer Science and Engineering
tuomas.aura@aalto.fi, aapo.kalliola@aalto.fi

Abstract

The goal of the research project is to develop protection mechanisms against request and packet-flooding denial of service (DoS) attacks. We consider both distributed and single-source attacks as well as packet flooding with spoofed source addresses. The main solution is packet filtering based on a normal traffic model. The filters from the model are translated into an OpenFlow forwarding policy, which is implemented in a switch ad could in the future be distributed across the network using software-defined networking (SDN) technologies. We have also investigated techniques for implementing a packet-loss-resistant transport protocol that could be used for system administration during DoS attacks.

1. Introduction

Denial-of-service (DoS) attacks are one of the key security problems in modern net-work and service architectures. The Internet has become a part of the society's critical infrastructure, and both public services and businesses depend on its continuous availa-bility. This can be observed in workplaces where disconnection from the Internet or business-critical online services, e.g. during an unplanned service break or power out-age in the computing facilities, causes immediate failure of most business functions. DoS attacks can cause similar failures – at the will of a malicious attacker. Moreover, the criminal economy on the Internet and, in particular, the availability of botnets for hire, means that anyone with a modest amount of money is able to mount such an at-tack is a short time and without much expertise.

Thus, the denial-of-service attacks have developed from isolated protests and vandal-ism to everyday crime and political pressure tools. For example, web pages with politi-cally controversial topics and the service providers hosting them fall regularly under at-tacks. There are also many unexplained attacks, e.g. against universities. The likely ex-planation for these is the testing of cyber-attack capabilities by criminals or semi-governmental entities. Unfortunately, the defense mechanisms against DoS attacks are far weaker than the protection technologies in many other areas of information securi-ty. Any network-connected system is vulnerable to flooding attacks if the attacker has sufficient resources. Only the biggest global ISPs and content-network providers have sufficient capacity and global distribution to be able to resist the attacks. Smaller play-ers that need protection against DoS attacks have to outsource their infrastructure to such global providers or pay for expensive DoS-protection services to filter their traffic. This is not always possible for financial reasons and, in the light of recent spying reve-lations, may be undesirable on data-protection grounds.

For these reasons, the goal of this our three-year research project is to develop DoS protection methods for local and low-budget services, such as small-business web servers, universities, or government agencies of small countries. While it is not possible to fully prevent DoS attacks, careful design of the network and services can increase the cost of the attacks and reduce the number of potential attackers. The results are system architectures, algorithms, design principles and protocols that are evaluated with simulations, analytical models and system-level security analysis.

Good DoS-protection mechanisms do not cause a significant overhead when there is no attack. Ideal solutions increase the reliability of the system even under normal opera-tion and can, thus, be deployed all the time. This is because it is difficult to detect in re-al time which service disruptions are caused by intentional attacks. For example, the fil-tering methods developed in this project can be used to prioritize clients even when the overload of the service is caused by suddenly increased demand by honest clients (flash crowd) rather than by attackers.

2. Research objectives and accomplishment plan

During the first year of the three-year project, we developed filtering methods for packet and request-flooding attacks, and these have been further optimized and evalu-ated during the second year. The developed method is based on machine learning tech-niques, more precisely data clustering. During normal operation, the system learns a model of the normal traffic. Figure 1 shows an example of a cluster model with IP pre-fixes and observed normal and attack-time traffic counts. When an attack starts and the server is unable to respond to all requests, the model is used to prioritize traffic that is closest to the normal. The main traffic feature used in the model are the client IP address and its prefixes. When the service is overloaded, regular clients or clusters of typical client addresses are given priority over new ones, which is a natural policy for any overloaded service.

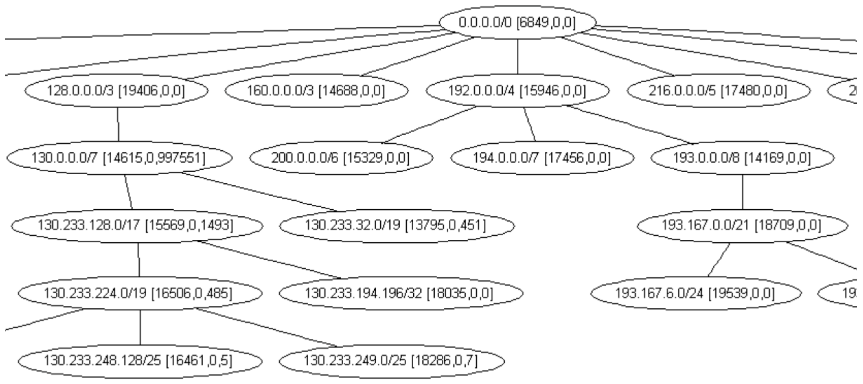

Figure 1: Hierarchical traffic model

The second year second year of the project had the following objectives:

- further optimization and evaluation of the filtering methods e.g. using simulations with real botnet address data sets,
- developing implementation techniques that could handle realistic traffic volumes, and
- exploration of complementary DoS defense techniques.

The filtering had already been evaluated with both simulations and analytical proofs, in-cluding an optimality proof of the chosen resources-allocation strategy to the clusters. We have further implemented an NS3 simulation of the filtering to evaluate it in a more realistic network model. These tools were now used to verify the hypothesis that the defense strategy works equally well against a single attack source and distributed at-tacks by botnets. This was made possible when we obtained several data sets of real botnet addresses, which were used in the simulations together with the normal client distribution of real university web servers. In the former case, the attack traffic falls into one source-address cluster, leaving the other clusters unaffected, and in the latter case, most of the attack traffic falls into the default cluster where there are few regu-lar clients. In DoS attacks with spoofed random source addresses, the attack traffic is similarly concentrated in the default cluster.

The same filtering principle can be applied both for web requests and IP packet flood-ing. Although we initially experimented with filtering on the server itself, in the current solution it takes place entirely on a firewall router or switch on the IP packet level. To counter both request and packet-flooding attacks, two sensors are needed to detect overload: one on the server itself to detect request flooding and another on the uplink router or switch to detect exhaustion of the access link capacity. The architecture, shown in Figure 2, was further refined in this year’s project.

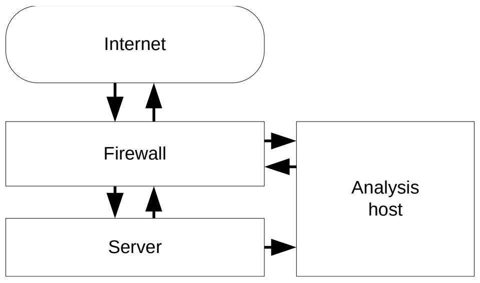

Figure 2 : High-level filtering system architecture

3. Materials and methods

The main methods used in the research were simulation and experimental implementa-tions. The main open question was how to filter realistic attack packet streams, which can reach 10 Gb/s or more. Our software filter implementation was capable of pro-cessing hundreds of Mb/s, but bandwidths over 1 Gb/s seem unreachable with PC hardware using the main CPU. This gap is too large to bridge by simply optimizing the software or by taking advantage of the multicore CPUs.

One approach that was considered but not selected for the implementation was using the GPU of a personal computer for parallel processing. This approach can perform large number of arithmetic operations in parallel. The packet filtering, however, is s relatively simple operation per packet. In an unrelated thesis project, we had observed that the bottleneck in GPU processing is moving the data to the graphics card and back, and that it is not always worth doing this for light processing per packet. It may never-theless be a direction for further investigation in the future.

Another potential approach was to distribute the packet processing to several comput-ers that form a scalable architecture. This could be an array of blade servers or even a P2P network of cooperating servers. This remains an interesting approach to be ex-plored. However, there is the critical question of how a 10 Gb/s data stream coming down one fiber would first be split into streams that can be processed by the individual CPUs. Such a load balancer itself would be an expensive piece of equipment, which we wanted to avoid.

The solution chosen in the project came from parallel developments in software-defined networking (SDN). The idea of SDN is that expensive network equipment such as high-bandwidth routers can be replaced with simpler switches that only perform the for-warding function for predefined flows. The routing logic, i.e. maintenance of the for-warding tables, is moved to an external controller. The results from this area of re-search are rapidly moving into products. In particular, there are reasonably priced 1 Gb/s and 10 Gb/s network cards and switches coming to the market that implement the OpenFlow specification. OpenFlow switches forward IP packets based on the 5-tuple of transport protocol, source and destination IP addresses, and source and destination port numbers, which identify TCP and UDP flows. Forwarding decisions for new flows are made by an external controller. This enables central control of flows and routing, for example, in a data center network. The ability to define new routing policies by re-writing the controller software has made OpenFlow a favorite tool of networking re-searchers. For example, researchers have noted that it is possible to implement distrib-uted firewall policies using OpenFlow.

Our solution is to implement the DoS filtering on a single OpenFlow switch. Since we use the OpenFlow switch as a filter, we only need two high-speed ports on the switch, keeping the cost further down. The controller samples packets from the network using a sensor, in practice the sFlow monitoring technology on the switch. It then builds a model of the normal traffic from the sampled packets and creates periodically a new filter from the traffic model. The cluster-based filter is translated to OpenFlow control commands. The latest filters are deployed continuously to the switch, but so that pack-ets in all clusters pass through the switch. However, if the link towards the server be-comes congested or the server reports overload, the policy is changed to drop the packets in selected clusters, based on an optimal filtering strategy.

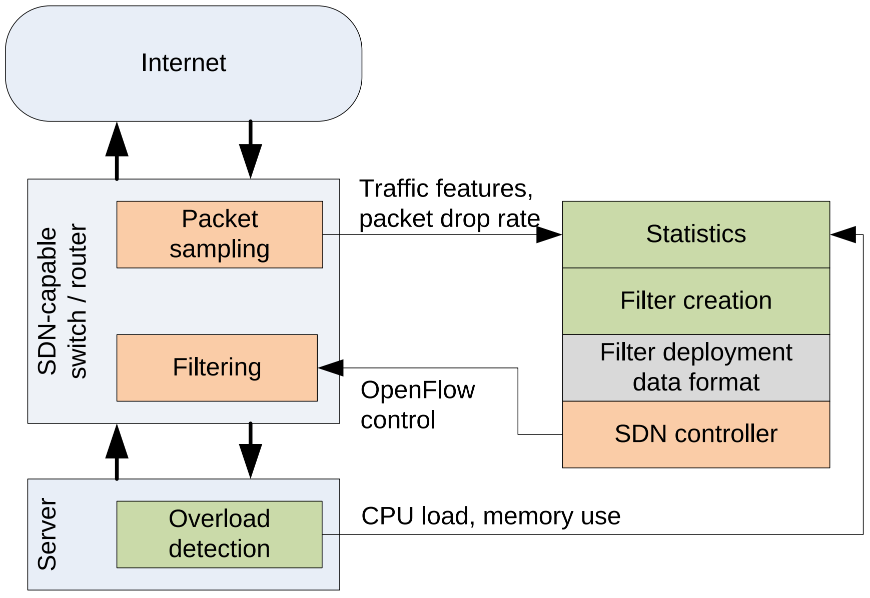

Figure 3: Detailed filtering architecture

Figure 3 shows the more details structure of the filtering system. A server running a specific service or services is connected to the internet through a switch (or router) that is OpenFlow enabled. This server may be subjected to an attack originating from multiple points in the internet. As can be seen, the filtering implementation is distributed to several parts of the system. In addition to the SDN switch that implements the sFlow and OpenFlow specifications, there is an analysis and control component, which can be typically deployed on a single host computer. This analysis and control component is connected to the sample output and control plane of the SDN switch and to the load status output of the server. Traffic sampling and filtering is located on the OpenFlow-enabled switch or router. In traffic sampling, a mechanism on the switch sends link-load and packet drop statistics and a partial copy of the server-bound traffic to the analysis and filtering control host. On the analysis host the traffic features and traffic statistics are combined with the current load information coming from the server. The current view of the state of the system is compared to old views of the system, and based on this comparison a set of blacklisted traffic clusters is created. These traffic filters are then proactively sent by the SDN controller to the SDN switch.

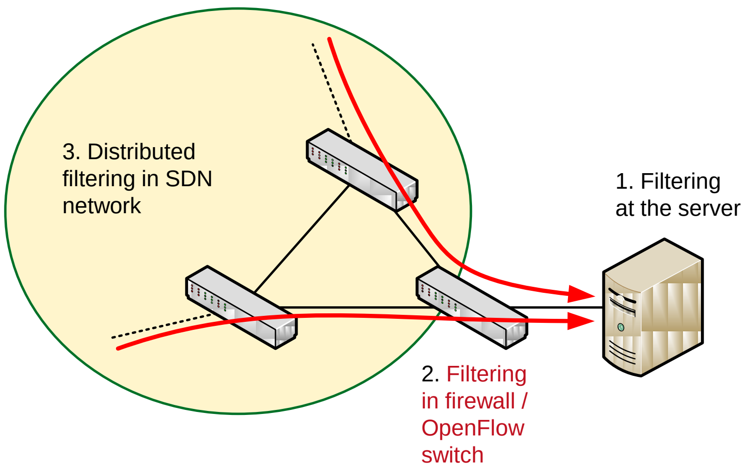
3. Distributed filtering in SDN network
1. Filtering at the server
2. Filtering in firewall / OpenFlow switch

Figure 4: Potential filtering locations

Figure 4 provides a higher-level view of the network. In our initial implementation, the filtering took place on the server itself. This works relatively well for request-flooding attacks where the server processing power is the bottleneck. In the current implementation, as described above, the filtering has been moved to the next upstream network element from the server. This was first a Linux-based firewall host but now an Open-Flow switch.

Our current experimental setup uses a software switch with OpenFlow emulation. This is because the number of OpenFlow products on the market is quickly increasing and the prices have only recently come to a range acceptable to a university project. We expect to move the implementation to the hardware switch in early 2014. This is not a significant change to the implementation, but will enable us to obtain filtering performance measurements on real hardware, which is a critical step in evaluating the ideas developed in this project.

The benefits of using SDN technology, however, do not need to stop there. SDN technology is not originally intended for single switches but as a replacement for routers and switches in an entire network. As Figure 4 shows, the network could be a fabric of SDN switches or routers. In that case, it makes sense to distribute the DoS filtering to the entire network. This has two benefits: First, distributed DoS attack filtering may need to be pushed further upstream anyway to prevent the congestion of the upstream links. Second, combining the DoS detection and filters with the routing can lead to an overall optimized forwarding policy. In the current project, however, we do not plan to implement the distributed filtering because it would not be feasible to test it in a real network. (Simulation with NS3 may be feasible.) Currently, only a few large companies have deployed OpenFlow on their internal networks. Thus, we focus on the single-switch implementation for the time being.

5. Results and discussion

The main results from the second year of the project are implementation techniques for the filtering algorithms and strategies that were previously defined.

Figures 2 and 3 show an example of the filtering efficiency during an overwhelming packet-flooding attack with random source addresses, and they also demonstrate the improvement over the early results. Figure 2 from last year shows how many more cli-ents were served during a DoS attack when the filtering was on. Figure 3 demonstrates the full potential of the filtering based on new simulation results. We have increased the attack volume to a level where no honest clients reach the server. With filtering ena-bled, over 80% of the honest requests and clients are served. The improvement is partly due to optimizations in the algorithms and partly to more accurate simulation.

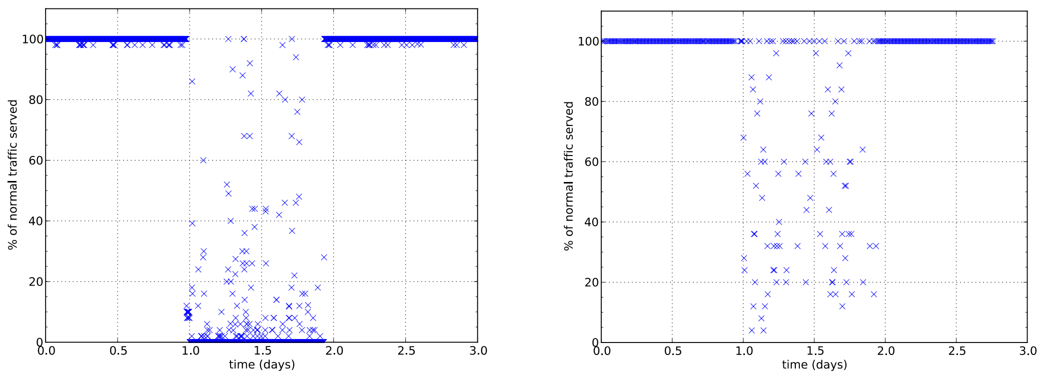

Figure 2: Simulated DoS attack with and without filtering defense, early results

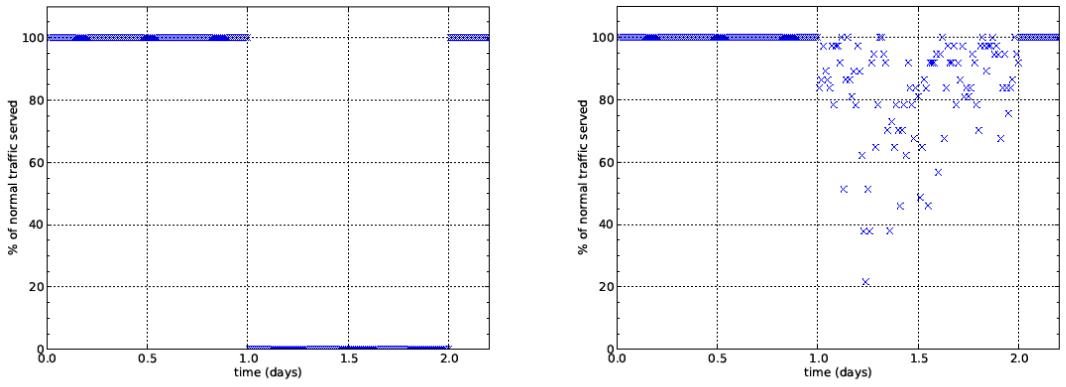

Figure 3: Simulated DoS attack with and without filtering defense, new results

The simulation results give a good idea about the accuracy of the filtering algorithms. These results have also been confirmed by experiments with the software-based filter-ing implementation. What the current results do not tell is how high traffic volumes can be handled on an OpenFlow switch that implements the filtering with hardware. For that, we will have to wait for the results from the third year of the project.

We also continued exploring alternative and complementary techniques of DoS defense. More specifically, we started developing a transport protocol that is resistant to packet dropping. The protocol sends preemptively redundant data packets, encoded with rateless erasure codes, in order to recover from the packet loss. We have partially de-fined the protocol and worked out some of the mathematical principles of controlling the number of redundant packets that need to be sent. Experimental evidence shows that low-bandwidth interactive connections can remain usable even with very high packet drop rates. Since the results are only preliminary and the technology only par-tially defined, we will defer the detailed explanation to next year's report.

5. Conclusions

We have continued the development of a filtering mechanism for local protection against packet and request-flooding denial-of-service attacks. The filters are created in that they prioritize traffic that is similar to the normal use of the service. This is achieved by utilizing recent historical information about normal server traffic flows and by comparing that information to the attack-time status of the same source address clusters. As a result of the filter deployment, only a manageable amount of traffic is al-lowed to reach the server. The filtering developed in this project can protect web serv-ers but also other online services.

Additionally, we have developed ideas and the theoretical foundations for a packet-loss-resistant transport protocol. This is still work under progress. The transport protocol should be useful for system administrators during Dos attacks.

The overall goal has been to introduce simple technologies that can be deployed locally by anyone. Our defense mechanisms have low overhead during normal operation when there is no DoS attack in progress, and they may increase the reliability of the network even during non-malicious overload or packet loss.

The attack scenario considered is one where the web servers of a small or medium-size organization fall under a flooding DoS attack. While large ISPs and content-network owners have the capacity to fight such attacks, smaller players do not always have such resources. Large companies are also able to distribute their services globally and balance the load between multiple data centers, while smaller and local organizations often have a single server or data center that is an easier target for flooding attacks. Our research results serve especially such companies and other organizations with lim-ited resources or only local presence on the Internet, enabling them to protect them-selves against DoS attacks.