Advisory on Application-layer Loop DoS Attacks

Contact

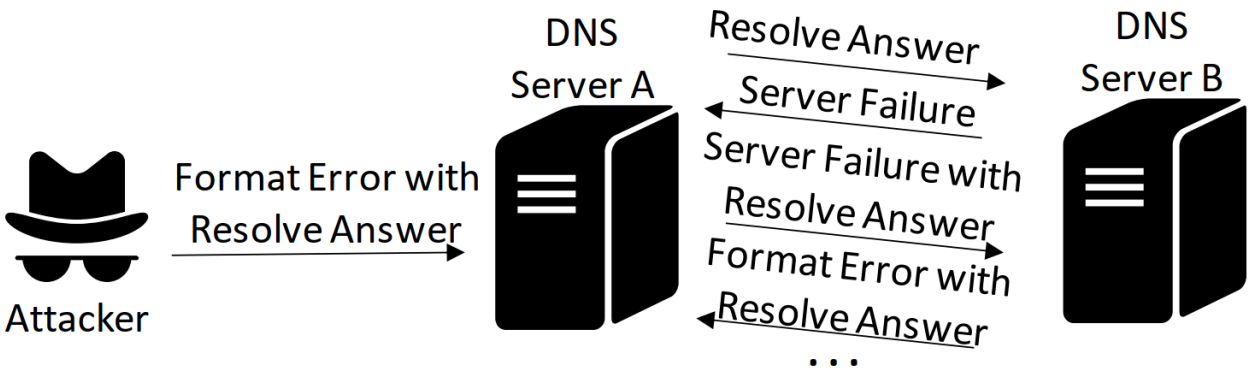
Christian Rossow and Yepeng (Eric) Pan
CISPA Helmholtz Center for Information Security, Germany
Web: https://cispa.saarland/group/rossow/
Mail: loop-dos@cispa.de

SITREP: This document describes a DoS attack vector in popular UDP-based application-layer protocol implementations. While, to the best of our knowledge, the attack is not yet abused by actors, the bar to do so is not high.

Background

Application-layer loops are a form of (D)DoS attacks. Spoofing-capable attackers can create such loops if two network services keep responding to each others' messages. For example, imagine two services that respond with an error message when receiving an error message as input. If an error as input creates an error as output, and a second system behaves the same, these two systems will keep sending error messages back and forth indefinitely.

To illustrate the attack vector, imagine two DNS resolvers with such error reflection behavior. If an error as input creates an error as output for two systems, upon receiving an attack trigger, these two systems will keep sending error messages back and forth — indefinitely. The figure below shows such an example we identified among real DNS servers. An attacker could now cause a loop among these two faulty DNS servers by injecting a single, IP-spoofed DNS error message. Once injected, the vulnerable servers continuously send DNS error messages back and forth, putting stress on both servers and any network link connecting them.



Such application-layer loop behaviors have now (i.e., in 2023) been discovered to exist in certain TFTP, DNS and NTP implementations. Furthermore, they always existed *by design* in at least six UDP-based legacy protocols (QOTD, Chargen, and Echo, Time, Daytime and Active Users), as also partially documented in the CERT advisory CA-1996-01 from 1996.

Application-level loops are different from known network-layer loops caused by misconfigured routers. Existing network-level defenses that detect loops at the network layer - like the Time-to-Live hop limit in IP - do not stop the attack. It requires orthogonal action items to mitigate the potential harm of application-layer loop DoS attacks.

Actions for the Operator Community

We are aware of servers with vulnerable implementations at least for TFTP (~23k hosts), DNS (~63k), NTP (~89k), Echo/RFC862 (~56k), Chargen/RFC864 (~22k) and QOTD/RFC865 (~21k). We have *not* tested other protocols and focused on those that were either widely deployed (TFTP, DNS, NTP) or easy to abuse (legacy).

Fixing all these servers at once seems not to be practical. Worse, while we know *some* affected products and software (see FAQ), we *cannot* yet attribute the vast number (~80%) of systems we found vulnerable to abuse. The vulnerabilities in NTP can likely be attributed to systems that use a pre-2010 *ntpd* version. The legacy protocols (Echo/Chargen/QOTD/Time/Daytime/Active Users) are vulnerable by design. For TFTP and DNS, we still require input from operators to learn about affected software setups.

Based on this, we suggest the following action items:

- **Notify ASNs about systems on their premise that can be abused for DoS attacks**. On December 20, 2023, Shadowserver delivered a one-time report to organizations that subscribe to their reports. The respective report page was **not** publicly announced. We suggest repeating this endeavor periodically (e.g., monthly for a year). We are currently working on automating the scans such that Shadowserver can periodically rescan for hosts vulnerable to application-layer loop DoS abuse.

- **Assist in the process of revealing which type of devices are affected.** We appreciate any help in coordinating the disclosure process with affected vendors / developers. We emphasize that we do our best to learn affected software products to help developers work on fixes. If you are aware of software that is not in the list of known vulnerable software or affected vendors, please get in touch.

Concrete Countermeasures

Preventive measures:
The below preventive measures shall help to reduce the attack landscape.

1. **Update or shut down vulnerable services.** Coordinate with affected parties to discuss if the affected hosts have to be reachable on the vulnerable services. If not, take the services offline. If yes, think of ways to introduce appropriate access control (e.g., firewalling).

2. **Restrict service access to clients with ephemeral source ports.** By design, loops are between two servers, where none of the servers uses client (ephemeral) source ports. The communication does *not* use UDP source ports typically chosen by clients (ports >= 1024[1]), but only the ports of the respective services (port range 0-1023). Vulnerable protocols can be protected by filtering non-ephemeral source ports towards the servers.

3. **Identify the software or product responsible for the behavior.** Report the vulnerability to the affected vendor with a reference to this document to give them a chance to fix the issue; alternatively, contact us.

[1] Note that some protocols may be found to be vulnerable in the future with dedicated server ports >= 1024. Then this check has to be adjusted accordingly.

Reactive measures:
In case of an attack, it is most effective to disrupt the loops. Any type of packet loss in the attack traffic terminates the loop and forces the attackers to reinitialize the loops. Packet loss thus effectively downgrades the application-layer loop attacks to amplification attacks. To this end, there are several options:

1. **QoS**: Give less preference to abused protocols to drop attack packets in case of network congestion. In particular, the UDP ports of the legacy protocols can be assigned low QoS priority. For the non-legacy protocols, this has to be decided on a case-by-case basis (69/TFTP, 53/DNS, 123/NTP).

2. **Rate limiting:** Networks can deploy rate limiting in case of loop attacks, which also terminates infinite loops.

3. **Attack detection:** Networks can detect loop patterns based on the fact that two servers communicate on their server ports, unless required by the protocol, which otherwise is a rather rare event[2]. For example, a loop within NTP would cause flows with source **and** destination port being UDP/123. Any combination of privileged (<1024) UDP ports for source and destination can be deemed suspicious in case of an attack event and be dropped or rate-limited as a last resort.

   Note: Given that the vectors to trigger loops are not fully explored yet, any type of payload matching to drop attack traffic will likely be incomplete.

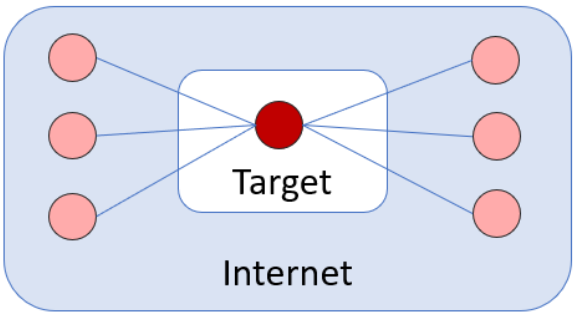
[2] In large networks this will likely cause collateral damage. Some legacy apps and configurations, most notably some NTP implementations and DNS configurations will source client requests from the same service port (123 and 53, respectively). Furthermore, there are the port address translation NATs that might use lower numbered ports in client requests. We hypothesize that these cases are relatively rare in comparison to the traffic this recommendation might mitigate. In addition, we want to highlight that legacy configurations such as DNS communications using 53 for both the source and destination port are discouraged and potentially dangerous. Well-known privileged port to well-known privileged port communications in general should not only be uncommon, but may lead to security problems such as the Kaminsky-style DNS cache poison exploit.

Security Implications

There are various DoS attack scenarios possible with loops.

Scenario A: DDoS a Loop Server

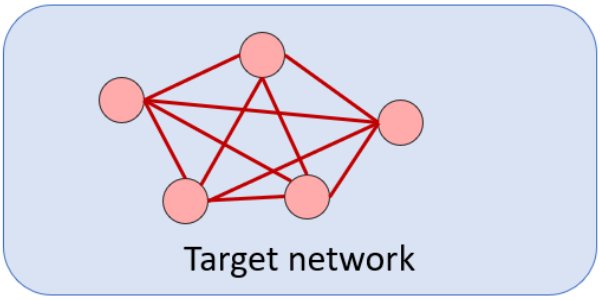
In the most simple scenario, an attacker overloads a loop server itself. To this end, they create many loops with other loop servers, all of which concentrate on a single target loop server; either exhausting its host bandwidth or computational resources. To defend against the attack, the loop server can be patched to escape loop patterns.



Scenario A: Attackers pair a target loop host with many others to overload its resources.

Scenario B: DDoS the Backbone of Network with Loop Hosts

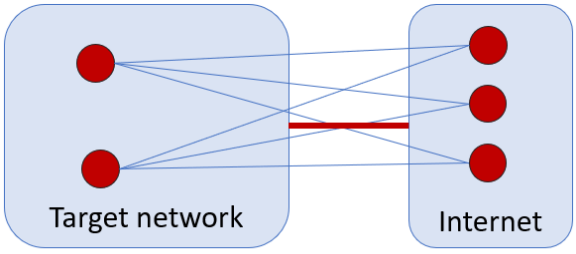
Attackers can also try to target backbones of networks that contain many loop hosts. If attackers pair these hosts with each other, they can create thousands to millions of loops within the target network. If networks deploy ingress filtering for IP-spoofed traffic, they are protected against such attacks from external hosts.



Scenario B: Attackers pair loop hosts *within* a network to overload its backbone.

Scenario C: DDoS Links (Uplink or On-Path Links)

Attackers could also try to pair loop servers in such a way to congest individual Internet links. In the most simple case, this could be a target network's uplink, as shown in the figure. To this end, attackers pair internal loop hosts with external ones, which puts stress on the target network's Internet uplink due to the loop traffic.



Scenario C: Attackers overload a target's network uplink by pairing loopy hosts *within* a network with external ones.

Note that the target link does not necessarily have to be an upstream link. Instead, attackers could also try to perform link-flood attacks (LFAs) against *any* Internet link that loop pairs cross. For this, attackers can pair loop hosts in such a way that all loops traverse a single (target) routing path, which would be congested this way. Normally, it is assumed that LFAs require botnets to launch them. From the attacker's perspective, loop hosts provide a cheap and thus attractive alternative.

Scenario D: Self-amplifying loops

We found rare cases in which loop servers would not send back a single response, but *multiple*. In these cases, it is possible to create self-amplifying loops that not only continue forever, but also intensify in their loop frequency. For the few samples that we looked at, we found that this behavior arises out of a combination of network-layer (routing) loops and application-layer loops. We believe that such attacks provide the most devastating attack type as it seems to be continuous even if defenses incur packet loss (unless they drop *all* attack traffic).

Acknowledgements

We would like to thank the operator community for their thankful suggestions and inputs to this document. In particular, we thank The Shadowserver Foundation, Barry Greene, Roland Dobbins, Rich Compton, Tim April, Dave De Coster, Piotr Kijewski, John Kristoff, Wim Biemolt and Damian Menscher for their rich contributions in various forms.

FAQ

Q. How bad are loops compared to other DoS attacks?

Application-layer loop DoS attacks range in the same league as amplification attacks. They also can cause large volumes of traffic. There are two major differences. First, attackers do not have to *continuously* send attack traffic due to the loop behavior unless defenses terminate loops to shut down the self-repetitive nature of the attack. Second, without a proper defense, the DoS attack will likely continue for a while; in fact, attackers have no control (!) to stop the attack once started.

Q. Would this attack be possible without IP spoofing?

No, attackers need a *single* spoofing-capable host to trigger loops. As such, it is important to keep up initiatives to filter spoofed traffic, such as BCP38. Note, though, that the loop hosts do *not* require/send spoofed traffic as part of the attack.

Q. Are cross-protocol loops possible?

Yes, we discovered a few combinations where loops exist across application-layer protocols. But the problem is not as widespread as same-protocol loops.

Q. Has this been exploited by actors already?

No, we don't think so. If you happen to find artifacts that make you believe this technique is used in the wild, get in touch with us.

Q. Are loops possible with more than two systems (i.e., rings instead of pairs)?

No, we don't think so. The nature of the attack requires a ping-pong behavior between a pair of servers. We're not aware of a situation in which more than two systems build a loop "ring".

Q. Are loop responses large enough to get fragmented at the network layer?

From the loop triggers known to us, we believe that the non-legacy protocols only create short responses that fit into a single IP fragment and thus do not fragment. We are, however, aware that some legacy protocols (e.g., Chargen) create larger responses that may fragment. Thus, with few exceptions, we believe that in most cases the UDP header is present in attack packets.

Q. Which software is affected?

We do not have a full list of affected software yet. The following preliminary list contains the confirmed cases known to us. If you find more software to be affected, please get in touch such that we can coordinate with the developers.

1. **TFTP:**
   atftpd not affected (uses random source port for responses)
   tftpd not affected (uses random source port for responses)
2. **NTP**:
   ntpd before version 4.2.4p8 and version 4.2.5 (CVE-2009-3563)
3. **DNS:**
   dproxy-nexgen
4. **Legacy protocols:**

At least the QOTD, Chargen, Echo, Time, Daytime and Active Users protocols are affected. These protocols were historically implemented as part of *inetd* in Linux and can be disabled in inetd.conf.

Q. Which vendors are affected?

We have an incomplete list of hardware products that are affected. Our hope is that most products import code from above vulnerable software projects. Patching the implementations centrally gives vendors an easy way to backport patches to their products. We are in contact with vendors to verify if their products are affected. Vulnerability scans suggest that the following vendors may be affected:

- Arris
- **Broadcom (2023-12-26)**
- **Brother (2024-02-06)**
- Cisco (e.g., out-of-life 2800/2970 routers; maintained products unaffected)
- D-Link
- **Honeywell (2024-01-03, CVE-2024-1309)**
- Hughes Network Systems
- **Microsoft (2024-02-19, in WDS)**
- **MikroTik (2024-01-09)**
- PLANET Technology Corporation
- TP-Link (e.g., out-of-life products TD-W8901G, TD-W8101G, R600VPN, WR740N, TD-W8960N)
- Zyxel (e.g., end-of-life ZyWALL; maintained products unaffected)

Note that this list is neither complete, nor does being listed here imply that vendors are for sure affected. We will mark cases confirmed by vendors **in bold**, add the dates of confirmation and the respective CVEs (if assigned any).

Q: "What if I believe I'm an affected vendor but have not been contacted yet?"

If your products expose any of the legacy services, you likely are affected. If your products expose NTP, DNS and TFTP, it depends on the concrete implementation. You can then check the version on the above table. If the software is not in the list, you can get in touch with us to clarify; please send your software information. We also have an incomplete collection of loop trigger messages in our code repository that can be used for testing; note, though, that negative tests does not imply that your product is not affected as there are loop triggers unknown to us.