December 2015

HTTP Evasions Explained - Part 9 - How To Fix Inspection

TL;DR

Since the first release of HTTP Evader the malware analysis in almost all of the tested firewalls, IPS, NGFW, UTM ... could be easily bypassed using uncommon or invalid responses from the web server. Some of these products provided fixes in the mean time and several other are still working on fixes. Some products might never be fully fixed.

But why did these products were affected at all? Even the devices which claimed to have Advanced Threat Protection (ATP)? This ninth part of the series about HTTP level evasions looks into the underlying problems of doing a reliable application level inspection, about the limitations of the different technologies used in firewalls and shows how it can be done better.

The previous article in this series was Part 8 - Borderline Robustness.

Theoretical Limitations: Passive vs. Active Inspection

There are two basic technologies in application level analysis: passive and active inspection. While the first one is usually faster and needs less resources, it is more limited in the protection it can achieve.

Passive inspection only checks if the data looks good or potentially malicious. In the last case it throws an alert or terminates the connection while in the first case it simply lets the data pass through unchanged. A typical example for passive analysis are classical Intrusion Detection Systems like the open source IDS Snort. But based on the behavior I've seen with the tested products I'm sure that several of the commercial products also do passive analysis only.

The main problem for passive inspection is the gray area where the data is neither fully good not does it look really bad. This happens when uncommon parts of the HTTP protocol or a slightly invalid protocol is used. Since browsers usually deal somehow with these kinds of problems it would be bad to block the data since this could result in broken websites. But it would not be a good idea to simply let all these data pass, because maybe the browser interprets these data different to the inspection device and thus malware slips through.

Thus passive inspection in firewalls faces the same problems as anti-virus which also only does passive analysis: either it blocks too much (false positive) and break web sites or it blocks to few (false negative) and potentially lets malware through. And the current solution is also the same as in anti-virus, in that it when in doubt it lets the data pass. Because otherwise customers would complain too often about broken web sites.

Active inspection instead not only looks at the data but can also modify it. Thus it can in theory sanitize the protocol to make sure that only a fully sane protocol is spoken with the client, i.e. no uncommon or invalid protocol parts. This way it can make sure that the data as seen by the browser is the same as seen by the firewall and that the analysis can not by bypassed due to different interpretation of the data. Typical examples of active inspection are application level gateways, i.e. proxies. Several of the tested products seem to employ this technology.

Another advantage of active inspection is that one can already modify the request so that so that the response can be better analyzed. Since many browsers today not only support gzip and deflate compression, but also sdch (Chrome, Opera), lzma (Opera) and brotli (new with Firefox 44) a device with passive inspection need to be able to understand all these compression methods. Devices with active inspection instead can restrict the compression methods by modifying the Accept-Encoding header before forwarding the request, and can then reject a response which uses an unsupported compression method because the response is invalid for the sent request.

Practical Limitations: Accuracy of Inspection

HTTP looks like a simple protocol, but it has parts where interpretation has been changed between protocol revisions or parts where the full implications are not obvious. It also borrows heavily from mail but differs in some aspects. This results in lots of different implementations which agree in the core functionality but differ in the fine details. And in the same way browsers adhere to the protocol only in the commonly used cases, firewalls differ in details too both from the browsers and from the specification.

Probably most of the HTTP implementations which can be found in the products are not designed with a protocol level attacker in mind. While I'm not able to look at the source code of these commercial products I had a look at the open source IDS Snort which is also often used in commercial products. And there they only pick the information the need and ignore everything else. This way they are not even aware that an attack is in process. Similar behavior can be seen when looking at the open source IDS Suricata and Bro.

How To Do It Better

I don't think a reliable inspection can be done with passive inspection only. Active inspection instead provides a way to do it right, by adhering in sending strictly to commonly used protocol features which are understood the same way by all HTTP implementations. Or to say it with the Robustness Principle: "Be conservative in what you send, be liberal in what you accept". Active inspection can do both parts of this principle and it should do it. Passive inspection only can liberally accept the data but is not able to be conservative in sending.

And as long the sending is done strict it actually does not matter if the interpretation of the incoming data is wrong. In the worst case it will break a web site. But it would never slip malware to the client just because client and firewall interpret the protocol in a different way.

Too Much Too Read, I Want To See Action

If you are curious how bad your browser or you firewall behave with uncommon or invalid responses from the web server head over to the the HTTP Evader test site. To read more about the details of specific bypasses go to the main description of HTTP Evader.