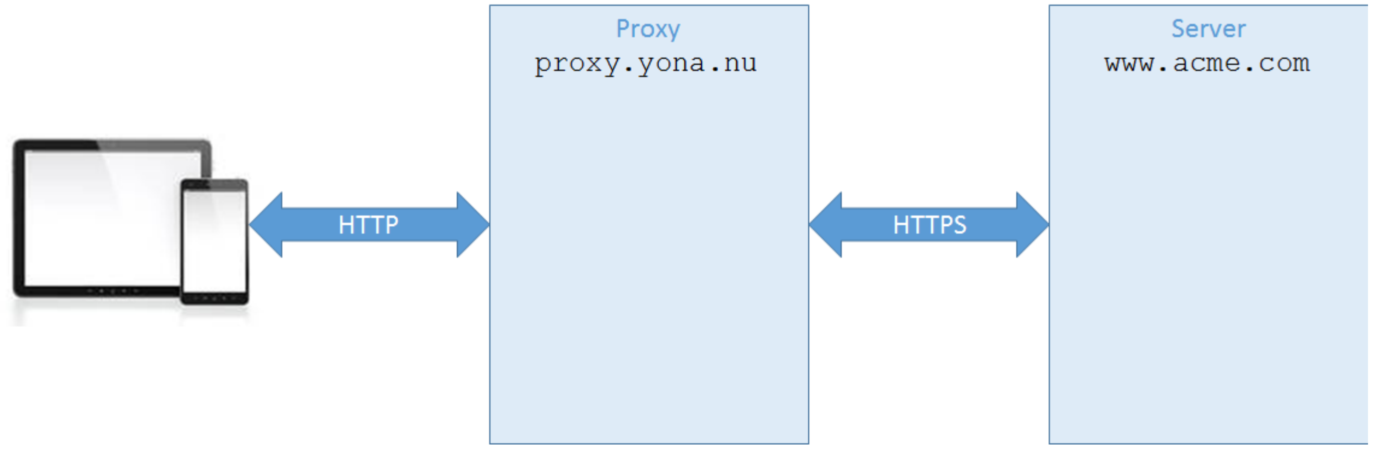
Intercepting HTTPS traffic

Introduction

Key functionality of the accountability is to screen all network traffic to verify that the internet is used in an accountable way. This is particularly challenging for HTTPS sites, because the conversation over this protocol is encrypted, with the objective to make interception (screening) of the data impossible. This page describes the few options that (still) exist to intercept HTTPS traffic.

SSL stripping

In this approach, all traffic from the device running the accountability app is routed through a proxy. That proxy creates a secured connection (HTTPS) between the proxy and the origin server, but establishes a regular HTTP connection between the device and the proxy. That way, all traffic can be screened.



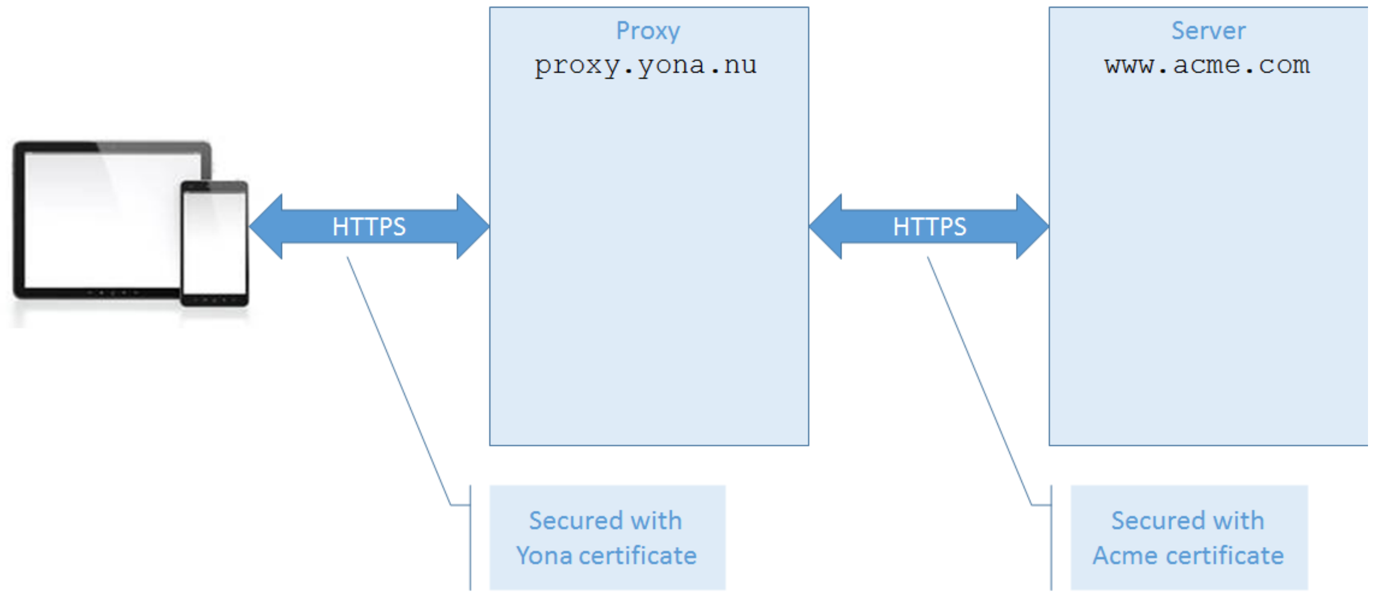
This approach is known as SSL stripping and it is used in a downgrade attack. The internet community has formulated an answer through HTTP Strict Transport Security (HSTS). Quote from Wikipedia:

> ***HTTP Strict Transport Security (HSTS)*** *is a web security policy mechanism which is necessary to protect secure HTTPS websi tes against downgrade attacks, and which greatly simplifies protection against cookie hijacking. It allows web servers to declare that web browsers (or other complying user agents) should only interact with it using secure HTTPS connections,[1] and never via the insecure HTTP protocol. HSTS is an IETF standards track protocol and is specified in RFC 6797.*

HSTS is not widely implemented yet, so SSL stripping could be used as a screening approach for certain sites for limited time, but it brings a severe risk to the users (traffic is not encoded) and, given the prevention measures being implemented, it is not worth the investment.

Man in the middle

The infrastructure for this approach is similar to the one for SSL stripping. The traffic is again routed through a proxy, but now the proxy sets up HTTPS at both sides. Between the proxy and the origin server, the connection is secured through the certificates of the origin server but the connection between the device and the proxy is secured with certificates of Yona. That allows us to decrypt the data and screen it.



This approach is known as a Man-in-the-middle attack. The internet community has formulated a generic answer to it by means of Public Key Pinning Extension for HTTP (RFC 7469). This was recently standardized (April 2015) and will replace the already existing nonstandard implementations for certificate pinning (aka SSL pinning), where applications and browsers come with a set of hard coded certificates linked to sites.

At first glance, this standard seems to prevent the development of a product that intercepts HTTPS traffic, but fortunately is not the case. The RFC defines a mechanism that allows users to load a certificate in their trust store, which can be used by a "man in the middle":

> *It is acceptable to allow Pin*
> *Validation to be disabled for some Hosts according to local policy.*
> *For example, a UA may disable Pin Validation for Pinned Hosts whose*
> *validated certificate chain terminates at a user-defined trust*
> *anchor, rather than a trust anchor built-in to the UA (or underlying*
> *platform).*

The objective of this mechanism is to enable debugging (e.g. Fiddler) and to allow content filtering in enterprise scenarios (see the explanation given here). The RFC describes it as an option, so browsers do not have to implement it, but the good news is that at least Chrome supports it (see the Chrome security FAQ). A detailed technical description of a man-in-the-middle proxy is available here on the Internet.

Device level interception

It is possible to intercept the traffic before the request is encrypted or after the response has been decrypted, but only for selected applications. A generic approach does not seem to be possible (Even the Parental Controls Internet content filter of Apple on Mac OS is not capable of filtering the content, see the HTTPS note here.). All browsers seem to have their own implementations: Firefox has provisions, Internet Explorer integrates with Windows Parental Controls and Chrome has supervised Users.

Conclusion

The man-in-the-middle approach is a viable solution. RFC 7469, designed to prevent a man-in-the-middle attack contains a specific provision to enable content filtering in a way Yona intents to provide.