Client-Initiated HTTP Covert Channels Using Relays

Gregory Daneault
Rochester Institute of Technology
Rochester, New York
Email: gjd6793@rit.edu

Daryl Johnson
Rochester Institute of Technology
Rochester, New York
Email: daryl.johnson@rit.edu

***Abstract*—This paper proposes a new covert channel utilizing open web relays. While the channel described is very straightforward, the addition of a trusted relay dramatically increases the anonymity and efficacy of this channel. Indirect, relayed communications disguise the actual endpoints of the communication making analysis, detection, and prevention more difficult.**

***Keywords*—Covert Channels, Information Hiding, Public Relays.**

I. Introduction

In general, communication over the internet is not designed to obscure the sender or receiver of the communication. In fact, the correct operation of many protocols depends on the explicit knowledge of both parties in order to successfully facilitate the communication. This makes masking the identities of those communicating very difficult. One common technique to obscure the fact that there is any communication whatsoever is the use of a covert channel [1]. Covert channels utilize a shared medium to send sensitive data by hiding it within legitimate non-sensitive data. As long as the encoding mechanism remains unknown, the fact that there is sensitive data being transmitted remains unknown.

A common protocol used to covertly transmit messages is the Hypertext Transfer Protocol (HTTP), as well as HTTPS, the more secure encrypted version. The presence of web traffic is nearly ubiquitous across all modern networks, making it an ideal vehicle for sending messages covertly. Section II will examine some currently existing covert channels operating over HTTP and identify some drawbacks to these methods. Section III will identify server-to-server relays as a potential solution to these drawbacks and provide important characteristics for those relays. Section IV will demonstrate the functionality of the relay in practice in a new HTTP covert channel. Finally Section V will discuss the successes of this method, as well as potential future enhancements.

II. Existing HTTP Covert Channels

There are a number of HTTP-based covert channels that have already been discovered. Several are explored in [2] and [3]. Many of the identified channels focus on covert server-to-server communication using unwitting client browsers as relays. They outline five potential mechanisms for relaying information between a source server and its intended destination:

1) Redirects
2) Cookies
3) Referer headers
4) HTML Elements
5) Active content

At their core, these mechanisms have three steps. First, a legitimate client connects to the source web server. That server then responds in a manner that enables one of the preceding techniques. The client browser then interprets the response and makes a request to the destination server. Figure 1 summarizes this sequence.

This methodology is effective at masking the fact that the source is communicating with the destination. It is also capable of bi-directional communication if each server implements both the sender role and receiver role. There are, however, several drawbacks to using these techniques. First and foremost, these channels have one critical point of failure. They are dependent on a source server with a sufficient amount of traffic to reliably communicate the desired messages. Each request can only contain a finite amount of data, so infrequently visited source servers are unable to transmit large pieces of data. Moreover, redirects to other untrusted websites are likely to garner suspicion from analysts. The basis of these channels is that the client will automatically visit the destination server on the source server's behalf. Host-based anti-virus, firewalls, and other mechanisms can potentially stop the operation of these channels.

There have also been covert channels established using direct client to server communications. Brown [3] describes a new timing-based channel wherein the client connects to the destination web server at regular intervals to transmit a message. An additional channel based on the browsing behavior of the client is also proposed [3]. These channels do not require the use of a source web server as the client is able to initiate communication with the destination whenever it is necessary. They do, however, suffer from the same limitation that repeated connections to an untrusted remote server may be treated as suspicious.

III. Client-initiated Server-to-Server Communication

There are many benefits to client-initiated covert channel. In client-initiated channels, communication may take place at whatever rate is desired. They eliminate the dependency on a third party visiting the website and becoming an unwitting agent. In addition, source-side initiation gives the source much greater insight into the reliability of the transmission. The sender is able to directly observe the status of each individual request. More than that, source-initiated storage channels can be adapted into bi-directional communication streams.

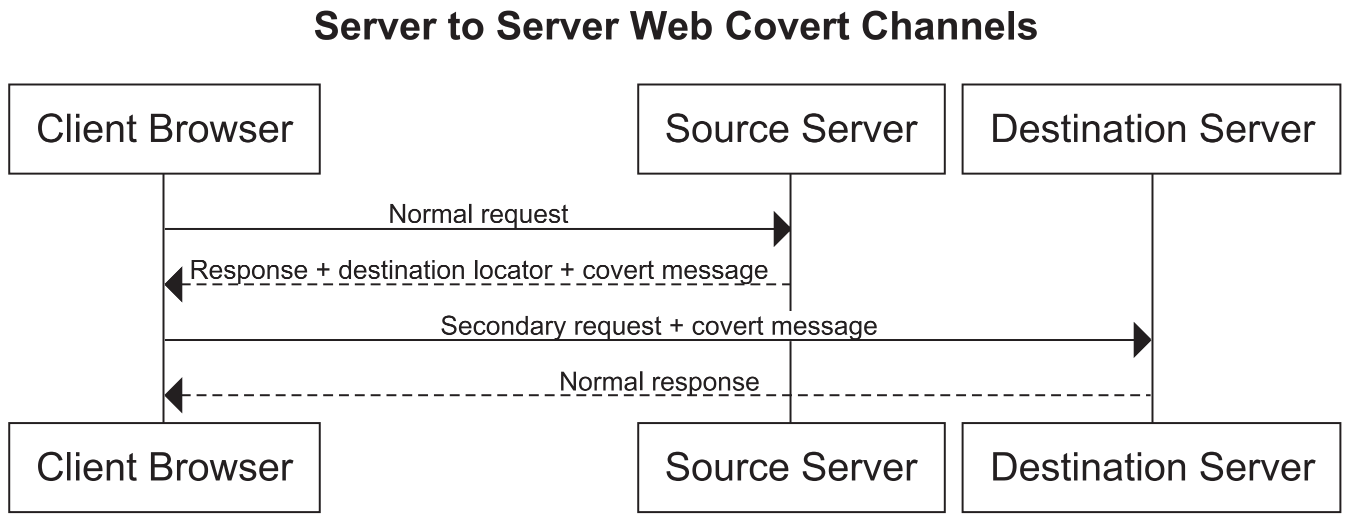

Fig. 1. Server to server covert channels using a client as a relay

These benefits come at the cost of making a connection to a destination-controlled server. Even well-disguised channels will appear suspicious when the source machine communicates with an untrusted destination. This is the primary advantage of the server-to-server channels; the source never directly communicates with the destination. Rather than connect directly to the untrusted destination, an innocent third party is manipulated into making the connection on the source's behalf.

It is possible, however, to combine the two strategies to have a client initiate server-to-server connections. This is made possible by the growing number of "open relays" that exist publicly on the internet. There are many services that automatically connect to external destinations on the behalf of their users in repeatable and predictable manners. Several of these will be explored in Section IV.

A. Relay Operation

The use of relays rather than direct connections allow for the channel traffic to be tunneled through legitimate services. Instead of the source connecting to the destination, the source is able to craft a communication to the relay, which will in turn deliver the intended message to the destination automatically. Figure 2 shows an example of this technique. The key difference between this communication flow and the flow of the channels described by [2] is the initiation. The relay operation is the server-to-server communication triggered by a client request to the innocent relay server. The existing HTTP covert channels perform a similar relay function. In those cases, however, the source server causes the innocent client to relay the covert message to the destination instead.

B. Choosing a Relay

There are many different ways that relays can be implemented. One of the simplest relays is the use of a proxy chain based on the HTTP `CONNECT` method. Public proxy servers which support the `CONNECT` method will automatically connect to the destination server on the behalf of the source. This connection occurs reliably and usually does not alter the content returned from the destination server before relaying it to the source client. Public proxies are not an ideal solution, however, as many are already blocked or monitored. Proxies can also modify the stream contents to inject advertisments and other payloads. Not all relays are appropriate for every situation. When selecting a relay to obscure the endpoints of a covert channel, there are a number of important factors to consider.

1) Relay Operation: First and foremost, the intended service must actually serve as a relay. This means that there must be some mechanism for automatically initiating connections to additional machines. Moreover, the specific destinations must be predictable, and the connections must in some way incorporate data from the original request. Without these two factors, a relay cannot be used to actually communicate in any meaningful way with the destination server.

2) Anonymity: There are many attributes of a potential relay that affect the anonymity of its use. Services that require user accounts or other similar authentication methods in order to access the relay functionality are clearly less desirable than ones without any such protections. Depending on the service there may be varying amounts of information required in order to procure an account. Generally, the less information required, the easier it is to create fake, anonymous accounts for use in the channel. If the operation of the covert channel is identified, it must be assumed that any accounts and services used by the channel will also be identified. Therefore, services that require the least amount of data and that perform the least amount of validation are ideal for protecting the identities of those involved.

Another factor to consider is the logging employed by the service. While it is unlikely that end users have full visibility into the data that is collected, it is still important to consider the effects that logs could have on the secrecy of the communication. Channels that are clearly abusing a legitimate feature in a way normal users do not may stand out from a logging perspective. On the other hand, channels that make only slight deviations from normal usage may be fine to log

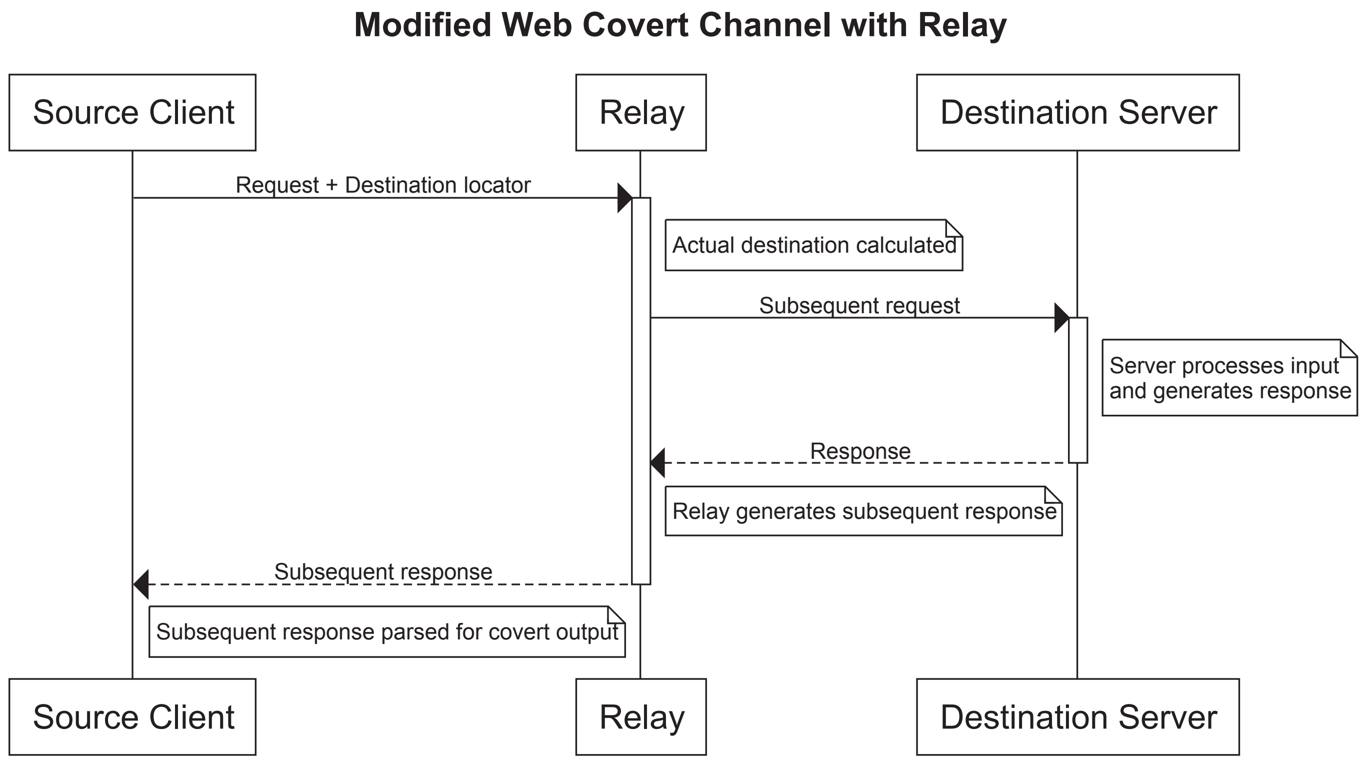
Modified Web Covert Channel with Relay
Source Client
Relay
Destination Server
Request + Destination locator
Actual destination calculated
Subsequent request
Server processes input and generates response
Response
Relay generates subsequent response
Subsequent response
Subsequent response parsed for covert output
Source Client
Relay
Destination Server

Fig. 2. Server to server communication initiated by the web client

without fear of discovery from the relay service operator.

3) Public Trust: An effective relay run by an untrusted or otherwise unknown third party adds little value or legitimacy to the covert channel. Therefore, it is important to select a relay that is commonly used and well-trusted. This is location-specific. The number one most visited website in the United States is `google.com` [4], but the number one website in China is `baidu.com` [5]. Observing traffic for Baidu in the United States would be suspicious, but it would not be unusual to see traffic for Google. Moreover, a sudden increase in communications to a relatively unknown site may be considered anomalous and receive additional scrutiny. The importance of the trust of the website increases with the rate of data transfer. Short bursts of communication to relatively unknown sites may in fact be a common occurrence. For example, clicking on a search result will likely take you to a much less frequently visited website containing the specific information requested. Sustained communication to that same unknown site, however, could be considered suspicious.

4) Predictable Output: In order to support bi-directional communication, the destination must be able to influence the content returned by the relay. The only opportunity the destination server has to communicate back to the source client is through its content embedded in the relay's responses. For example, if the only content returned by the relay after making the request is an automatically generated ID for the response, no meaningful data can be returned to the source. The extent to which the destination can control the output of the relay determines in part the bandwidth of the return channel. If only unidirectional communication is required, this attribute of the channel can be ignored.

5) Volume: Many potentially eligible relay services have rate-limiting or other limits in place to prevent abuse of their services. There are two facets in which these limits impose a maximum source-to-destination channel bandwidth. First, rate limiting dictates the maximum number of requests a user is allowed to make in a given period of time. Second, there is an upper limit to the amount of data that a service will actually relay to the intended destination. The product of these two factors yields the maximum source-to-destination throughput in a given time interval.

6) Encryption: A relay's use of encrypted HTTPS instead of plaintext HTTP is an important security mechanism. This provides the covert channel with an additional layer of security and obfuscation essentially for free. It is important to note that this encryption is not a perfect solution. There are many enterprise devices that provide man in the middle SSL decryption [6] which allow network administrators to see the secure conversations in plaintext. There may also be network administrators who either prohibit or provide additional scrutiny to SSL connections making the covert channel in fact stand out more than if it used traditional HTTP. Because of this, the specific situation in which the channel will be used must dictate the encryption requirements.

IV. Constructing a Covert Channel using Google Image Search as a Relay

The proposed covert channel is based on a simplified version of the redirect channel described in [2], [3]. The original redirect channel takes advantage of HTTP 302 redirects to communicate between two web servers. The source server responds to a third party client's request with a 302 redirect to the destination server. Built into this redirect is the covert message, added to the query string received by the second server. In its current form, this channel is initiated at the server side. The simplified version of this channel eliminates the source server and 302 redirects. Instead, the web client acts as the source and directly encodes all data in the request to the destination server. This operation is shown in figure 3.

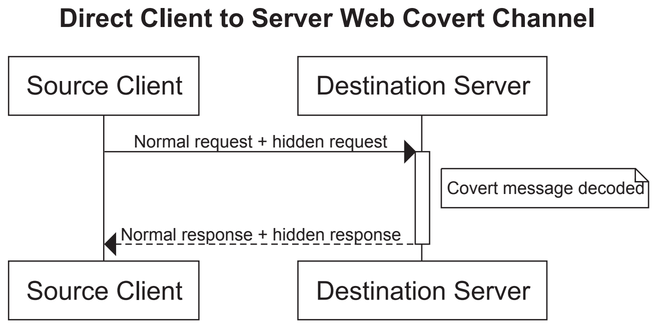
Direct Client to Server Web Covert Channel
Source Client
Destination Server
Normal request + hidden request
Covert message decoded
Normal response + hidden response
Source Client
Destination Server

Fig. 3. The HTTP covert channel before adding a relay

A. *Message Format*

The basis of this covert channel is the transmission of data to a destination server using only the query parameters of an HTTP GET request. Consider a request for `http://destination/normal/page.html?secret=password123`. This request sends the secret message "`password123`" to the destination server. The page returned by this request appears to be a completely normal document, specially crafted for the target environment in which the channel is operating. In this way, the source is able to communicate the sensitive data covertly.

The actual implementation of the channel uses a more complex encoding method for generating the query strings. The query string is broken into three parts:

1) The length of the message being sent.
2) The timestamp of when the message was sent.
3) The actual base64 encoded payload.

The length of the message being sent is also modulated to indicate the end of a transmission. The RFC places no upper limit on the maximum length of a URL [7], but in practice some browsers only support up to approximately $2,000$ characters [8]. In order to indicate the end of transmission, a length greater than this limit is supplied, indicating that the real length is $2,000$ less. For example, a length of 1024 would indicate that 1024 bytes of data are included in the request and there are future requests pending. A length of 3234 indicates that there are $3234 - 2000 = 1234$ bytes of data and this is the final transmission.

The inclusion of the timestamp of the message serves two purposes. First, it prevents caching. If for whatever reason the request with the covert data already exists in a cache, the request may be fulfilled without actually transmitting the message to the destination server. Second, it is used to ensure messages are recombined in the proper order.

These three pieces of information are concatenated together to produce the final query string. This is certainly not the only method for generating a suitable query string. The actual generated query should be specifically crafted for the target network to best blend in with normal traffic while still providing whatever information deemed necessary.

In its current state, the channel is able to covertly send data directly from the source client to the destination server. If the channel is using HTTPS it will be very difficult to determine that the covert communication is taking place. Even without the encryption, casual observation of the traffic would likely not reveal the true nature of the requests. There are many services that automatically generate reasonably long URLs with seemingly random characters. These connections, however, will still reveal that there is some communication taking place between the source and destination. This is where the use of the relay becomes important.

B. *Using Google Image Search as a Relay*

This channel can now be easily modified to use a relay. One such compatible relay service is Google Image Search. Google provides the ability to perform reverse image searches — searches based on images rather than text. A user can either upload a photo to search by, or provide the link to the photo. When a link is provided, Google will automatically download the image at that URL and perform an image search. If the image can be downloaded, the results page will contain some basic statistics about the image, as well as a best guess for the subject of the image, if one has been ascertained. This reverse image search feature meets all the relay criteria outlined in Section III-B.

1) Relay Operation: This is a clear example of an open web relay. End users have complete control of the URL that will be visited by a Google server. Searching for an image based on the URL reliably triggers a connection to the specified server. Additionally, there is no modification of the parameters passed between Google and the destination server; the initiating user has full control over the query string.

2) Anonymity: There is no requirement for an account to be created in order to perform an image search in this fashion. Google makes a record of previous searches available to authenticated users, implying global logging of search terms. That said, without associating the search activity to a registered user, it will likely be very difficult to trace the series of anonymous searches, especially if performed from disparate network addresses and with varying user agent strings.

3) Public Trust: In general, Google is a trusted website. There is certainly controversy regarding the amount and extent of personal data the company collects. That said, they are also synonymous with internet search and are currently the number one most visited site in the United States [4]. This is a large factor in public trust: the total usage of the service. Google already makes up a large portion of the web traffic for many networks [9], so the presence of additional search queries in network logs is not likely to stand out. In addition, all Google search products redirect to the same top level domain, `google.com`. This means that it is even more difficult to differentiate between normal web searches and image searches that may be relaying content over the covert channel.

4) Predictable Output: Google Image Search does provide semi-predictable output. All image searches include the dimensions of the relayed response. A shrunken thumbnail of the returned image is also shown. For sufficiently small images, the entire image may be displayed unaltered, allowing for encoding of response data in the image itself. If it was possible to identify the contents of the image, a textual form of the image may also be included and used to convey data. This interpretation, however, is subject to change at any time and would likely not prove reliable enough for serious use.

5) Volume: There is rate limiting in place to prevent abuse of Google's services. An observed upper bound of approximately 1.75 requests per second were permitted before requests began experiencing errors. In addition to rate limiting, there is a limit to the number of characters that a search URL can contain. At the time of this writing, the longest URL that can be entered is 1812 characters. This limit is placed on the entire URL, including the '`http://`' and the hostname of the server. During testing, each request contained an average of 1633 base64 encoded characters. This leads to an approximate maximum throughput of 2142 bytes per second.

6) Encryption: Google Image Search does support encrypted HTTPS traffic. In fact, visitors to the site are automatically redirected to the secure version of the search page whenever they begin searching over an insecure HTTP connection. Moreover, Google will download images over both HTTP and HTTPS. This means the destination server may use encryption as well.

C. Results

The modified covert channel is very effective at hiding the true flow of information. The prevalence of Google searches on modern networks make it an ideal carrier for the covert message. The use of SSL for the connection to Google makes inspecting the specific searches more difficult. More importantly, there are literally no observable connections between the source and the destination. These factors combined make it very difficult to prohibit the use of this channel.

This is a robust method of establishing a covert channel. HTTP and HTTPS are routable, TCP based protocols. This means that this covert channel will be able to automatically traverse NATs, as well as benefit from error checking and correction built into the TCP protocol. The relay operation itself is also fairly reliable. It is, of course, subject to the availability of Google's services and continued existence of this functionality. That said, Google strives for at least 99.99% uptime for their core services [10]. General availability will not pose an issue for most channel usage. Google does regularly update its featured offerings, so while this specific channel may be eliminated, there are likely other avenues that could be utilized within other Google offerings.

Detecting this channel after the addition of the relay is much more challenging than without it. Without the relay, the source browser directly connects to the destination server. Frequent connections to an external server of no repute is likely to garner additional attention during analysis of network traffic. After adding the relay, the challenge in detection increases two-fold. First, all traffic is now sent through Google, which is likely already a common occurrence on many networks. Simply identifying potentially malicious connection endpoints is no longer adequate. Second, traffic to Google is encrypted, so an SSL decrypting appliance will be necessary to fully identify the presence and means of operation of the channel. Once decrypted, the payloads will need to be very carefully analyzed to determine whether the search is a normal search or a search capable of sending a covert message.

These advantages are not a complete defense, however. Careful frequency analysis remains a viable attack vector for detecting this covert channel. It may be true that Google searches are a common occurrence on many networks, but a constant stream of searches performed at regular intervals will stand out from legitimate searches. Legitimate searches are performed at more random intervals, and only during certain hours of the day. If this channel were used overnight it would appear very out of place to see hundreds of requests when the rest of the network is essentially unused. In addition, depending on the exact encoding mechanism utilized, it may be possible to characterize all the covert searches based on a lack of entropy of length or content of the search string. In essence, searches that consistently match the maximum allowed length and use the same character set may be recognized as suspicious.

Preventing this channel is even more difficult than detecting it. One drastic option is to block all Google searching. This will seriously degrade the user experience for those using the network. Using deep packet inspection, it is theoretically possible to employ a more granular approach to block searches by image URL. This is not a good general solution however, as there are other Google features that provide similar functionality such as image insertion in Google Docs and link sharing on Google Plus.

Even a complete block of Google will be ineffective at stopping all variations of this covert channel as there are many other services which can be used in this manner. Facebook automatically visits URLs posted as status updates. Reddit automatically downloads images from shared links to generate thumbnails. Web scrapers automatically index links posted as Twitter status updates. These websites are among the top ten most visited websites in the United States [4] and could easily serve as effective relays. To prevent channels using this sort of relay, all outbound traffic will have to be inspected for URLs, and those URLs will need to be validated in some fashion.

V. Conclusions and Future Work

This model of using a relay works very well for simple channels which encode data in the URL. It provides greater anonymity and makes blocking the covert channel more difficult. That said, the individual application of the covert channel needs to be carefully considered before deciding to implement such a solution.

While this paper only explored one potential relay, there are many other publicly available services with similar functionality. The use of multiple such services in conjunction could help increase fault tolerance and further decrease the chances of detecting the covert communication. By spreading communication across multiple relays, statistical analysis becomes more difficult. This also can help increase throughput as the individual component services maintain their own usage limits.

The use of a domain generation algorithm could also further anonymize the endpoints of the communication. In its current state, the relay is sent a hard-coded destination URL, either an IP address or a domain name. If these remain static it is possible to perform analysis after the fact to determine the true destination of the communication. The use of a domain generation algorithm, in conjunction with a free DNS provider, would allow for ephemeral domain names to be used as relay targets. This prevents later analysis as the domains used would no longer be held by the same party as when the communication occurred.

References

[1] Butler W. Lampson. "A note on the confinement problem". In: *Communications of the ACM* 16.10 (1973), pp. 613–615. URL: http://dl.acm.org/citation.cfm?id=362389 (visited on 11/30/2015).

[2] Matthias Bauer. "New Covert Channels in HTTP: Adding Unwitting Web Browsers to Anonymity Sets". In: *Proceedings of the 2003 ACM Workshop on Privacy in the Electronic Society*. WPES '03. Washington, DC: ACM, 2003, pp. 72–78. ISBN: 1-58113-776-1. DOI: 10.1145/1005140.1005152.

[3] Erik Brown et al. "Covert channels in the HTTP network protocol: Channel characterization and detecting Man-in-the-Middle attacks". In: *Proc. 5th Intern. Conf. Information Warfare and Security. Ohio, USA*. 2010, pp. 56–65. (Visited on 11/30/2015).

[4] Alexa. *Top Sites in United States*. 2015. URL: http://www.alexa.com/topsites/countries/US (visited on 11/29/2015).

[5] Alexa. *Top Sites in China*. 2015. URL: http://www.alexa.com/topsites/countries/CN (visited on 11/29/2015).

[6] "SSL decryption: SonicWall delivers". English. In: *Network World* 29.9 (May 2012), p. 32.

[7] Fielding, et al. *Hypertext Transfer Protocol – HTTP/1.1*. RFC 2616. June 1999, p. 18. URL: https://www.ietf.org/rfc/rfc2616.txt.

[8] Microsoft. *Maximum URL length is 2,083 characters in Internet Explorer*. English. Dec. 2007. URL: https://support.microsoft.com/en-us/kb/208427 (visited on 12/03/2015).

[9] Simon Tabor. *Google's downtime caused a 40% drop in global traffic*. English. Aug. 2013. URL: https://engineering.gosquared.com/googles-downtime-40-drop-in-traffic (visited on 11/29/2015).

[10] Randall Stross. "99.999% Reliable? Don't Hold Your Breath". In: *The New York Times* (Jan. 2011). URL: http://www.nytimes.com/2011/01/09/business/09digi.html.