Detecting and blocking onion router traffic using deep packet inspection

Conference Paper · September 2016

DOI: 10.1109/ELECSYM.2016.7861018

3 authors, including:

Ferry Astika
Electronic Engineering Polytechnic Institute of Surabaya
39 PUBLICATIONS **290** CITATIONS
SEE PROFILE

Isbat Uzzin Nadhori
Electronic Engineering Polytechnic Institute of Surabaya
18 PUBLICATIONS **143** CITATIONS
SEE PROFILE

Detecting and Blocking Onion Router Traffic Using Deep Packet Inspection

Ferry Astika Saputra, Isbat Uzzin Nadhori, Balighani Fathul Barry
Department of Informatic and Computer Engineering
Politeknik Elektronika Negeri Surabaya
Surabaya, Indonesia
{ferryas,isbat}@pens.ac.id, balighani@it.student.pens.ac.id

***Abstract*—TOR (The Onion Router) has been a very popular anonymous proxy service. Since its first usage, TOR has been improved and become a very big network consisting more than 7000 relays. Beside used by journalist, activist, and writer as a tool for their freedom of speech, its highly anonymous service is also used by bot-nets, malware, distributed denial of service attacks, hidden services that sells illegal things, spams, and many more. This paper will explain about a TOR usage detection system by using deep packet inspection to extract and analyze its network traffic. The result of the analysis will be used as a parameter for a proxy server to block TOR traffic in the network. Using this network, detection, and blocking design, we can block the Onion Router Traffic originated from TOR browser.**

Keywords—TOR; anonymity; detection; deep packet inspection

I. INTRODUCTION

The internet has been developed, evolved, and also used by so many positive activities, for example, by innovating our way in information sharing and education, helps us in the field of health and medical, or some other innovation that cannot be listed one by one. But in other side, internet can also be misused to do negative or illegal activities, for example, illegal hacking and attacks, spreading malware or scam, pornography sites, narcotics and other illegal transaction, and many more.

It is a common sense that when people are going to do these activities, they will think about their privacy first, about their identity, whether they are doing these activities anonymously or not, whether someone knows what they are doing or not. Because of these reason, they will seek some technique or method to gain more anonymity in the internet. Anonymity cannot be obtained without using some special method or technique. There are many methods that can be used to obtain various levels of anonymity in the internet, one of which is by using TOR (The Onion Router).

The Onion Router, also known as TOR, is a free software that could provide anonymity by using onion routing method. Onion routing is a technique to redirect the user's internet traffic to certain servers, and hide their sent and received data by wrapping them with some encryption layers. Those method are used in order to anonymize the sender and the receiver identity and data.

TOR is basically created to help the internet users in their freedom of speech and privacy problems, including illegal traffic tapping, network surveillance that threatens their personal identity and privacy. TOR can also be used for securing the user's business or private activities, hiding their identity when executing freedom of speech, and some more good reasons, but just like the internet itself, TOR can also be misused to do some negative activities.

In Indonesia, various sites that have negative contents like pornography, online betting, malwares, are blocked by the government using simple DNS and IP address filtering. By using TOR, people can easily bypassing those limitation enforced by the government or law.

Besides bypassing that blocking system, TOR can also be used for other negative activities, for example, using its anonym service to spread malwares and conduct illegal hacking anonymously.

TOR also has some special things that make it different with other anonymous service provider. Using TOR, we can surf a hidden part of the internet that usually called the onion websites. The onion websites are hidden websites which addresses ended with .onion. These websites are only accessible by using TOR service. In this hidden part, we can find so many example of negative websites, such as online illegal drugs and narcotic store, illegal gun store, hire an assassin service. All the information of these sites' provider and users are flowing over TOR networks, so, all the information are anonymous and really difficult to be traced.

Just like the internet, the number of TOR users is also increasing year to year, according to the official data from the beginning of 2012 until the end of 2015 published in their website, the average number of TOR users reached more than 3.5 million users per day, with the highest number of more than 11 million. This means that TOR is now commonly known and used by the internet users.

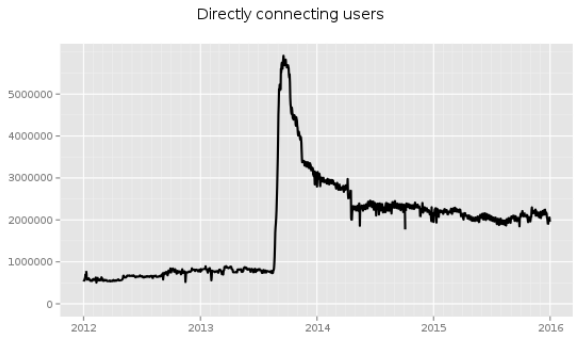

Figure 1: TOR Daily User Graph 2012 – 2015 (Source: https://metrics.torproject.org).

In this research, we will analyze the information of internet traffic in a network that consist of TOR user and ordinary internet users. The TOR users are using TOR with the TOR Browser Bundle software that provided free by Tor Project in its official website, and the detection will be done by using DPI (Deep Packet Inspection) method.

DPI is a packet filtering method that extract the inside of the packet, including header and its payload, and then examines them. The examination method is varies according to its function. It can be used to examine viruses, intrusions, spams, etc. and perform some action, or just log them to collect their information data.

DPI has been used by many users, from a small community, corporations to governments. In the corporation level, DPI usually used for preventing a virus or worm spreading through network, or preventing network attacks. In government level, DPI usually used for law enforcement, for example to block some website access, filter bad traffic like DDoS, spam, etc. and some other cases.

DPI has been implemented by many countries as their method for traffic filtering, limitation, and law enforcement. The most famous is China's, it's called The Great Firewall of China. China uses DPI to enforce government blocking system. The websites they block are including Google, Wikipedia, and some VPN websites.

There are many software that can be used to implement DPI. In this research, Bro-IDS will be used. Bro-IDS is an event-based network security monitoring platform that could trigger some action according to events that happening in the network.

II. Related Works

Onion routing method began developed in 1995, and then the architecture of onion routing was published in 1996. The first generation of onion routing was designed to hide the routing information and minimalize the potential of traffic analysis on a network. The first architecture of onion routing was a virtual circuit that established by the user all the way to the destination. This method can be used in two-way direction, and can be used for any protocol that can adapt to use proxy.

Over time, TOR improved by many developers, and in 2004, second generation onion router (TOR) architecture was introduced. The research was done by Roger Dingledine *et.al*[1]. The architecture and work flow of this second generation architecture was explained in detail in their research publication.

Research in the field of TOR user detection has also been done by Aanders Olaus Graneud from Gjøvik Universty College **Error! Reference source not found.**. The research he did was using the TLS protocol that used by TOR. He analyzed the differences and anomalies of TOR's TLS with other ordinary TLS connection.

Anders used some methodology to identify and differentiate the implementation of TLS in TOR's TLS connection with other common TLS connection. From this research, he found some anomaly in the TOR's TLS connection, for example, the port number used by TOR was 9001, but normal TLS usually uses 443. The combination of cipher suites that offered by TOR is always consist of a same combination.

But, along with the development of TOR, some anomaly and characteristics found by Anders is not valid anymore and cannot be used as a reference.

III. How TOR Works

Now TOR has more than 6000 active relays that provided free and also helped by volunteers from all over the world. All these relays are registered in some special and trusted servers, these servers are called directory servers.

TOR relays communicate using fixed-size cells. Those cells size are 512 bytes, consist of header and payload. These cells classified into 2 categories, those are control cell and relay cell. All communication that passed on the TOR traffic are executed using TLS protocol to maintain its security and secrecy.

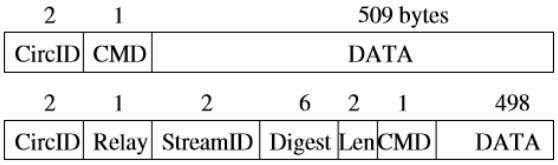

Figure 2: TOR Control Cell and Relay Cell Structure

The header of TOR cell consist of a circuit identifier (CircID) to define its cell's circuit, because in one TLS connection, there can be more than one multiplexed circuit), and a command (CMD) to bring a command to the destination. This command is also the one that classified either the cell is a control or a relay cell. Control cell commands are: padding, create (to create a connection), created (indicates connection created), and destroy (destroy a circuit).

Relay cell has a similar structure, but it has a relay header placed before CMD. It contains stream identifier (StreamID), end-to-end checksum for integrity checking, and the length of the relay payload. The relay commands are : relay data, relay begin, relay end, relay teardown, relay connected, relay extend, relay extended, relay truncate, relay truncated, relay sendme, and relay drop.

TOR works by first downloading the list of the relays provided in their respected directory servers. Then, after they downloaded all the relays information, TOR client will choose some random relays to be a circuit.

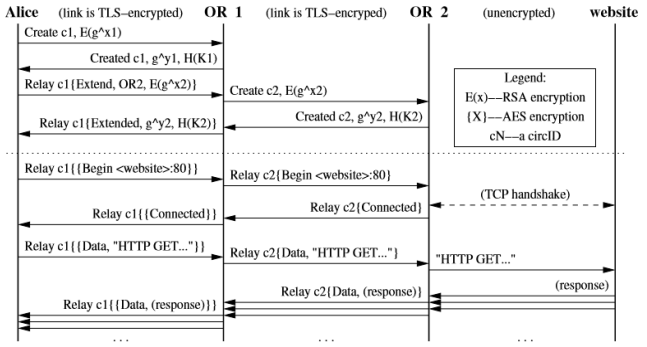

Figure 3: TOR Circuit Making Process. "OR" is abbreviation of Onion Routing

After TOR chose some relays to be a circuit, TOR client then send a control cell with a "create" command to the first relay in the circuit, then, after the first relay received this control cell, it will send a reply in the form of a control cell too, to the TOR client with "created" message or command to indicate that the connection to that relay has been established. Along with this reply, the first relay will send its public key that will be used for their communication encryption.

If there is more or another next relay in the circuit, then TOR client will send a relay cell with a "relay extend" command and the address destination of the next relay to the first relay, then that first relay will receive it, and responded by sending a control cell with "create" command to the second relay. After the second relay received the create command, it will respond by sending a "created" command to the first relay along with the second public key for TOR client to indicate that the connection has been created. The first relay will receive it and send an "extended" relay cell along with the second public key to TOR client. This process will be repeated until the last relay in the circuit. The last relay on the circuit will then be the exit node.

Then when opening a TCP stream, TOR client on the user's computer will send a relay cell with a begin command inside its CMD, then it will be translated by the relay and will be sent to next relay following the circuit until the last relay (the exit node). When got a begin command, the exit node will make a three way handshake to the user's destination address. After the handshake successfully executed, the exit node will send a relay cell with a connected command to indicate that the connection is established.

After it's all done, the user can finally send the converted data as a relay cell through the established circuit to the destination. There will go the data transfer or request – response activities through the circuit.

After all the request – response activities are done, the connection for the TCP stream will be closed. This process is done by sending a relay cell with a close command through the circuit until the exit node. The exit node will then close the connection between it and the destination.

IV. System Design and Implementation

The network topology that will be used in this research is described in figure 4.

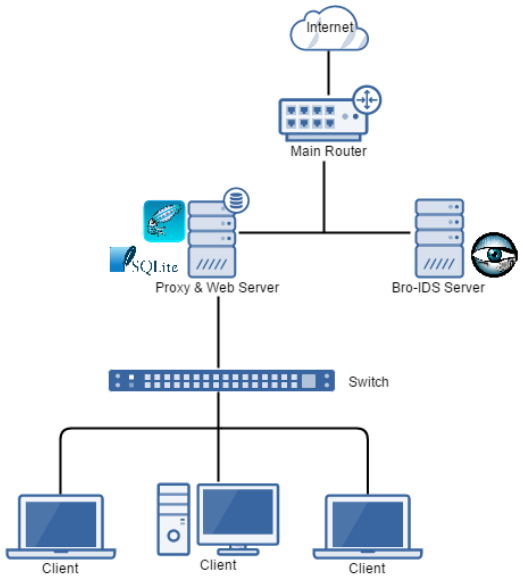

Figure 4: Network Design

As seen on the figure 4, this system needs an Internet connection, one proxy server (in this research, we will use Squid proxy), one IDS server (using Bro-IDS platform), and client (the switch usage is optional, we can utilize it if more than one client is exist).

A. Main Router

Main router is the router that directly connected to the internet. In this research, MikroTik RB1100 X2AH is used as the main router and it is connected to the internet using a public IP.

B. Proxy Server

Proxy server is used for internet sharing and give access to clients in the network. We use Squid3 as the proxyserver.

C. Bro-IDS Server

Bro-IDS[3] is a tool that can be used to analyze a network traffic in real-time. Bro-IDS uses deep packet inspection method to look inside the packet's header and payload. Bro-IDS's scripts and actions are event-based, meaning that it can triggered when there is some events happening in the network traffic, for example, we can set Bro-IDS to trigger or run some actions or scripts when there is a new TCP connection established.

In this research, Bro-IDS will be the tool to analyze this network's traffic, and try to differentiate TOR user traffic in the network. The traffic that went to Bro-IDS server is actually a

mirror from client-proxy server communication, using MikroTik's port mirroring method.

D. Client

What we referred as a client here is not the TOR client, but all the internet users in the network. All these clients are connected and can access the internet using proxy server. Among these users, there will be some TOR users who use TOR browser to surf the internet. These clients are the clients that we will try to detect.

E. Flowchart

By using the tools that have been mentioned before, in this research we use an algorithm that can be illustrated in figure 5.

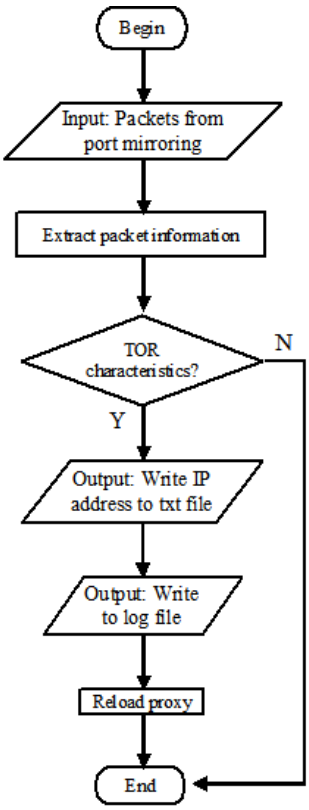

Figure 5: Blocking System Flowchart

The input that will be analyzed by Bro-IDS is originated from client – proxy traffic, and mirrored to Bro-IDS server by the router. Then all the information from those packets will be extracted by Bro-IDS, and then analyzed to decide whether it has a TOR packet characteristics or not.

If it does not have any TOR characteristic, then there is no action that will be triggered. But if there is a packet that has TOR characteristics, then some useful information from that packet will be extracted and written into Bro-IDS log file. Bro-IDS will also trigger 2 action after writing into log file, the first is to write the destination IP address of that packet into a file which will be used as the proxy's blocking reference. The second action is to reload the proxy so the newly added IP address will be blocked.

V. TOR Characteristics

Bro-IDS can analyze network traffic in real-time, or use a TCPdump file from a captured network traffic to be analyzed offline. In the experiments that have been done in this research, the traffic that analyzed is a TCPdump file obtained by using wireshark. The TCPdump file contains connections traffic of TOR browser and a regular browser users.

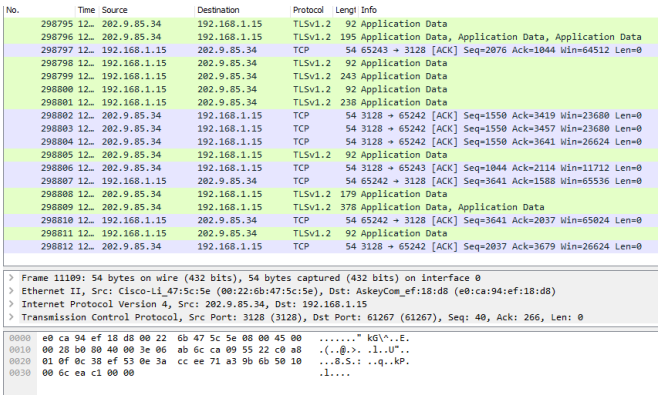

Figure 6: Wireshark's TCPdump File

As mentioned before, all TOR network traffic is encrypted using TLS. Because of this, we cannot see inside the packet when TLS encryption is already established. To see the anomalies and characteristics of TOR traffic, we can only analyze its TLS connection establishment process.

There are two processes that will be analyzed. The first is the process of connection establishment or usually called three-way handshake between TOR users / clients and TOR network, and the second is the process of making TLS protocol secure session or usually called TLS handshake process.

A. Connection Establishment Process

Connection establishment process is the first step that will be processed when a client wants to make a connection with a server. Normally, this process will be done in 3 steps, usually called three-way handshake.

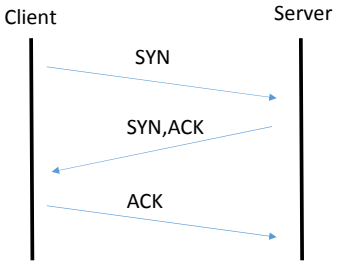

Figure 7: Three-way Handshake in TCP Connection Establishment Process

In a three-way handshake, the client will initiate the steps by sending a TCP packet with SYN flag. After that, the server which get the packet will respond by sending a TCP packet with SYN,ACK flag. And the last, the client sends a TCP packet with ACK flag to the server. Once this process is

completed, the connection has been made between the client and the server, and they can start data transfer.

When the header of TCP packets in TOR connection establishment and normal connection establishment compared, no anomalies were seen. TOR do a three-way handshake with SYN, SYN,ACK and ACK flags like other normal connections.

B. *TLS Session Establishment*

TOR traffic uses TLS protocol as its encryption method. TLS is a replacement for SSL. Until now, TLS has evolved to version 1.2.

In general, websites that use TLS (HTTPS) have implemented TLS version 1.2, as well as TOR. TLS also have TLS handshake process which is a process for making a TLS session. The process can be illustrated in figure 8:

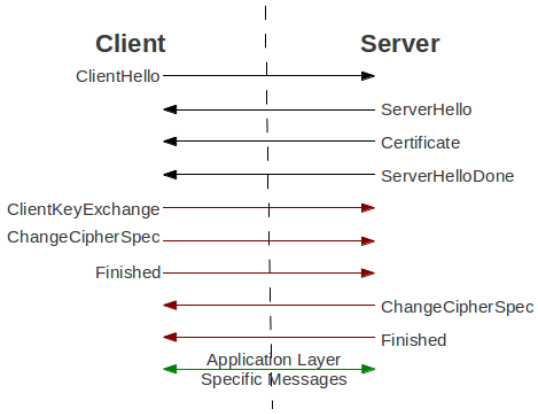

Figure 8: TLS Handshake Process

1) *Client Hello*

Client Hello (*ClientHello*) is the first message sent that sebt by the client to start or initiate a TLS handshake. The structure of the ClientHello messages are described below:

- Handshake Type: Type of the handshake message that sent.
- Length: The length of handshake message.
- Version: TLS version which the client wants / offers to use.
- Random: An information consist of time and 28 random bytes.
- Session ID: ID of the last session that client can and wants to use, so they do not have to make a new session.
- Cipher Suites: List of encryption algorithm that offered by the client.
- Compression Method: Compression method that can be used by the client to decrease bandwidth use.
- Extension: Consist of some extension or additional information that sent for the server. TLS protocol has so many extension that can be used, one of the important extensions is “server_name” that used for giving a name information for the server, which hostname is visited by the client. By using this extension, server with 1 IP can have more than 1 hostname.

After comparing and analyzing the *ClientHello* message from the TOR Browser and ordinary browser, there are some characteristics or differences found.

The first characteristic is the combination of cipher suites. The combination of cipher suites offered by the browser / client is decided from a combination of the most known by the browser, thus, a combination of cipher suites offered by TOR Browser is always the same. The second is in its "server_name" extension. Server name designated by TOR has a format that is always the same, that is "www.<randomstring>.com" or "www.<randomstring>.net".

2) *Server Hello, Certificate, Server Hello Done*

After getting the *ClientHello*, the server will provide answers by sending a S*erverHello* message. Like the one in the TLS handshake illustration, the *ServerHello* message delivery is also accompanied by a certificate and *ServerHelloDone* delivery. The structure of each message is as follows:

- *ServerHello* :
  - Handshake Type: Type of the handshake message that sent.
  - Length: The length of handshake message.
  - Version: TLS version which the server accepts to use.
  - Random: An information consist of time and 28 random bytes.
  - Session ID: ID of the last session that client can and wants to use, so they do not have to make a new session.
  - Cipher Suite: Encryption algorithm that accepted / chosen by the server, chosen from client’s cipher suites offer.
  - Compression Method: Compression method that can be used.
  - Extension: Extension or additional information that delivered by server along with the message.
- *Certificate* :
  - Version: Version of the certificate that used.
  - Serial Number: Used to identify the certificate.
  - Signature Algorithm: Algorithm that used to make the signature.
  - Issuer: Name / information about who issued and verified the information in the certificate.
  - Validity: Date range of the certificate’s valifity.
  - Subject: Subject of the destination server.
  - Subject Public Key Info: Consist of public key and algorithm that used in that public key encryption.
  - Extension: Extensions or additional information that provided.
- *ServerHelloDone*: This message indicates that the server hello is done, so the process can be continued to next phase.

In packets sent by TOR in this process, discovered another characteristic of TOR existed in the certificate. In the subject

and issuer information, TOR provides a string with the same format as the "server_name" information in the client hello, which is "commonName=www.<randomstring>.com" or "commonName=www.<randomstring>.net".

3) Client Key Exchange, Change Cipher Spec (Client), Finished (Client)

Client makes master key using his and the server's random key, and then encrypted with the server's public key with algorithm that has been previously approved in the negotiation process. In this process, no anomalies were seen when compared to a regular connection.

4) Change Cipher Spec (Server), Finished (Server)

This process is the same as the previously performed process by the client, indicating that traffic from the server will use the appropriate encryption algorithm that has been negotiated. No anomalies were seen in this process.

VI. Detecting and Blocking

TOR traffic characteristics obtained from the analysis described in Chapter V is used as parameters to detect TOR traffic. The detection is performed by using Bro-IDS tool.

The work flow of the script that used is when Bro-IDS detects the presence of TLS / SSL Establishment, Bro-IDS will perform a check on the TLS certificate, if it has the characteristics of TOR, Bro-IDS will write down some information from the connection to torconn.log file in the logs folder belonging to Bro-IDS.

```
#separator \x09
#set_separator     ,
#empty_field       (empty)
#unset_field       -
#path     torconn
#open     2016-05-05-02-23-39
#fields   tor_ip    source_ip ts
#types    string    addr      time
178.63.140.246     192.168.1.254
          1462429419.444899
93.115.84.143      192.168.1.254
          1462429419.664135
```

Figure 9: Log File Generated By Bro-IDS

In addition to writing them on the log file, Bro-IDS also write down the destination IP address of the detected TOR packet into a file that will be used by proxy to block.

```
178.63.140.246
93.115.84.143
94.242.228.174
90.155.23.218
213.246.56.62
...
```

Figure 10: Squid Proxy Reference File

VII. Conclusion

TOR (The Onion Router) is a very famous anonymous service. TOR works by passing network traffic into multiple relays it provided, before leading to the actual destination. Additionally, TOR disguise its connection so it looks like an ordinary HTTPS connection. Its communications that implement TLS v1.2 protocol also makes it better at providing anonymity for its users. Although the contents of its data traffic is encrypted, TOR traffic still could be recognized by analyzing at its TLS handshake process and observe the certificate that exist on that process, especially in the subject and issuer field of the certificate. These characteristics that exist in TOR can be used to detect if there is a TOR connection.

Using the network design, detection and blocking system explained in this research, we can detect and block the Onion Router traffic that originated from TOR browser.

References

[1] Roger Dingledine, Nick Mathewson, and Paul Syverson. 2004. Tor: the second-generation onion router. In Proceedings of the 13th conference on USENIX Security Symposium - Volume 13 (SSYM'04), Vol. 13. USENIX Association, Berkeley, CA, USA, 21-21.

[2] Anders Olaus Granerud. *Identifying* TLS *Abnormalities in* TOR.,MA Thesis of U. Gjøvik University College, 2010.

[3] Vern Paxson. 1999. Bro: a system for detecting network intruders in real-time. Comput. Netw. 31, 23-24 (December 1999), 2435-2463. DOI=http://dx.doi.org/10.1016/S1389-1286(99)00112-7.

[4] Jeff Moser, *The First Few Milliseconds of an* HTTPS *Connection*.,2009, http://www.moserware.com/2009/06/first-few-milliseconds-of-https.html

[5] Marek. *Dissecting SSL handshake*, 2012, https://idea.popcount.org/2012-06-16-dissecting-ssl-handshake/

[6] Wikipedia. *Transport Layer Security*, https://en.wikipedia.org/wiki/Transport_Layer_Security