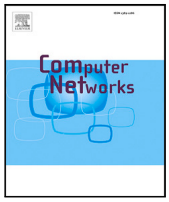
Computer Networks 250 (2024) 110570

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet



HoDiNT: Distributed architecture for collection and analysis of Internet Background Radiation

Rodolfo García-Peñas *, Rafael A. Rodríguez-Gómez, Gabriel Maciá-Fernández

Network Engineering & Security Group (NESG), Universidad de Granada, C/ Periodista Daniel Saucedo Aranda s/n, 18014, Granada, Spain

ARTICLE INFO

Keywords:
IBR (Internet Background Radiation)
IBN (Internet Background Noise)
Distributed architecture
Network telescope
Network security

ABSTRACT

Attacks on the Internet are constant, with different typologies and processes. The initial stages usually involve an enumeration of targets and available services, generating what is known as Internet Background Radiation (IBR). Capturing and analysing this traffic has proven to be crucial for the early identification and detection of attacks.

Commonly used architectures for the acquisition of background traffic are based on "black holes", which are systems that collect this traffic by advertising large blocks of unused IP addresses to the Internet, identifying the traffic received as IBR. These systems have a number of inherent drawbacks, such as the requirement to process large volumes of data, and deal with the existence of a large amount of repetitive data, the fact that they are easily identifiable by the IP addresses used and, finally, that they are expensive to maintain.

With the aim of improving the above undesired characteristics, this paper proposes "HoDiNT" (*HOme DIstributed Network Telescope*), a distributed architecture for the acquisition of Internet Background Radiation. HoDiNT is implemented with low-cost advanced acquisition techniques and without the need to use specific IP address ranges, making it easier to hide the sensors. An initial scan of the traffic received for one month is performed on the probes deployed, and a subsequent analysis is performed on the collected data to draw conclusions.

1. Introduction

On the Internet, the majority of traffic is legitimate, connecting users and systems to each other and enabling various forms of communication. In these communications, the sender transmits information to the receiver, who then waits to receive it and, if necessary, respond.

In addition to legitimate communications, there is also unsolicited traffic from senders to destinations that either do not exist or do not expect such communications. These destinations may be IP addresses that are not in use or IP addresses that are in use but do not provide the services requested by the sender. This traffic is known as Internet Background Noise (IBN) or Internet Background Radiation (IBR) [1].

Despite the fact that IBR traffic might be generated by configuration errors in equipment connected to the Internet, or sent by network intelligence companies or research organisations, most of this traffic is due to scans and enumerations of devices and services, which are common initial stages of Internet attacks, or direct attacks [1]. The acquisition of IBR traffic has therefore proven to be essential for the early identification and detection of attacks and the discovery of new attack typologies.

Traditional IBR detection systems (known as *network telescopes*) use a large number of IP addresses in order to have a larger traffic capture area to receive the IBR traffic. These addresses are usually aggregated in a single network of very large size (millions of addresses). The network telescopes, like other networks on the Internet, advertise their addresses using BGP (Border Gateway Protocol), to attract all traffic to a single point and thus capture large volumes of IBR traffic. However, because they are large blocks of unused networks, they can be easily identifiable by attackers. This could prevent them from generating network traffic directed towards network telescopes, and therefore limit the reception and capture of such traffic. In addition, because the IP addresses are usually contiguous, the scans received by these network telescopes may contain a lot of repetitive traffic [2] that could be identified with a smaller set of destination addresses, resulting in an excess of bandwidth and storage consumption [3].

In addition, the infrastructure used by the network telescopes is costly, both in terms of the high-performance equipment needed to handle the traffic, and in terms of bandwidth and energy consumption. High availability with redundant systems increases these costs even more.

A final fact about traditional IBR telescopes is that they are passive and do not transmit traffic. They only receive traffic and log it. This is

* Correspondence to: Universidad de Granada, C/ Periodista Daniel Saucedo Aranda s/n, CP. 18014, Granada, Spain.
E-mail address: rodgar@correo.ugr.es (R. García-Peñas).

https://doi.org/10.1016/j.comnet.2024.110570
Received 9 February 2024; Received in revised form 26 April 2024; Accepted 3 June 2024
Available online 8 June 2024

why they are known as "black holes". As shown in some articles [4], there are two-phase attacks where it is necessary to interact with the attacker to get further information from him. If the attacked host does not respond, the attacker assumes that the host is down or the port is closed or filtered and does not continue with the attack. In two-phase attacks, if the attacked host responds, then the attacker continues sending more information (new connections, payloads,...). As a result, classical network telescopes are not suitable for obtaining useful information from such attacks.

One of the most important examples of a classical network telescope is CAIDA's network [5], which advertises more than 12.5 million IP addresses. In recent years, other network telescopes have been developed that use both unused and used addresses as IBR traffic sensors [6], making it easier to hide these sensors. There are also telescopes that actively respond to IBR traffic to capture those attacks that require sensor responses [4].

The aim of this paper is to propose HoDiNT (HOme DIstributed Network Telescope), a distributed architecture of probes for the detection and capture of Internet Background Radiation. The characteristics imposed on this architecture are the following: Low cost, easy to install, support two-phase attacks and being not easily identifiable. It is based on the installation of probes in home Internet access routers that are able to report traffic in a distributed manner to collection devices, where this traffic can be analysed.

The structure of this article is as follows. Section 2 presents the background and state of the art of IBR traffic and network telescopes, showing in chronological order the most relevant studies on the topic and the different methods used to date for IBR traffic collection. Section 3 presents HoDiNT, showing its advantages over classical network telescopes and describing both its architecture and the different parts that make up the distributed network telescope. Section 4 presents the analysis carried out to demonstrate the viability of HoDiNT, detailing the types and volumes of traffic collected over a four-week period, making comparisons between sensors supporting two-phase attacks and including, as examples, some attacks collected by the Network Telescope. Finally, Section 5 draws the conclusions of the study.

2. Related work

The term IBR was defined in 2004 by Pang [7], who described it and outlined its characteristics. The term "Net's Background Radiation" had been referred to by columnist Andrew Orlowski a few months earlier in a newspaper article [8], and had been discussed several times before [9], but without the concept being explicitly named.

At a time when most services were poorly configured or had vulnerabilities, attacks to gain access to systems were common, mainly via services such as Telnet or Secure Shell (SSH). The article by Pang [7] describes these situations, classifying attacks by the protocol, application used and even by the specific type of exploit being used (where relevant). At that time, attacks were mainly targeted at misconfigured services or services with vulnerabilities, especially those that allowed access to the system.

The Internet had been around for many years before Pang's paper, and in 2007 Allman [1] wrote a paper showing the analysis of 12.5 years of network logs obtained at the Lawrence Berkeley National Laboratory (LBNL). The study makes an interesting distinction between what are scans and what are not, identifying scans as those connections that are made from a single host or computer. On the other hand, it analyses the most scanned ports over time, the most important being HTTP, SMB, NetBIOS, RPC, SQL, FTP, SSH, Telnet, with less intensity SMTP and DNS, and only during a period that starts with a lot of intensity in 2004 and then decreases until the end of the study, port 9898/TCP used by the backdoor Sasser [10].

During this period, events occurred that may be commonplace today, but were unprecedented at the time. In October 2008, the Conficker worm appeared, which was a notorious case because approximately 6 percent of the computers on the Internet at the time were infected by this worm. The worm used scanning techniques to infect other computers as it attempted to connect to the Windows Server service and exploit a vulnerability in it, generating IBR traffic [11].

A report on the status of IBR was produced in 2010, following on from Pang's earlier analysis [7]. This report, called *Internet Background Radiation Revisited* [12], presents some new concepts and views.

The first concept is related to the IPv4 address exhaustion that occurred in those years. In 2004, when Pang wrote his study, there were 1.4 billion IP addresses in use [13]. In 2010, IP address exhaustion was approaching, which occurred in 2011 [14], and therefore address blocks of size /8 were effectively allocated. This led to a larger attack surface on the Internet, but also to a larger number of potential attackers.

As addresses run out, new network telescopes would not be able to use as high addressing volumes as existing ones, which might even have to reduce their addresses. For example, CAIDA went from using a full /8-size block [15] to a /9-size block and a /10-size block, losing 25 percent of its addressing space, as shown in the recent network telescope architecture [16].

With the increase in the number of addresses used and therefore the attack surface, new mass scanning techniques emerged in 2013. Until then, it was common practice to perform network and port scans using tools such as `Nmap` [20]. However, `Nmap` is not designed for mass scanning. This year, two tools were developed for this purpose. The first one is ZMap [21], which was presented at USENIX and at the time allowed a complete scan of the entire Internet address space in about 45 min, using one computer and a Gigabit Ethernet connection. ZMap states on its website [22] that it is currently capable of performing a full analysis of the Internet in 5 min using a 10 Gigabit Ethernet connection, which currently requires a corporate or university type connection for that amount of traffic. The second tool that appeared in 2013 was MASSCAN [23], which has a similar performance and a rate of 10 million packets per second for a 10 Gigabit Ethernet connection, also taking 5 min to analyse the entire Internet.

From 2014 onwards, IBR traffic has followed a much more continuous pattern. The study "An Internet-Wide View of Internet-Wide Scanning" [24] took a new look at IBR traffic, with new patterns emerging and becoming commonplace. According to the article, network scanning was widespread across the Internet, with the use of ZMap and Masscan tools increasing and botnets growing. The source of malicious scans were no longer primarily network-scanning botnets, as in previous years' studies, but hosting providers, a new actor. Nowadays, previous attacks, such as brute force attacks on SSH or Telnet connections, are still being used, but there are also attacks on emerging vulnerabilities, such as a flaw in Linksys routers, OpenSSL Heartbleed or vulnerabilities in NTP services.

The attackers' procedure is that, when a new vulnerability is identified, mass analysis tools are used to scan the entire address space and find the targets. The attack is then executed using the appropriate tools. All of this is done within 24 h of the vulnerability.

This trend has continued in subsequent years, following the same approach and only varying the type of vulnerability. In 2018 and 2019, there has been a strong impact with Mirai botnet targeting distributed denial of service (DDoS) and DDoS attacks via reflection [25]. These attacks have an impact on the information collected in network telescopes, both because of the origin of the attacks and because of their content and volume.

Studies related to network telescopes and the information they collect are still important today. There have been recent publications looking at alternative ways of creating network telescopes, such as using clouds [26,27], as well as studies related to network telescopes for capturing IPv6 traffic [28] or analysing the data collected [29,30].

Table 1
Known network telescopes.

Network	Addresses	Name	Date	Reference
1/8	16M	APNIC	23/03/2010 - 30/03/2010	[12]
44/8	16M	UCSD N.T.	01/01/2001 - 04/06/2019	[15]
44/8	12M	UCSD N.T.	05/06/2019 - Present	[15]
35/8	11M	Merit Network	05/10/2005 - Unknown	[15]
50/8	16M	ARIN	12/03/2010 - 19/03/2010	[12]
107/8	16M	ARIN	03/25/2010 - 03/31/2010	[12]
Various	1300 networks	Akamai	In 2009 and 2019	[6]
/16	65K	HEAnet	03/2019 (1 week)	[17]
/15	131K	SURFNet	Unknown	[18]
2a10::/12	2^{52} /64–subnets	RIPE NCC	2020-01-13 - 2020-01-16	[19]

2.1. IBR traffic acquisition

The different types of IBR traffic shown above have been captured using a variety of methods, including firewalls, Intrusion Detection Systems (IDSs), and also darknets and network telescopes, which are networks that are not in use and therefore do not send traffic, so the traffic that reaches them is considered IBR. Table 1 shows some of the telescopes used in the referenced articles in the previous section. The *UCSD Network Telescope*, also known as *CAIDA*, is one of the most used and significant telescopes in numerous research articles.

On the other hand, the introduction of IBR traffic detectors interspersed with service-providing devices [6] enables the identification of attackers that do not interact with unresponsive networks, such as classical network telescopes. This phenomenon is often referred to as the "greynet" [31]. An example of this type of network telescope is the Akamai telescope. Unlike the other telescopes, it was built using Akamai's own infrastructure, with the IBR traffic collection nodes interleaved between the nodes used to deliver content to its customers. This network telescope does not use a single aggregated network, but multiple networks spread across different continents. Attackers who detect that there is traffic coming from the network will continue to attack the server nodes and will inadvertently attempt to attack the IBR traffic collection nodes.

Recently, new methods of acquiring IBR traffic have emerged. The use of partial responses to receive traffic [4] has made it possible to simulate open ports that attackers identify as victims, allowing them to receive a subsequent attack that would not exist without the response.

There are many articles on IBR traffic analysis, some of them dating back many years, such as the aforementioned [7,12,24], several articles on IBR traffic or IBR traffic analysis are written practically every year (botnets [32], crawlers, etc.) and there are several active network telescopes (known and private) showing the importance of IBR traffic. It is therefore interesting to explore new, more dynamic and easier ways to obtain and analyse this type of traffic. Thus, according to the studies analysed, the use of distributed sensors, mixed with devices that provide services, as well as sensors that respond to attacks, allows the acquisition of IBR traffic that would otherwise be unattainable.

After analysing the evolution of network telescopes, this paper proposes HoDiNT as a network telescope based on a distributed architecture of probes to capture Internet background traffic. HoDiNT is designed to be a low-cost, easy-to-deploy, distributed solution that supports advanced attacks.

3. HoDiNT architecture

The complete architecture of the HoDiNT sensors for distributed traffic monitoring is shown in Fig. 1. It consists of the following elements:

1. Router: This is the component responsible for receiving all user traffic and sending IBR traffic to the sensor. For cost reasons, the equipment provided by the Internet Service Providers (ISPs) is used.

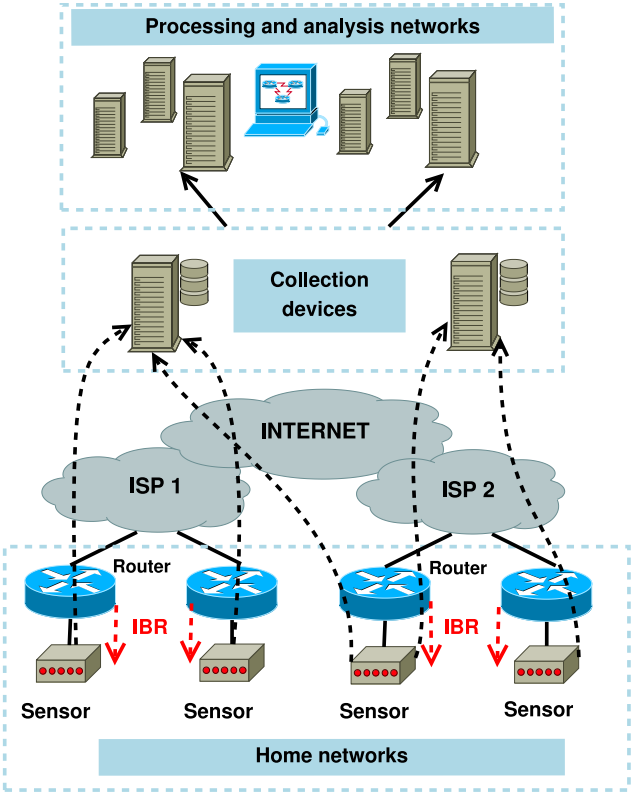


Fig. 1. HoDiNT's components.

2. Sensor: This is the main component of the proposed architecture, since it is responsible for receiving IBR traffic from the router, processing it in real time, performing the functions deemed appropriate for this sensor, such as responding or not to the traffic, and reporting the data to the Collection devices.
3. Collection devices: They receive the information from the sensors and store it for further processing and analysis.
4. Processing and analysis network: These components receive the information from the Collection devices and perform all the analysis and processing of the information.

3.1. Router

The router is responsible for the communication between the user's devices and the Internet. Due to the use, in IPv4, of private IP addresses in the home network, when the traffic returns from the destination node in the Internet to the source node in the home network, the router must forward the traffic. In IPv4 the mechanism used for forwarding to private addresses is called NAT (Network Address Translation). In IPv6, this is done through routing and a state machine firewall. In this paper

we will only consider the use of IPv4, which is the protocol commonly used for IBR monitoring by network telescopes, leaving IPv6 for future studies.

In the case of non-expected IBR traffic, when the communication is initiated from the Internet, the traffic arrives at the router and the router looks for the previously established connection. Since this connection was not previously established, the router drops the traffic.

In the operator-provided router, there are usually two ways to send this type of traffic from the Internet (incoming direction) to a device on the internal network:

1. Port forwarding. A mechanism where a static entry is included in the NAT configuration to indicate that any communication destined for a particular port will be forwarded to the configured IP address and port.
2. DMZ host. This is a mechanism that allows a node of the internal network to be set up as a DMZ (DeMilitarised Zone) device by specifying its IP address. In this way, all incoming connections that do not have a previously established connection in the NAT table will be redirected to it.

As all traffic that has not been initiated by the user is really not expected, it is considered by the system as IBR traffic. For forwarding to the sensor the DMZ host functionality, if available, is suggested, being much easier to configure as it allows traffic from all ports to be redirected.

Other port forwarding configuration options, such as UPnP (Universal Plug and Play), have been considered to allow dynamic configuration. It has the advantage of not having to manually change the configuration on the router, making installation and configuration easier. In addition, UPnP would allow the external IP address of the router to be known. However, this dynamic configuration requires that the router has UPnP support and also that the user wants to activate it, which may present reluctance for security reasons. On the other hand, UPnP operation is not compatible with ICMP, which would make it impossible to redirect this protocol. For these reasons, it has been considered better to manually configure port forwarding on the router only once, thus avoiding dependency on other protocols.

3.2. Sensor

The sensor is the key component of the architecture. It is responsible for receiving traffic and reporting it to the Collection devices. Two different sensor configurations are used in this study, namely "active sensors" and "passive sensors". Passive sensors are those that receive and store IBR traffic and do not respond to the source IP of the traffic. Active sensors respond to the source IP of the IBR traffic, allowing interaction with the source and enabling detection of advanced or two-phase attacks [6].

The sensor consists of the following functional modules (Fig. 2):

- Capture module: This module performs packet filtering and acquisition in PCAP format, capturing only IBR traffic and storing it in the file system. If the sensor is configured in active mode, the capture module will be responsible for interacting with the source of the IBR traffic.
- Collection module: This module is responsible for managing the PCAP captures from the file system and sending them to the Collection devices. It is responsible for controlling the size of files and managing the transfer of files when they exceed a certain size or number of packets to the Collection devices configured.
- Configuration module: This module allows the sensor to be configured. It also provides management connections from the Collection devices and sends status information about the sensor.

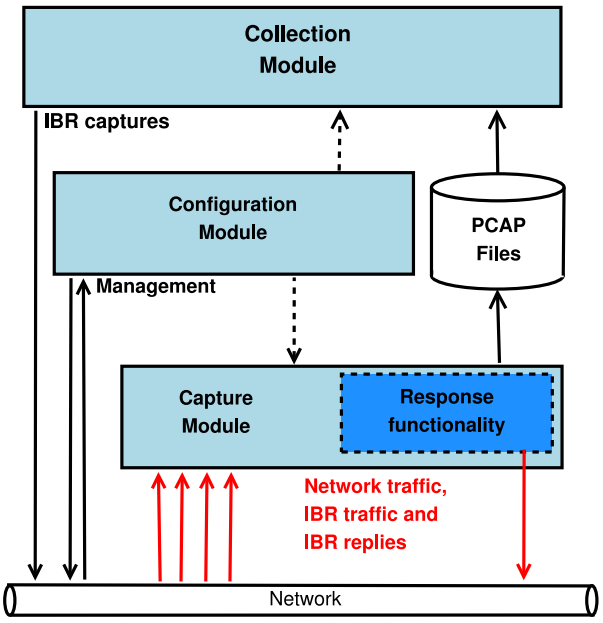


Fig. 2. Functional blocks of the sensor.

3.2.1. Capture module

The capture module is responsible for filtering all the traffic arriving at the sensor, extracting IBR traffic for capture, storage and, if configured, response.

The traffic is captured using PCAP (Packet CAPture) files, a standard traffic capture format used by most network telescopes. The sensor receives the following types of traffic:

1. IBR traffic: Its IP source is the address of the IBR traffic generator (or an address spoofed by the attacker) and its physical source address (MAC) is that of the router device. The destination addresses, both physical and IP, are those of the sensor.
2. Sensor control and management traffic: This traffic is the traffic necessary for the operation of the sensor, but it is not IBR traffic and, therefore, it must not be reported. It consists of management traffic such as SSH (Secure SHell), DNS (Domain Name Service) resolution traffic, reporting traffic to the Collection devices and time synchronisation via NTP (Network Time Protocol).
3. Unwanted traffic: This is the traffic of the local home network where the sensor is located that should not be reported, for example layer 2 and layer 3 broadcast and multicast traffic. This traffic, if it has a high volume, such as the multicast traffic received for multimedia streaming, could have an impact on the sensor, because it would have to process it.

Since HoDiNT needs to be able to react to IBR traffic in order to be able to access attacks in two phases, it is necessary to be able to treat traffic packets individually. This is done using *Python* and the free *Scapy* library. As this processing is CPU intensive, a first filtering with *iptables* is used to remove all traffic that is neither IBR nor necessary for the operation of the sensor. The traffic to be dropped would be as follows:

- Incoming traffic that does not have the router's MAC address as its source MAC address. This allows IBR traffic and sensor operation to pass through and avoids all local network traffic. If local management via SSH is requested in the configuration, a rule is created to accept this traffic as an exception.
- Layer 2 and layer 3 broadcast and multicast traffic.

Once the traffic has passed the first filter, it arrives at *Python/Scapy*, where the IBR traffic is separated from the sensor control and management traffic by a filter. The Scapy filter allows the following traffic:

- Traffic must be IPv4.
- The traffic has to have the sensor as source or destination, allowing IBR type communications and also active sensor responses. For this, it is verified that the IP and MAC addresses are those of the sensor in origin or destination.
- The management traffic with Collection devices and DNS servers is allowed but is not stored in the PCAP files. This is done by filtering the IP addresses of these servers (DNS server and Collection devices), thus allowing real IBR (all TCP and UDP ports and ICMP) traffic to be stored.

IBR traffic is captured by the *sniff* function of *scapy*, which applies the filter described above, when it is passed to the *handler* function. This can be seen in the code below:

```
# Create the filter. Prevent non-IBR packets.
hw_filter = self.create_sniff_filter()

"""
Listen on the interface (eth0, wlan0).
Call handler for processing.
"""
sniff(iface=self.interface,
      prn=self.handler,
      filter=hw_filter, store=0)
```

The *handler* function, in order to capture, calls the *pcap_write* function, which writes the packet to a *PCAP* file.

To facilitate the subsequent analysis of the traffic and preventing private IP addresses from appearing in the captures, a pre-processing is performed on this traffic, replacing the private IP address of the sensor, which is contained in the IP header, with the public IP address of the router. In spite of the fact that this change affects the header's checksum, which was previously calculated by the router. In the pre-processing it is decided not to recalculate it in order to keep most of the fields intact.

Getting traffic from the network interfaces is a task that can be done with very good performance using kernel space functions. However, processing the information at the user plane (e.g. to change the IP address, and especially the real-time response, as we will see in the next section) can mean a drop in performance. It would be possible to implement a pure capture sensor and additional scripts for the post-processing of the capture, but this would not be valid for active sensors (which require a real-time response) and two separate developments would have to be maintained. To validate the feasibility of the solution, we have preferred to use a single piece of software that is sufficiently simple to maintain and compatible with both types of sensors.

To obtain the public IP address, the capture module connects to the Collection devices via *HTTPS* and queries their public IP address via a remote script. As it is not feasible to make a connection for every packet received, a connection is made every five minutes.

3.2.2. IBR traffic response functionality

In active mode, the capture module allows TCP - SYN, UDP and also ICMP traffic to be responded too. This simulates that there is a service to be attacked and increases the level of interaction and therefore the ability to obtain additional attack information. To avoid potential security issues and improve response time, the information received is not used for the response. Generating customised information, for example based on destination port or packet payload, is possible with Scapy, but requires further software development, which is closer to the functionality of a honeypot than that of a network telescope.

In order to protect the sensor and also to avoid being part of attacks, typically DDoS and amplification attacks, carried out by spoofed sources using IP spoofing techniques, the responses are limited. The sensor will only respond to the first packet of a connection for a period of five minutes. This is done by tracking (and counting) the packets for all connections for each of the specified protocols.

In the case of the TCP protocol, only packets with the *flag SYN* are answered, thus continuing the possibility of establishing the connection to the indicated port. The reply packet therefore has the *flags ACK+SYN* active. In UDP, since there is no connection, the response is made by including a character in the *payload*, which has a random numeric value between 0 and 9. In the case of ICMP, only packets of type *echo request* are responded to with a *echo reply* packet.

In all cases, a limit of 50 packets per connection is maintained during the 5 min period, so that if this is exceeded, the response limit is extended by a further 5 min, thus avoiding the need to respond to continuous attacks.

In the case of the TCP protocol, in normal operation, the operating system responds with the RST flag to any connection directed to a port on which it has no active service. To prevent the response generated by the operating system from being sent and the connection not being established, it is necessary to create a rule using *iptables* that prevents this packet from being sent to attackers.

This two-phase interaction mechanism can make the sensor appear to implement typical functionalities of a honeypot. One of the most commonly used definitions of a honeypot is that it is "a decoy computer resource whose value lies in being probed, attacked or compromised" [33]. To achieve this goal, depending on the level of interaction with the attacker, honeypots can send banners, simulate specific services, allow access to the system, etc. In the case of HoDiNT, the sensors respond in order to emulate the existence of a system and services, but the response contains as little information as possible, the response is generic to all ports, it does not include banners, it does not emulate any service and, above all, the sensors respond only once, which avoids complex and structured attacks. Therefore, it does not require a great deal of software development and the information it can collect does not require in-depth and structured analysis, two main pillars of honeypot development [34]. For these reasons, HoDiNT is more a distributed network telescope than a low-interaction honeypot.

3.2.3. Collection module

The Collection module is responsible for sending the *PCAP* files generated by the Capture module to the different Collection devices. The sending is done via HTTPS, including a unique user and password for each sensor, which are obtained from the configuration module.

The module is also responsible for counting the number of packets received in each *PCAP* file, as well as the time elapsed since the last file was sent. The aim is to keep the minimum amount of information captured in the sensor, so that in the event of a failover or crash, the information does not get lost, also minimising the constant sending of information to the Collection devices. It has been established that if the number of packets saved in a PCAP file reaches a limit (1,000 by default) or one hour has passed since the last transmission, the captured information will be sent. The choice of values for time and number of packets has been made based on an initial exploration of capture volumes of a few sensors. The purpose was to send information within a reasonable period of time, preventing the sensors from storing a large amount of information locally before sending it to the collection devices.

Another function of this module is to rotate the *PCAP* file on which the capture module writes, so that the capture module always writes to the same destination, but physically it is a changing file.

Finally, it checks that if the capture process receives a completion signal (syscall), rotates the file and sends it, so that the information obtained is not lost.

3.2.4. Configuration module

The configuration module is responsible for reading the sensor configuration file and providing the information to the other modules.

In this configuration file, sensor-specific parameters are specified, such as the network interface *Ethernet* or *Wi-Fi* (including the name of the access point and, in this case, the password), the IP address that the sensor will have and that will be configured statically in the *router*, the DNS server, the addresses of the Collection devices, as well as the user and password that will be used to send the data to the Collection devices. In addition, the user can specify a *SSH* port which, by means of reverse *SSH*, allows the remote administration of the sensor. A password can also be specified for management from the local network.

If the user creates a configuration file and places it in the */boot* directory, the sensor will detect it and switch to configuration mode. In this mode, the sensor is configured from scratch. This allows the sensor to be configured as many times as required without having to reinstall it.

This configuration file sets the configuration of the first part of the filter module (*iptables*) and defines all the necessary parameters in the main configuration file. Some of these parameters include *PCAP* storage directories, enable responses for *TCP, UDP, ICMP*, user-specified networks to filter, *bogon* networks, logging level (*log*), and so on.

In addition, this module retrieves the configuration used by *iptables*, as well as the CPU and memory load values, sending them to the Collection devices from time to time, allowing the status of the sensor to be known.

3.2.5. Integration of all sensor modules

Once the different modules of the sensor have been described, it is possible to show the interactions between them, as well as the flow followed by an IBR traffic packet.

The sensor launches three processes through the task scheduler *crontab*:

1. A stateful process that collects information from *iptables*, CPU and memory and sends it to the Collection devices. It is a process that runs every five minutes, collecting and sending data. This information is used to monitor the behaviour of the firewall, the packets dropped by each rule, and the CPU and memory load on the sensors.
2. Remote management via SSH. It is executed by means of a script-type watchdog, which checks if this process is already running and, if not, restarts it again. It establishes a *SSH* connection to one of the Collection devices, if configured by the user.
3. IBR Traffic Manager process. This is the main process, it is also runs with a *watchdog* and performs the functions described by the capture and collection modules.

Due to the large number of actions that the IBR traffic manager process has to perform in real time for each packet received, it is necessary to structure its operation in different execution threads, in order to parallelise the processing of the packets and the rest of the functions.

The IBR Traffic Manager process calls the configuration module at startup, reads all the parameters and activates the logging level specified by the user. It then calls the collection module to set up the monitoring of the system process termination signals, which allows the sending of the data processed so far in case of process termination. It obtains the information from the network environment to create the environment-dependent filters and launches the different threads:

1. IBR traffic capture thread: This thread is responsible for the processes described by the capture module (see 3.2.1), such as filtering, receiving, storing received traffic, and generating responses.
2. Response management thread: This thread is responsible for managing the TCP, UDP and ICMP response status tables, maintaining the information about responses made and cleaning the tables each time they are refreshed.
3. Collection thread: This thread performs the functions described in the collection module, sending the files to the Collection devices and performing the change (rotation) of the files.
4. Public IP address update thread: This thread, from the Capture module, is responsible for consulting the sensor's public IP address of the sensor from time to time.

This execution structure allows IBR traffic to be captured and responded to in real time using low performance hardware.

3.3. Collection devices

Collection devices receive the information sent by the sensors and provide the services necessary for the sensors to operate:

- File collection: The Collection devices receive the *PCAP* files from the sensors using a secure web server and authentication with a user name and password. These files are stored to be sent to the processing and analysis modules.
- Time server: Since it is necessary that the *PCAP* records are synchronised, the sensors maintain the time configuration through the NTP protocol, which is served by the Collection devices.
- Public IP Resolver Server. The Collection devices respond to the queries from the sensors to find out their public IP, so that it can be inserted in the capture files.
- Management server. From the Collection devices it is possible to connect to the sensors to manage and troubleshoot them. This communication is done by reverse SSH, where the sensor connects to the Collection devices and opens a local port to connect back to the sensor. This connection uses a shared key.

For higher availability, each sensor can have multiple Collection devices configured and will send the *PCAP* files to all of them.

3.4. Processing and analysis modules

The processing and analysis modules obtain the *PCAP* files from the Collection devices and process them.

The processing is carried out by means of scripts that generate information, statistics and analysis and also by means of software such as ELK (ElasticSearch, Logstash and Kibana) that allow a more visual analysis.

3.5. HoDiNT features

After describing the architecture of HoDiNT, it can be seen that it has the following characteristics that distinguish it from classical network telescopes:

1. Low cost. By using low-cost and low-power hardware, such as Raspberry Pi devices, it is possible to create a sensor network for IBR traffic acquisition. The sensors are deployed over existing home connections (ADSL, FTTH, etc.), which also saves on connection costs.
2. Easy installation. The sensors are installed using a disk image that is recorded on a MicroSD card and a ten-line configuration file. In addition, the IP address that the sensor will have in the traffic forwarding section must be configured in the router. All these configurations are accessible to network telescope operators.

3. Distributed solution. Sensors can collect background noise traffic simultaneously with and without interfering with the user's connection. This allows the creation of a distributed and scalable network, increasing the number of probes to capture more traffic samples and avoiding the capture of repetitive and irrelevant traffic. By embedding the sensors in the operator's various client connections, the sensors are not identifiable. In addition, by using the addressing of the home connection, no additional addressing is consumed and no correlative addressing is used.
4. Advanced attack support. Sensors are able to respond intelligently to attacker connections, allowing two-phase attacks to be detected [4]. In addition, it is possible to have sensors that responds to attackers and also sensors that operate without responding to connections, through hybrid behaviour. This allows the comparison of possible behaviours that occur when responding versus not responding and also allows correlation of data between the two types of sensors.

3.6. Deployment model

To deploy the sensors, a minimal image for Raspberry PI has been created. It is based on a RaspiOS Lite image, version Bookworm for ARHhf (ARM hard float) architecture. On this image, unneeded software packages have been removed and the basic software required to use the sensor has been included, the main software being Python version 3.9 and the Scapy library version 2.5.0. NTP time synchronisation software is also included. The image includes the necessary SSH configurations to allow reverse SSH connections from the collectors. For maintenance and version upgrades, the image is created using HashiCorp's "Packer" software.

The software that performs the sensor function is developed separately from the image generation. The set of Python scripts that perform the functions described earlier in this section, such as capturing, sending to the collectors, etc., are included in a Debian package (.deb). This allows the package to be installed on systems other than the Raspberry PI image, and also makes it easy to upgrade the sensors. The Debian package is installed during the image building process described above.

The sensors are hosted by volunteers who receive a Raspberry PI device with an SD card with the software pre-installed. The user configures a DMZ host in the router, communicates the assigned IP address to the researchers and receives a fully configured sensor. When the sensor starts up, it connects to its collectors, establishes a reverse SSH connection for remote management and checks that it has the latest software version and the mode in which it should operate (active or passive mode). These checks are performed hourly via HTTPS requests and it is possible to remotely update the sensors and also change the operating modes to perform the desired tests and analyses.

The Collection devices have a JSON file containing the configuration of the collector itself, the configuration of the Processing and analysis nodes, and the information of all the sensors. In the case of the Collection device configurations, the main information available is the location of the PCAP files received, and the files and versions of the Debian packages available for the sensors. In the case of the Processing and analysis nodes, the directory information is used for file synchronisation. For the sensors, the JSON file contains all the configuration, mainly the working modes, credentials and software versions (it is possible to test versions on some sensors before deploying to all installed sensors). Although it is possible to synchronise the files via SSH/SCP or RSync, it was preferred to use HTTPS and the configuration in a single JSON file for all services, so that a homogeneous and centralised configuration and maintenance process is available.

By monitoring the connections of the sensors and the files being sent, it is possible to know if the sensors are active or if there is a problem. There is also a notification mechanism via telegram when a problem is detected.

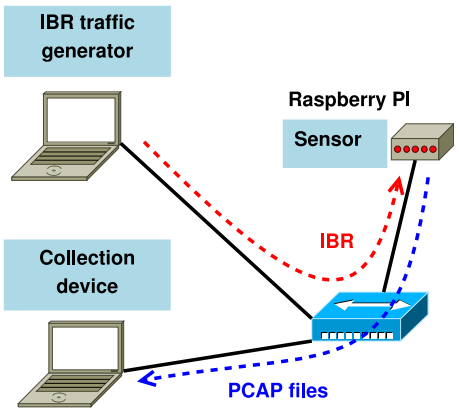


Fig. 3. Controlled environment setup.

At the time of writing, there are eighteen active sensors spread across five different ISPs. Two of the sensors are located behind ISP connections with CG-NAT (Carrier Grade NAT) to detect unexpected traffic and were not used for the analysis in this paper. The remaining sensors are located on FTTH connections with public IP addresses. Ten of the sixteen sensors were used for these analyses because they were connected for the entire duration of the tests.

4. Analysis and results

The ability of the HoDiNT architecture sensors to receive, store and respond to real-time traffic has been evaluated. The following analyses have been performed:

1. *Performance analysis.* Performance tests have been carried out in with *hping3* in a controlled environment with synthetic traffic, which makes it possible to evaluate the real functioning and performance of the sensors in extreme traffic situations.
2. *Comparison between passive and active sensors.* Using sensors in a real environment, the differences between traffic received by active and passive sensors are analysed.
3. *Exploring IBR traffic port distribution.* The distribution of TCP and UDP ports of received traffic is shown.
4. *Exploring known attacks in received IBR traffic.* The payload of captured IBR traffic is compared with information from databases of known attacks and with rules from IDS intrusion detection systems.

4.1. Performance analysis

The first analysis was to measure the performance of the devices in a controlled environment. The aim is to assess the feasibility of using the devices as sensors in the traffic acquisition network, particularly under challenging traffic conditions.

The setup for this environment is shown in Fig. 3. We used one computer to act as collection device, another computer to act as IBR traffic generator, and various Raspberry Pi models, connected one at a time, to act as sensors, receiving the IBR traffic. The three devices are wired together using a Gigabit Ethernet switch. The network has no connection to other networks or the Internet.

The Raspberry Pi devices used as sensors are the Raspberry Pi 1B+, Raspberry Pi 2B, Raspberry Pi 3B+ and Raspberry Pi 4 models. As shown in Table 2, the Raspberry Pi 1B+ model uses a 32-bit architecture, with a single core and a single thread, while the other models use 64-bit processors, with 4 cores and 4 threads [35]. The same software configuration has been used for all devices, with no model or architecture specific settings.

Table 2
Raspberry Pi specifications.

Model	Arch	CPU Speed	Cores	Threads
RPi 1B+	32	0.7 GHz	1	1
RPi 2B	64	0.9 GHz	4	4
RPi 3B+	64	1.4 GHz	4	4
RPi 4B	64	1.5 GHz	4	4

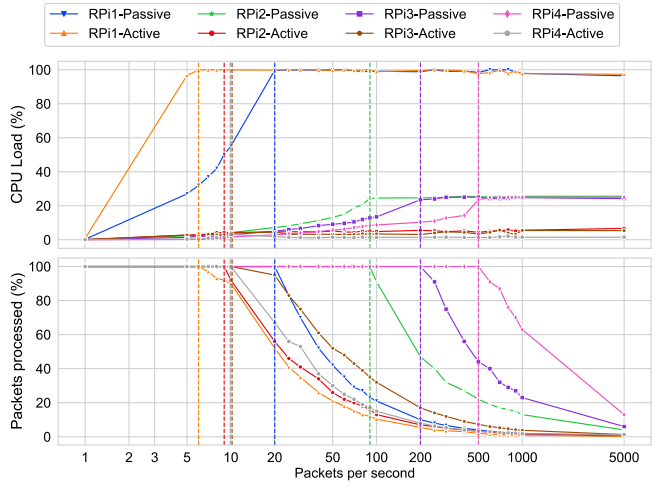


Fig. 4. CPU load and packets processed in TCP tests.

In this first test, 10,000 packets were sent at different packets per second (pps) rates, from 1 to 5,000 pps, to analyse the CPU load on the devices and the number of packets they were able to process and store. These tests were conducted independently for the TCP, UDP, and ICMP protocols and repeated with the sensor both in active mode (configured to respond) and passive mode (configured not to respond). The tests were performed three times for each measurement of every Raspberry Pi model to ensure statistical significance (mean values were taken).

To perform these tests the standard *hping3* was used as traffic generation tool. We used 8 bytes of data (`--data 8`), specified the base port (`-s 10000`), the number of packets to send (`-c 10000`) and various interval times (parameter `-i`). The SYN option (`-S`) was used for TCP, while the default parameters (`--udp` and `--icmp`) were used for UDP and ICMP.

The obtained results for the tests of the three protocols (ICMP, TCP, UDP) are very similar. Therefore, only the results of the TCP protocol are shown as an example. Fig. 4 displays the evolution of the CPU load in relation to the traffic rate sent to the sensor. While, as expected, the CPU load increases as the sent traffic rate increases, there is a difference in behaviour between active and passive sensors.

For passive sensors, when the CPU reaches its maximum load, the processing of packets is affected and thus, the number of processed packets becomes lower. On the single-core Raspberry Pi 1, this happens when the CPU reaches 100% at 20 pps. On the Raspberry Pi 2, Raspberry Pi 3 and Raspberry Pi 4 models, which have 4 CPU cores, this situation occurs at a CPU load of 25%. This is because although the implementation uses threads, only one core is used for execution. This is a known limitation of Python due to the use of GIL (Python Global Interpreter Lock) and could be solved in future work by using additional libraries such as *multiprocessing*. The dashed lines in Fig. 4 and Table 3 show the limit without packet loss for passive mode sensors (both TCP and UDP). It is important to note that these limits may be exceeded in case of occasional bursts of traffic.

When the sensor is configured in active mode, it has to perform more processing, i.e. it has to update the received packet tables and decide whether to respond to the source and store the packets. By performing more processing for the received traffic, the number of packets the device can handle is reduced. This situation occurs in all models from a few packets per second. The packet loss curves displayed in all models are similar. This is particularly true for the TCP and UDP protocols, as for ICMP the response and non-response behaviour is similar, as only the first packet is responded to, increasing the processing of the device very little. Similar to the passive sensors, the Table 3 and Fig. 4 show the limit without packet loss in active mode.

Table 3
Maximum rate in packets per second observed for the different Raspberry Pi models without packet loss.

Model	Packets per second	
	Passive	Active
Raspberry Pi 1	20	5
Raspberry Pi 2	95	9
Raspberry Pi 3	200	10
Raspberry Pi 4	500	10

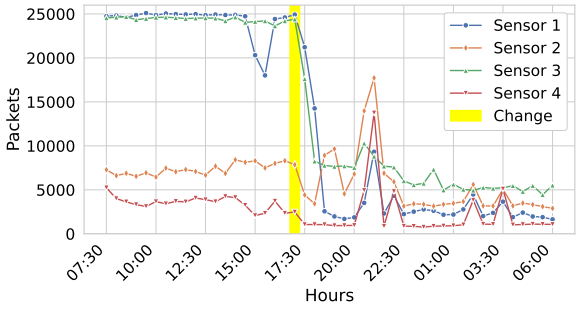


Fig. 5. Packets received by 4 sensors changing from active to passive mode.

As mentioned above, multiple CPU threads are used to process the packets. Some of these threads are common to both passive and active sensors, but others are only used in active devices, such as those used for sending reply packets or managing connection tables.

It may be possible to improve the software implementation or use more powerful hardware to alleviate this situation. However, the objective of this initial work is to explore the possibilities of capturing IBR traffic in a distributed manner.

Finally, we can see that the Raspberry Pi 3 is able to process more packets in active mode than the Raspberry Pi 4. Based on the benchmarks analysed [36,37], the Raspberry Pi 4 is expected to be faster. The test were repeated and the results confirmed this anomaly.

4.2. Comparison between passive and active sensors

One of the things to check is whether active and passive sensors receive the same amount of traffic. To analyse this, four sensors were taken and kept active for several days. Then, on 27 October 2023, the mode of these four devices was changed from active to passive at 17:15. Fig. 5 shows the number of packets received by each sensor grouped in 30 min blocks. Here it can be seen how the volume of received traffic on the four devices decreases as soon as the change (marked in yellow) is made. Note that the traffic received by the sensors in active mode is significantly higher than that received in passive mode. In Section 4.2.2 we explore the reasons for this difference.

4.2.1. Description of captured traffic

After this change, traffic data was collected for 31 days using 10 sensors, 5 in active mode and 5 in passive mode to compare passive and active mode sensors.

The result of the daily average (rounded down) of all packets captured during this period is shown in Table 4. Active sensors are identified with “A” in the Sensor column and passive sensors are identified with “P”.

Table 4
Mean (rounded down) number of daily packets received and replied by the sensors over a 31-day period.

Sensor	TCP		UDP		ICMP	
	In	Out	In	Out	In	Out
A1	150,703	22,327	38,278	6,210	178	0
A2	141,208	27,350	14,822	3,810	355	0
A3	227,705	30,597	89,514	23,078	1,011	256
A4	180,999	33,106	12,585	2,463	4,417	1,314
A5	128,329	25,072	5,361	1,378	236	0
P1	55,755	0	156,220	0	1,859	0
P2	19,233	0	7,538	0	100	0
P3	23,476	0	47,797	0	298	0
P4	17,267	0	11,796	0	2,227	0
P5	27,657	0	175,788	0	0	0

The table shows a large disparity in the ratio between captured and replied traffic for the three protocols, especially for ICMP. In addition, there is a large difference in the traffic volume of the TCP protocol between the active and passive sensors.

For other transport protocols, the packet volume is on average less than 5 packets per day and for that reason they have not been included in Table 4. These protocols are IPv6 (IPv6 encapsulated in IP), IP-ENCAP (IP encapsulated in IP), SCTP (Stream Control Transmission Protocol), ESP (Encapsulating Security Payload), RSVP-E2E-IGNORE (Reservation Protocol (RSVP) End-to-End Ignore) and GRE (General Routing Encapsulation).

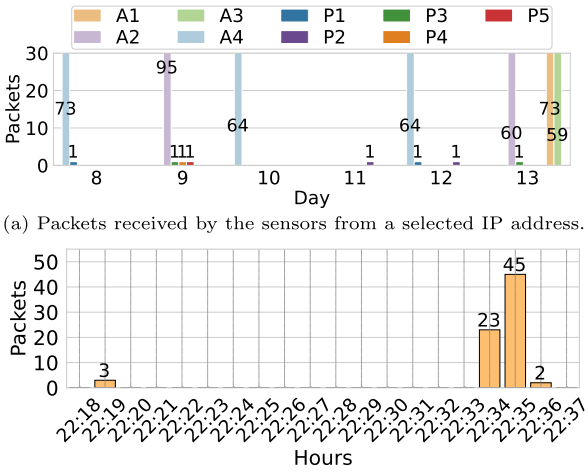
4.2.2. Analysis of the difference in traffic between active and passive sensors

To investigate a possible explanation for the variation in traffic received by passive and active mode sensors, communications from the same source IP addresses directed to both types of sensors were analysed. Active sensors were able to collect TCP traffic patterns that did not appear on passive sensors, even when the same source sent traffic to both types of sensors. In Fig. 6(a), the traffic received from a selected source IP address is represented for the 9 out of 10 sensors that received traffic from that source. The active sensors (A1-A4) show sets of more than 50 packets, while the passive sensor shows only one packet (the SYN packet sent by the attacker) at the same time. This is because, as shown in Fig. 6(b), the active sensors, receive the SYN packet, respond with an ACK+SYN packet, and the IBR traffic source closes the connection with an RST packet. After a few minutes, varying between 15 and 55 min, the same IP address sends a series of packets that are only observed on the active sensors and this is the reason why active sensors receive a greater amount of IBR traffic from these specific IP addresses.

For example, in Fig. 6(b), we can see a zoom on the set of packets received by sensor A1 on day 13th, where three packets (SYN, ACK/SYN, RST) are shown at 22:19 h and later, between 22:33 h and 22:36 h, it receives the set of attacks. This behaviour is present in a similar way on all active sensors, varying the time between the three-way handshake and the posterior burst of packets.

A two-phase behaviour is observed, as indicated by Spoki [4], although in our case we identify correlated behaviour at longer times than those considered in their study (10 min between phases).

This test demonstrates how, by using a distributed network telescope with active sensors, it is possible to identify different scanning patterns, such as source IP addresses, the times these addresses take to repeat connection attempts, and to obtain information about these sources, such as their networks, countries or autonomous systems [6]. In addition, it is possible to identify source IP addresses with similar behaviour between them, and following in the same pattern towards the active sensors, which cannot be detected by traditional telescopes. It is also possible to obtain this information with a very small number of sensors.



(a) Packets received by the sensors from a selected IP address.

(b) Zoom in packets received by active sensor (A1) on day 13th.

Fig. 6. IBR traffic received from a selected IP address by active and passive sensors.

Table 5
ICMP packets received and replied by the sensors during a 31 days period.

Sensor	ICMP type	Packets	
		In	Out
A1	Destination Unreachable	5,534	0
	Time Exceeded	4	0
A2	Destination Unreachable	10,900	0
	Time Exceeded	130	0
A3	Destination Unreachable	7,236	0
	Echo Request/Reply	24,100	7,937
	Time Exceeded	11	0
	Timestamp	2	0
A4	Destination Unreachable	10,225	0
	Echo Request/Reply	126,709	40,739
	Time Exceeded	5	0
	Timestamp	4	0
A5	Destination Unreachable	7,320	0
	Time Exceeded	15	0
P1	Destination Unreachable	1	0
	Echo Request	57,650	0
P2	Echo Request	3,111	0
P3	Echo Request	9,254	0
P4	Echo Request	69,060	0
	Timestamp	5	0
P5	Echo Request	0	0

4.2.3. ICMP analysis

Due to limitations in the firmware of the routers, for the sensors A1, A2, A5 and P5, it was not possible to activate an ICMP traffic forwarding option between the router and the sensor. For this reason, these sensors appear with no outgoing traffic in Table 4. In these sensors the traffic received corresponds to other types of ICMP packets than the Echo request type. In the case of sensor P5, the incoming packets show a value of zero. It is observed that the active sensors A3 and A4, which are the sensors that respond to ICMP Echo traffic, present a higher number of received packets than the other active sensors.

Table 5 contains a more detailed information of all ICMP packets collected during the observation period. Analysing the different types of ICMP types, it is observed that the vast majority of packets in passive sensors are of type ICMP Echo. The Timestamp packets at sensors A3, A4 and P4 have different source IP addresses and are residual in nature. Note that this type of ICMP packets are typically used as a replacement for an Echo request packet to find out if a computer is active, to

Table 6
Main payloads found in UDP traffic and number of packets captured from each one.

Type	Payload	Packets
T1	DHT messages: `get_peers`, `ping`, `find_node`, `announce`	11,039,399
T2	`<tds:GetDeviceInformation />`	1,796,129
T3	`<?xml version="1.0" encoding="utf-8"?><Probe><Uuid>string</Uuid><Types>inquiry</Types></Probe>`	934,287
T4	`~*\x9d\x0c@\xd0@\xca=-H-@\xe4\xca\xd8\x00\x00`	162,397
T5	`'M-SEARCH\r\nST:ssdp:all\r\nMAN:"ssdp:discover"\r\n'`	126,088

perform geolocation or to identify the remote operating system [38]. It is also possible to use the timestamp returned by the target to attack time-based security algorithms, such as random number generators, or time-based authentication mechanisms [39]. Regarding the ICMP Time Exceeded packets, the total number of packets is 165, where half of these packets have been sent by the same source IP address to two different probes (A2 and A3). However, there are two particularly interesting details. The first one is that four contiguous IP addresses (185.x.x.34,35,37,38) generated 47 of these packets. These nodes have not generated any other ICMP messages and the destinations are only two sensors. The second detail is related to two sensors (A3 and A4), located in two different ISPs, that have received four packets, at separate time instants, where the source IP addresses are private (10.0.0.0/8 and 172.16.0.0/12). This might indicate that there is some kind of node (in this case three nodes) used by the ISPs to communicate with the clients' routers or to provide some network service, or that there is a configuration error in the network's anti-spoofing filtering, although it was not possible to verify the cause.

Packets with private source addresses were also found in ICMP echo packets. In this case, sensor A4 was constantly receiving ICMP Echo connections from a private IP address (10.0.0.0/8) belonging to the ISP using this sensor. This behaviour was subsequently observed in other sensors of the same ISP, which are not included in this analysis.

Regarding ICMP Echo packets, when dissecting the ICMP packet types in Table 5, a ratio of about 33% between received packets and responded packets is observed for sensors A3 and A4. This shows that traffic is sent in bursts or groups of packets, and since the sensor only responds to the first packet of the connection to avoid participating in DDoS attacks, this ratio of received to responded packets is obtained.

In addition to ICMP Echo packets, the vast majority of packets received at the active sensors are of the ICMP Destination Unreachable type. These packets are generated by TCP and UDP replies sent by the active sensors to the source IP addresses of the packets. The volume of these packets is similar in all active sensors, which indicates that regardless of the number of packets sent by the source, the procedure established in the sensors for responding to the packets is working correctly, responding to a single connection from time to time and giving a homogeneous response across the sensors. These responses can be used in future analysis to identify traffic corresponding to connections with source address spoofing or amplification attacks.

4.2.4. TCP and UDP analysis

Regarding TCP traffic, as shown in Table 4, there is a significant disparity in the number of packets received by active and passive sensors. Active sensors receive an average of 165,000 incoming packets per sensor, while outgoing packets average is 27,700. This means that 5.9 times more packets are received than sent. For passive sensors, the average number of packets received is around 28,700, almost a sixth of the traffic received by active sensors.

As analysed in Section 4.2.2, sensors responding to TCP traffic receive subsequent connections and also the number of packets sent in the second phase is relatively high. As seen above, this explains why active sensors have much more traffic in TCP than passive sensors. Furthermore, if we look at the average number of packets received by the passive sensors (28,700) and compare it with the average number of packets responded to by the active sensors (27,700), it shows a very close value. This is explained by the fact that the response mechanism only responds to the first packet, which would be the one received by the passive sensor, and discards all other packets from the same connection within five minutes. In cases where the second phase of the attack takes longer than five minutes, similar to the case shown in Section 4.2.2, this second phase is treated as a new connection, responding in the same way only to the first packet and increasing the number of responses by one packet.

When analysing the UDP protocol traffic captured by the sensors, no significant differences were found between sensors in active and passive mode. Even sensors in passive mode, such as P1 and P5, received a considerably higher volume of traffic.

When examining the collected data, it was found that large volumes of packets are continuously sent to a sensor from a single source. This situation has also been found in isolated cases for TCP protocol. In order to compare active and passive sensors, it is necessary to find out the impact of this type of traffic.

After several analyses attempting to correlate source IP address, destination ports and other information, it was determined that only five types of payloads accounted for the majority of the traffic. Out of a total of 17,350,669 collected packets, these payloads were present in 14,039,399 packets (80.92%). The strings of the payloads and the number of packets can be seen in Table 6. The DHT field corresponds to the strings "*get_peers*", "*ping*", "*find_node*" and "*announce_peer*". These strings correspond to the BitTorrent DHT protocol requests [40]. The destination UDP ports do not correspond to a specific range, with the following ports being the most used: 2905, 13193, 1068, 49514, 6881 and 6882.

As it can be seen, more than 63% of the packets correspond to UDP traffic of BitTorrent, which is difficult to identify without observing the payload, as it uses different ports. After identifying the most used payloads, the number of packets of each of them was observed in the different sensors, so that it is possible to check whether they are used more in active sensors, in passive sensors or independently. The traffic received by each sensor according to the types shown in Table 6 can be seen in Table 7.

Table 7 shows the number of packets received by each sensor for each traffic type in Table 6. Packets that are not of the main types identified are found in the Others column. It can be seen that only some types of traffic are found on some sensors, such as type T4, which is only found on sensor P4, and T5 traffic on sensors A4 and P5. It can also be seen how T5 traffic is found on all sensors with a similar volume of packets associated with connections from multiple origins to all of them. The T1, T2 and T3 types are distributed over more sensors and it can be observed that the sensors where the BitTorrent type traffic (T1) is higher also have a high number of packets in the Others column. This could indicate that there is other traffic, not identified by the payload, that could be related to this protocol. This case is left for future studies.

With the information obtained after these analyses, and looking at the Others column of Table 7, no difference is shown between active and passive sensors. It would be necessary to perform a specific and exhaustive analysis to classify the remaining traffic and to check which services, protocols or scanning tools are affected by the response traffic generated by the sensor. It is not possible to establish a relationship between traffic received and traffic responded to due to the large volumes of BitTorrent traffic and the other types identified, although a more direct relationship is observed between the Others column and the packets responded to in Table 7.

Table 7
Number of UDP packets of the main payload types received by each sensor.

Sensor	UDP Recv	UDP Sent	T1	T2	T3	T4	T5	Others
A1	1,186,618	192,510	10,082	688,295	246,998	0	19	241,224
A2	459,482	118,110	183	24	201,984	0	39	257,252
A3	2,774,934	715,418	1,299,624	600,183	19	0	23	875,085
A4	390,135	76,353	233	109,585	25	0	56,227	224,065
A5	166,191	42,718	168	93,931	22	0	25	72,045
P1	4,842,820	0	4,652,781	25	25	0	43	189,946
P2	233,678	0	91	78,772	125,430	0	28	29,357
P3	1,481,707	0	624,666	103,201	127,745	0	28	626,067
P4	365,676	0	93	25	25	162,397	23	203,113
P5	5,449,428	0	4,432,577	122,088	232,014	0	69,633	593,116
Total	17,350,669	1,145,109	11,020,498	1,796,129	934,287	162,397	126,088	3,311,270

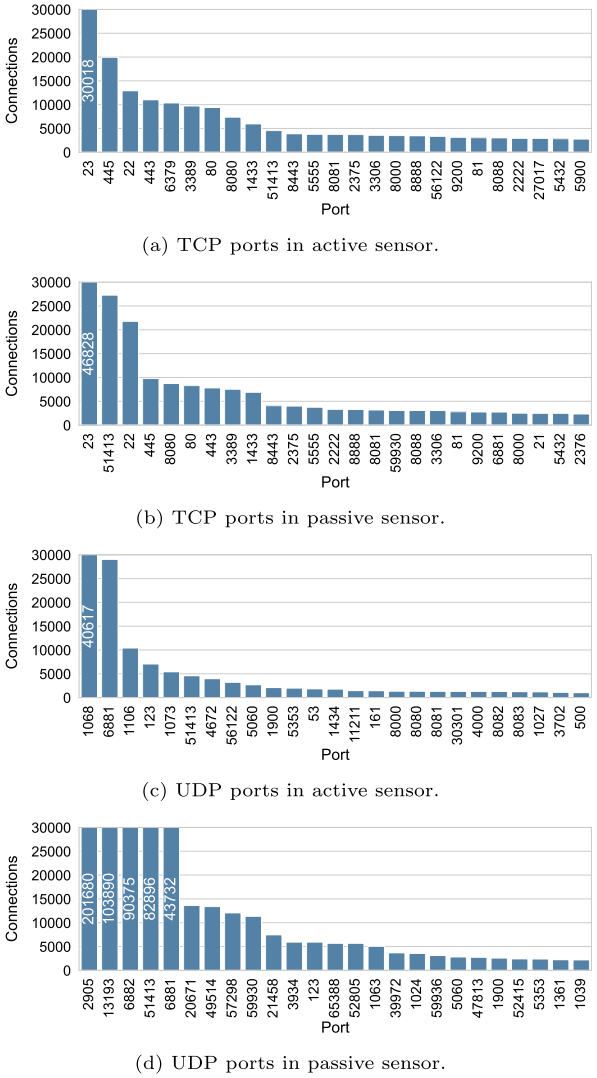
4.3. Exploring IBR traffic port distribution

In order to carry out the analysis of the most used ports, both in TCP and UDP, the volume of traffic received on each port was analysed using active and passive sensors. The results (see Fig. 7) showed that in all cases the distribution of traffic directed to different ports is very diverse and that some of the ports with the highest traffic volume are not commonly used ports. As explained in Section 4.2, this is caused by some source IP addresses generating a high number of packets to a destination port, usually on a single sensor. By increasing the sample period and the number of sensors, the significance of this traffic can be reduced.

Rather than counting the packets received on each port, and thus giving importance to the source nodes that generate large volumes of traffic, it was decided to count unique connections received each day from each different source IP. In this way, the ports that receive connections from several different sources become the most important. By performing this analysis many of the ports with the highest number of packets, which have proven to be the most used, are no longer taken into account, and well-known ports now appear among the most used. The payload of this suppressed traffic has been analysed and most of it corresponds to BitTorrent protocol traffic.

In the case of TCP, the analysed ports are practically the same for active and passive sensors. In both cases they correspond to typical traffic received by other network telescopes [12], such as Telnet ports (23), SSH (22) and HTTP or HTTPS (80, 81, 8080, 8081, 443 and 8443). There are also remote connection ports, such as RDP (3389) and VNC (5900), that allow access to files or data, such as SQL Server ports (1433) and Server Message Block (445) or access to container-based information, such as ElasticSearch (9200) or Docker (2375).

Regarding UDP captured traffic, the well-known DNS (53) and NTP (123) ports are commonly used for amplification attacks [41]. The MemCache port (11211) [42,43] is also used for this purpose. As discussed in Section 4.2.4, most of the traffic received on UDP is related to the BitTorrent protocol. Since the first BitTorrent clients listened on ports 6881 - 6889, a high number of connections are observed in this port range (especially on 6881 and 6882), but nowadays clients randomise the listening port. In this sense, packets captured by passive sensors on the most used ports, as well as port 2905, contain in the payload the text strings "*get_peers*", "*ping*", "*find_node*" and "*announce_peer*", related to this protocol. Other ports used in UDP are SIP (5060) and Web Service Discovery (3702), which is used in several Windows systems.



(a) TCP ports in active sensor.

(b) TCP ports in passive sensor.

(c) UDP ports in active sensor.

(d) UDP ports in passive sensor.

Fig. 7. Histogram of the daily most used TCP and UDP connections from unique IP addresses.

4.4. Exploring known attacks in received IBR traffic

The next analyses carried out correspond to the exploration of known attacks in the captured IBR traffic. There are many attack payloads found in the IBR; three cases are presented as examples, leaving a more comprehensive analysis for later studies.

4.4.1. Apache RocketMQ exploitation

The first of the attack payloads found corresponds to an exploit for a vulnerability (CVE-2023-33246 [44]) in Apache RocketMQ (a distributed messaging and streaming platform). This exploit has been identified in the attacks received by all active sensors in Section 4.2. The exploit displays the following strings:

```
{"code":105,
"extFields":
 {"Signature":"/u5P/wZUbhjanu4LM/UzEdo2u2I=",
  "topic":"TBW102",
  "AccessKey":"rocketmq2"
 },
"flag":0,
"language":"JAVA",
"opaque":1,
"serializeTypeCurrentRPC":"JSON",
"version":401}
```

4.4.2. Download exploits

Multiple exploits have been found in which an attempt is made to execute code downloaded from a remote server on the attacked computer.

In the analysis of TCP traffic, the Huawei Home Device Upgrade RCE Exploit [45] can be found in the packet payload:

```
/bin/busybox \
wget -g 185.254.xxx.xxx -l /tmp/kh \
-r /faith.mips;
/bin/busybox chmod 777 * /tmp/kh;
/tmp/kh huawei
```

This exploit downloads a file from a server to execute it by upgrading the device's firmware. The code is for MIPS architecture, as can be seen from the filename (`faith.mips`).

In UDP, for example, an attempt at code injection can be observed with the following payload:

```
cd /tmp || cd /var/run ||
cd /mnt || cd /root ||
cd /; wget http://195.58.xx.xx/trc.sh;
curl -O http://195.58.xx.xx/trc.sh;
chmod 777 trc.sh; sh trc.sh; rm -rf *
```

The IP address of the node containing the malicious code can be identified in the TCP and UDP payloads. This allows the malicious code to be captured for investigation and analysis.

4.4.3. Reflection DDoS attack

It is also possible to identify Distributed Denial of Service (DDoS) attacks based on amplification and reflection. The following payload contains LDAP traffic (identified by the destination port, 389) with the string "objectclass0":

```
02 01 00 1c 04 00 0a 01 00 0a 01
00 02 01 00 02 01 00 01 01 00 87 0b
objectclass0 00
```

The text string "objectclass0" appears as a rule for identifying the attack "*ET DOS Potential CLDAP Amplification Reflection*" in the emerging DoS detection rules (line 253) of the Suricata IPS (Intrusion Detection System) software [46].

4.5. Summary of results

Throughout this section, various analyses have been carried out to validate the proposed solution.

In Section 4.1, a performance analysis has been carried out to show the viability of the software developed, with the aim of observing whether the distributed network telescope provides satisfactory results. It has been observed that HoDiNT provides quality information even with few sensors and that the option of using distributed sensors in household connections is feasible, but points for improvement in terms of performance have been observed. Based on this result, although the objective of validating HoDiNT is satisfactory and the performance is acceptable, it is proposed as an improvement point to look for alternatives for better performance.

In Section 4.2, a comparison was made between active and passive sensors, showing that the volume of traffic received by the active sensors is much higher than that of the passive sensors. The ability of HoDiNT to change the mode of operation and how the volume of traffic immediately changes when this is done was shown. It has also been shown how, unlike classical telescopes, it is possible to identify patterns of source node behaviour that are only observable when the node is operating in active mode. Furthermore, the distributed network telescope can be used to correlate information from source nodes with similar behaviour for further classification. Moreover, all this has been achieved with a very small number of sensors.

In the same section, for each type of sensor, the statistical information on the traffic volumes received for each transport protocol was observed. An analysis of the most used ports was also carried out in Section 4.3, in line with other network telescopes.

As the main point of improvement in these sections, it would be interesting to have a larger number of sensors to correlate other types of information. In addition, a larger number of sensors would minimise the effects of large volumes of packets generated from one source, which could alter the overall statistics of the network telescope.

The last set of analyses was on the packet payloads in Section 4.4. The Network Telescope has been able to find useful information for cyber intelligence. It would be interesting to be able to automatically classify and process the information received, associate it with CVEs (Common Vulnerability and Exposures) and extract relevant information from the payloads, such as servers with exploits and the exploits themselves. This is left for future work.

5. Conclusions and future work

After analysing the results obtained in this study, it can be concluded that the proposed distributed architecture for the acquisition of Internet Background Traffic is feasible. As demonstrated with HoDiNT, it is possible to build a low-cost sensor network, with minimal use of IP addresses and obtain large volumes of IBR traffic.

By relying on home connections, the sensor network is fully scalable and it is very difficult for the attacker to distinguish between the connections of users without sensors and users with sensors, especially if the sensors are not responding. This prevents the attacker from detecting the existence of this network telescope.

It has also been shown that active (responsive) sensors capture much more traffic than non-responsive sensors. As this is an unusual behaviour for classical network telescopes, it allows traffic to be captured that would otherwise not be possible. It has been shown how two-phase attacks are leveraged by active sensors to obtain additional attacks patterns than those obtained by passive ones.

The differences between network telescopes and honeypots have been outlined and it has been shown that HoDiNT, with its current features of generic responses controlled by packet boundaries, is on the side of network telescopes. However, if necessary for an investigation, it would be possible to extend HoDiNT's response capabilities, bearing in mind that this could affect its performance and also its security, and could require more powerful hardware and more complex software developments.

The use of a distributed model, even with only a few sensors, together with the implemented response system, has made it possible to relate behaviours generated by a single source to all destinations of the network telescope. This is a major difference compared to classical network telescopes, which are unable to observe these behaviours due to a much larger addressing space.

From the analysis of the non extensive dataset of received IBR traffic, it can be seen that most of the traffic corresponds to BitTorrent. Despite this, we have shown that it is still possible to identify attack traces (Section 4.4), which is one of the main motivations for IBR traffic monitoring.

It has been shown how it is possible to identify sources of malicious code by analysing packet payloads. This allows the use of detection measures by analysing the collected information, as well as response measures by filtering the URLs and IP addresses that appear in the collected information.

Future work will address the increase in the number of distributed probes and analyse an extended monitoring time period. A comparison of the detection capability of these distributed probes against traditional network telescopes would be interesting. This would allow for a more detailed exploration of the behaviour of attackers on active and passive probes, as well as the identification of possible networks of attackers performing node and port scans and concerted attacks.

As mentioned in the Introduction (Section 1), classical network telescopes use contiguous addressing, which in many cases generates a large number of similar packets for all telescope addresses, and thus a significant amount of redundant information [2]. In order to extract the relevant information, data sanitisation mechanisms must be implemented to remove the redundant information and extract the relevant information for cyber intelligence [47]. One of the advantages of HoDiNT is that the amount of redundant information is drastically reduced, but it is necessary to establish a correlation model of the information received by the sensors that also allows the extraction of relevant information. It is therefore necessary to create such a model as future work.

While the proposed software has allowed validation of the distributed network telescope model, an optimised implementation that allows the acquisition and response of a larger volume of traffic will be studied as future work.

It is considered interesting as a future work to carry out an automatic classification of the information received, obtaining interesting information on cyber intelligence, such as the relationship with CVEs and the extraction and analysis of payloads.

However, the exploration of IBR traffic on IPv6 has not been included in this initial version of HoDiNT. Nonetheless, it would be worthwhile to consider adding support for this type of traffic, given its current deployment.

CRediT authorship contribution statement

Rodolfo García-Peñas: Writing – original draft, Validation, Software, Investigation, Data curation, Conceptualization. **Rafael A. Rodríguez-Gómez:** Writing – review & editing, Validation, Supervision, Methodology, Investigation, Funding acquisition, Data curation, Conceptualization. **Gabriel Maciá-Fernández:** Writing – review & editing, Validation, Supervision, Methodology, Investigation, Formal analysis, Data curation, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The data that has been used is confidential.

Acknowledgements

To the PEople who have participated by contributing their sensors, connections and time.

This publication has been partially funded by MCIN/AEI/10.13039/501100011033 under the projects PID2020-113462RB-I00 and PID2020-114495RB-I00 and the project NetSEA-GPT (C-ING-300-UGR23) funded by Consejería de Universidad, Investigación e Innovación and the European Union through the ERDF Andalusia Program 2021–2027. Funding for open access charge: Universidad de Granada / CBUA.

References

[1] M. Allman, V. Paxson, J. Terrell, A brief history of scanning, in: Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement, ACM, 2007, pp. 77–82, http://dx.doi.org/10.1145/1298306.1298316.

[2] R. Zhang, C. Yang, S. Pang, H. Sarrafzadeh, UnitecDEAMP: Flow feature profiling for malicious events identification in darknet space, in: L. Batten, D.S. Kim, X. Zhang, G. Li (Eds.), Applications and Techniques in Information Security, Springer Singapore, Singapore, 2017, pp. 157–168.

[3] K.C. Claffy, D. Clark, The 11th workshop on active Internet measurements (AIMS-11) workshop report, 49, (3) Association for Computing Machinery (ACM), 2019, pp. 39–43, http://dx.doi.org/10.1145/3371927.3371933,

[4] R. Hiesgen, M. Nawrocki, A. King, A. Dainotti, T.C. Schmidt, M. Wählisch, Spoki: Unveiling a new wave of scanners through a reactive network telescope, in: Proceedings of 31st USENIX Security Symposium, 2021, http://dx.doi.org/10.48550/ARXIV.2110.05160, arXiv, arXiv:2110.05160.

[5] Center for Applied Internet Data Analysis, 2023, https://www.caida.org/. (Accessed: 17 October 2023).

[6] P. Richter, A. Berger, Scanning the Scanners: Sensing the Internet from a Massively Distributed Network, in: Proceedings of the Internet Measurement Conference, ACM, 2019, pp. 144–157, http://dx.doi.org/10.1145/3355369.3355595.

[7] R. Pang, V. Yegneswaran, P. Barford, V. Paxson, L. Peterson, Characteristics of Internet background radiation, in: Proceedings of the 4th ACM SIGCOMM Conference on Internet Measurement, ACM, 2004, pp. 27–40, http://dx.doi.org/10.1145/1028788.1028794.

[8] Watching the Net's background radiation, 2003, published: 2003-11-27, https://www.theregister.com/2003/11/27/watching_the_nets_background_radiation. (Accessed: 17 October 2023).

[9] D. Moore, C. Shannon, D.J. Brown, G.M. Voelker, S. Savage, Inferring Internet denial-of-service activity, ACM Trans. Comput. Syst. 24 (2) (2001) 115–139, http://dx.doi.org/10.1145/1132026.1132027.

[10] Updated: MS04-011 LSASRV Exploit; Sasser Worm Update: Sasser.b, 2004, published: 2004-05-01, https://isc.sans.edu/diary/Updated+MS04011+LSASRV+Exploit+Sasser+Worm+Update+Sasserb/182/. (Accessed: 17 October 2023).

[11] B. Irwin, A Source Analysis of the Conficker Outbreak from a Network Telescope, SAIEE Africa Res. J. 104 (2) (2013) 38–53, http://dx.doi.org/10.23919/saiee.2013.8531865.

[12] E. Wustrow, M. Karir, M. Bailey, F. Jahanian, G. Huston, Internet background radiation revisited, in: Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement, ACM, 2010, pp. 62–74, http://dx.doi.org/10.1145/1879141.1879149.

[13] X. Meng, Z. Xu, B. Zhang, G. Huston, S. Lu, L. Zhang, IPv4 address allocation and the BGP routing table evolution, SIGCOMM Comput. Commun. Rev. 35 (2005) 71–80, http://dx.doi.org/10.1145/1052812.1052827.

[14] ICAAN News Release: Available Pool of Unallocated IPv4 Internet Addresses Now Completely Emptied, 2011, published: 2004-05-01, https://itp.cdn.icann.org/en/files/announcements/release-03feb11-en.pdf. (Accessed: 03 February 2011).

[15] K. Benson, A. Dainotti, k.c. claffy, A.C. Snoeren, M. Kallitsis, Leveraging Internet Background Radiation for Opportunistic Network Analysis, in: Proceedings of the 2015 Internet Measurement Conference, ACM, 2015, pp. 423–436, http://dx.doi.org/10.1145/2815675.2815702.

[16] CAIDA's Measurement and Data Infrastructure, 2022, https://catalog.caida.org/presentation/2022_caida_measurement_data_infrastructure_overview. Jan 2022.

[17] J. O'Hara, Cloud-based network telescope for Internet background radiation collection, 2019, p. 16, Trinity College Dublin, https://www.scss.tcd.ie/Stefan.Weber/PDFs/Joseph%20O%20Hara%20-%20MCS%20Dissertation%202019.pdf. (Accessed: 17 October 2023).

[18] L. Metongnon, R. Sadre, Beyond telnet: Prevalence of IoT Protocols in Telescope and Honeypot Measurements, in: WTMC '18: Proceedings of the 2018 Workshop on Traffic Measurements for Cybersecurity, 2018, pp. 21–26, http://dx.doi.org/10.1145/3229598.3229604.

[19] The Debogonisation of 2a10::/12, 2020, published: 2020-01-17, https://labs.ripe.net/author/emileaben/the-debogonisation-of-2a1012/. (Accessed: 17 October 2023).

[20] Nmap.org, 2023, https://nmap.org/. (Accessed: 17 October 2023).

[21] Z. Durumeric, E. Wustrow, J.A. Halderman, ZMap: fast Internet-wide scanning and its security applications, in: Proceedings of the 22nd USENIX Conference on Security, SEC '13, USENIX Association, USA, 2013, pp. 605–620.

[22] ZMap: The Internet Scanner, 2017, https://github.com/zmap/zmap. (Accessed: 17 October 2023).

[23] MASSCAN: Mass IP port scanner, 2023, https://github.com/robertdavidgraham/masscan. (Accessed: 17 October 2023).

[24] M.B. Zakir Durumeric, J.A. Halderman, An Internet-Wide View of Internet-Wide Scanning, in: Proceedings of the 23rd USENIX Security Symposium, 2014, pp. 65–78.

[25] D.T. Pearson, An exploration of the overlap between open source threat intelligence and active Internet background radiation, 2020, URL http://vital.seals.ac.za:8080/vital/access/manager/Repository/vital:32299.

[26] F. Bortoluzzi, B. Irwin, L.S. Beiler, C.M. Westphall, Cloud Telescope: A distributed architecture for capturing Internet Background Radiation, in: 2023 IEEE 12th International Conference on Cloud Networking, CloudNet, 2023, pp. 77–85, http://dx.doi.org/10.1109/CloudNet59005.2023.10490018.
[27] E. Pauley, P. Barford, P. McDaniel, DScope: A Cloud-Native Internet Telescope, in: 32nd USENIX Security Symposium, USENIX Security 23, USENIX Association, Anaheim, CA, 2023, pp. 5989–6006, URL https://www.usenix.org/conference/usenixsecurity23/presentation/pauley.
[28] J. Ronan, D. Malone, Revisiting and Revamping an IPv6 Network Telescope, in: 2023 34th Irish Signals and Systems Conference, ISSC, 2023, pp. 1–6, http://dx.doi.org/10.1109/ISSC59246.2023.10162033.
[29] E. d'Andréa, J. François, O. Festor, M. Zakroum, Multi-label Classification of Hosts Observed through a Darknet, in: NOMS 2023-2023 IEEE/IFIP Network Operations and Management Symposium, 2023, pp. 1–6, http://dx.doi.org/10.1109/NOMS56928.2023.10154356.
[30] L. Izhikevich, M. Tran, M. Kallitsis, A. Fass, Z. Durumeric, Cloud watching: Understanding attacks against cloud-hosted services, in: Proceedings of the 2023 ACM on Internet Measurement Conference, 2023, pp. 313–327.
[31] W. Harrop, G. Armitage, Defining and evaluating greynets (sparse darknets), in: The IEEE Conference on Local Computer Networks 30th Anniversary, LCN'05L, IEEE, Sydney, NSW, Australia, 2005, pp. 344–350, http://dx.doi.org/10.1109/LCN.2005.46, URL http://ieeexplore.ieee.org/document/1550875/.
[32] J.-L. Danger, M. Debbabi, J.-Y. Marion, J. Garcia-Alfaro, A. Zincir-Heywood, Foundations and Practice of Security: 6th International Symposium, FPS 2013, La Rochelle, France, October 21-22, 2013, Revised Selected Papers, Springer, 2014, http://dx.doi.org/10.1007/978-3-319-05302-8.
[33] L. Spitzner, The Honeynet Project: trapping the hackers, Secur. Priv., IEEE 1 (2003) 15–23, http://dx.doi.org/10.1109/MSECP.2003.1193207.
[34] M. Nawrocki, M. Wählisch, T. Schmidt, C. Keil, J. Schönfelder, A Survey on Honeypot Software and Data Analysis, 2016, URL http://arxiv.org/abs/1608.06249.
[35] Raspberry Pi Documentation - Processors, 2023, https://www.raspberrypi.com/documentation/computers/processors.html. (Accessed: 17 October 2023).
[36] Raspberry Pi 4 Benchmarked with 32-bit and 64-bit Debian OS, 2020, published: 2020-01-29, https://www.cnx-software.com/2020/01/29/raspberry-pi-4-benchmarked-with-32-bit-and-64-bit-debian-os/. (Accessed: 17 December 2023).
[37] R. Longbottom, Raspberry Pi 4B 32 Bit Benchmarks, 2019, http://dx.doi.org/10.13140/RG.2.2.16864.74244, URL https://www.researchgate.net/publication/333973011_Raspberry_Pi_4B_32_Bit_Benchmarks.
[38] E.C. Rye, R. Beverly, Sundials in the Shade, in: D. Choffnes, M. Barcellos (Eds.), Passive and Active Measurement, Springer International Publishing, Cham, 2019, pp. 82–98.
[39] Common Attack Pattern Enumeration and Classification. CAPEC-295: Timestamp request, 2018, published: 2018-07-31, https://capec.mitre.org/data/definitions/295.html. (Accessed: 30 December 2023).
[40] DHT Protocol, 2020, created: 2008-01-31, last-modified: 2020-01-21, https://www.bittorrent.org/beps/bep_0005.html. (Accessed: 26 January 2024).
[41] M. Nawrocki, M. Jonker, T.C. Schmidt, M. Wählisch, The Far Side of DNS Amplification: Tracing the DDoS Attack Ecosystem from the Internet Core, in: Proceedings of the 21st ACM Internet Measurement Conference, IMC '21, Association for Computing Machinery, New York, NY, USA, 2021, pp. 419–434, http://dx.doi.org/10.1145/3487552.3487835.
[42] 11211 - Pentesting Memcache, 2023, https://book.hacktricks.xyz/network-services-pentesting/11211-memcache. (Accessed: 10 December 2023).
[43] Memcached Server Attacks, 2023, https://westoahu.hawaii.edu/cyber/vulnerability-research/memcached-server-attacks/. (Accessed: 10 December 2023).
[44] NIST (National Information Technology Laboratory), National Vulnerability Database - CVE-2023-33246, 2023, https://nvd.nist.gov/vuln/detail/CVE-2023-33246. (Accessed: 17 October 2023).
[45] Exploit Huawei Home Device Upgrade RCE, 2023, https://github.com/lcashdol/Exploits/blob/master/HuaweiHomeDeviceUpgrade.txt. (Accessed: 17 October 2023).
[46] Alienvault OSSIM source code - Emerging DoS rules, 2017, Line 253, https://github.com/jpalanco/alienvault-ossim/blob/master/suricata-rules-default-open/rules/1.3.1/emerging.rules/emerging-dos.rules. (Accessed: 17 October 2023).
[47] E. Bou-Harb, M. Husák, M. Debbabi, C. Assi, Big Data Sanitization and Cyber Situational Awareness: A Network Telescope Perspective, IEEE Trans. Big Data.

Rodolfo García-Peñas is a Ph.D. student in the Department of Signal Theory, Telematics and Communications at the University of Granada, Spain, and a member of the Network Security and Engineering Group (NESG) research group at this university. His research interests are focused on network security, especially traffic analysis, early detection of new threats and automated mitigation.

Rafael A. Rodríguez-Gómez is an assistant professor of the Department of Signal Theory, Telematics, and Communications at the University of Granada, Spain, and a member of the research group Network Security and Engineering Group (NESG) of this university. Nowadays, his research interest is focused on network security, more specifically on the early detection of new threats and adversarial machine learning attacks and defense methods in the cybersecurity field.

Gabriel Maciá-Fernández is Full Professor at the Department of Signal Theory, Telematics and Communications of the University of Granada (Spain) and researcher in the Information and Communication Technologies Research Centre, CITIC. He received an M.S. in Telecommunications Engineering from the University of Seville, Spain and the Ph.D. in Telecommunications Engineering from the University of Granada. His research interests are focused on system and network security, with special focus on intrusion detection, ethical hacking, network information leakage and denial of service.