Enlightening the Darknets: Augmenting Darknet Visibility with Active Probes

Anonymous Author(s)

ABSTRACT

Darknets collect unsolicited traffic reaching unused address spaces. They bring insights into malicious activities, such as the rise of botnets and DDoS attacks. However, darknets provide a shallow view, as traffic is never answered. We here quantify how their visibility is increased by responding to *some* traffic. To this end, we deploy interactive responders (e.g., honeypots) that can answer unsolicited traffic.

Simple at first sight, determining *how* to answer requires ingenuity: From the selection of the protocol to talk to the risk of polluting the collectors with uninteresting data or saturating the infrastructure. We consider four deployments: Darknets, simple L4-Responders, vertical L7-Responders tied to specific ports, and AcmePot, a new horizontal honeypot that identifies protocols on the fly on any port.

We contrast these alternatives by analyzing traffic attracted by each deployment. We show that interactive responders increase the value of darknet data, uncovering patterns otherwise unseen. We measure *Side-Scan* phenomena in which whenever a host starts responding to a particular port, it attracts traffic to other sometimes random ports. AcmePot unveils attacks that darknets and classic L7-honeypots would not observe, e.g., large-scale activities on non-standard ports. Some strategies, however, trap senders in certain states, thus hindering visibility too. Beyond our findings, our comprehensive analysis can inform the deployment of future monitoring infrastructures combining both darknets and active probes.

1 INTRODUCTION

Darknets or network telescopes are IP addresses advertised by routing protocols without hosting any services. They have been used for years as *passive sensors* in a variety of network monitoring activities. Darknets have supported multiple research activities too [21], and multiple organizations worldwide deploy, analyze, and eventually archive years of data from large darknets for research purposes [7, 14, 50]. Traffic reaching a darknet is necessarily unsolicited. As such, it is helpful to highlight network scans (from both malicious and legitimate scanners) backscattering (i.e., traffic received from victims of attacks carried out with spoofed IP addresses) and traffic due to bugs and misconfigurations [7].

The visibility of darknets is intrinsically limited, as traffic reaching darknets is never answered. Researchers, operators and practitioners have often relied upon external *honeypots* to obtain further information about events seen in darknets [3, 28, 30]. Unlike darknets, honeypots are *active sensors* that obtain information about attacks by answering unsolicited traffic. The goal is to engage with the possible attackers using both simulators that reproduce basic functionalities of real systems (low-interaction honeypots) or actual live system deployed in controlled environments (high-interaction honeypots). Typically, such honeypots are deployed as vertical systems, targeting a particular scenario [33]. For instance, database or terminal server honeypots [11, 18] supporting the respective protocols are deployed on the standard service ports. Only few honeypots try to dynamically determine protocols present in the incoming traffic on-the-fly [22, 27].

Darknets and honeypots are complementary: The first offers a broad but shallow view on ongoing scanning activity touching a wide spectrum of services; The second offers deeper insights about specific cases. Combining "the best of both worlds", achieving at the same time wide coverage and deep insights, could enrich the type of information currently extracted from darknets. Indeed, it is known that darknet traffic changes substantially, not only across the IP address space, but also because of production services hosted "nearby" the darknet [7, 36, 41]. As such, the deployment of active services *inside* the darknet (e.g., using honeypots) could represent a novel perspective to enrich characterization of unsolicited traffic. However, such a deployment also raises many questions: (i) How much *extra information* one would get when responding some unsolicited darknet traffic? (ii) How does the presence of active services change *ports and senders*[1] seen in different deployments? Does the presence of these services affect *neighbouring darknet addresses*? (iii) Does the presence of a specific service attract traffic to *other services*? (iv) What if one answers any service on *non-standard ports*?

Having these questions in mind, this paper presents our journey in quantitatively comparing different levels of interactive responders that we deploy in our darknet address space. In particular, we consider the following four levels:

- Darknet, silent listeners that capture received traffic.
- L4-Responders, that only complete the TCP handshake, saving any possible application-layer requests sent by clients.

[1] We use the generic term "sender" to indicate attackers, scanners, or even victims of attacks that send traffic to us.

- L7-Responders, low-interaction honeypots that mimic specific application protocols that are expected on the usual well-known ports.
- AcmePot, a novel responder that searches for the application that is most suited to handle the incoming traffic, regardless of the probed port.

We deploy these alternatives in our darknet space and capture data for one month: in our measurements, we observe traffic volumes on the order of *hundreds* (on the darknet) to *millions* (on AcmePot) of flows per hour, collecting more than 1 *billion* flows overall. Digging into this dataset, we show that active responders can increase the value of darknet data and provide a detailed quantitative view on the behavior of bots and scanners when facing responders with different interaction levels. In some cases, we revisit and update well-known facts about darknet/honeypot deployments. Most importantly, we gather novel insights that highlight the benefits and drawbacks of the alternative response strategies. Summarizing our key findings:

- We measure and quantify *Side-Scan* phenomena, in which a host responding on a particular service sees also a surge of traffic for other services and ports. In contrast to [36], who observe similar patterns in CDN nodes, we quantify how the type of service one deploys in the darknet decisively influences *Side-Scan*.
- Activating responders leads to a surge of traffic on darknet neighboring IP addresses too. The joint use of active probes and passive darknets therefore leads to an overall picture that is different from what is obtained by a pure darknet deployment, even in the addresses remaining dark.
- L4-Responders augment the visibility of darknets, triggering senders to send additional traffic that uncovers malicious activities in some cases. Interestingly, the lack of application-layer response, while limiting the interaction, also uncovers events that may get unnoticed on classic honeypots, such as "door-knocking" attempts.
- L7-Responders engage senders and attract hundreds of times more traffic to the darknet. Deploying different L7-Responders to cover specific service categories helps to increase the *Side-Scan* phenomena, thus augmenting the value of darknet data.
- Thanks to AcmePot ability to decouple services from ports, we shed light on large-scale activities directed to non-standard ports, offering a rich picture that is unseen in other deployments. However, it may trap senders on some in particular activities, saturating them and thus limiting visibility. This trade-off suggests a complementarity between completely passive and totally interactive deployment.

Aside from these findings, we make several other technical contributions as follows. First, we offer insights to guide the community in the deployment of augmented monitoring infrastructures that systematically combine darknets and active responders, such as honeypots. Second, we contribute the complete set of responders used in our analysis to the community as open source software. In particular, we release AcmePot to foster its development and usage. Finally, to allow reproducibility, we make all data evaluated in the following sections[2] available online in anonymized form.

After providing an overview of related work (Sec. 2), we detail our design and deployment (Sec. 3. We then provide a generic overview of the macroscopic changes we observe (Sec. 4), then we investigate the changes in the contacted ports (Sec. 5.1) and senders contacting the different deployments (Sec. 5.2). We then drill down on the benefits of AcmePot (Sec. 6) before final discussions and conclusions (Sec. 7).

2 RELATED WORK

2.1 Darknet research and infrastructure

Darknets have been employed for years in several network monitoring and research activities [21]. Examples include the study of (i) DDoS attacks [20, 28, 31], (ii) IPv4 address space utilization [13], (iii) Internet censorship [15], (iv) Internet scans [14, 19, 35] and (v) the investigation of malware spread [42].

In only few cases, darknet traffic has been used in isolation to characterize particular phenomena [19, 20, 40]. However, darknet traffic alone misses a comprehensive view to characterize events. Prior work [3] combined darknet traces and external sources, including honeypots, to characterize the Mirai botnet. A similar strategy has been employed before for the characterization of the Sality botnet [14]. These cases emphasize how data collected by additional probes can enrich darknet viewpoint. Here, we take a step further and observe the impact of bringing responders with increasingly sophisticated interactivity abilities inside the very same darknet space.

In terms of darknet infrastructure types, previous efforts have characterized the differences between centralized and sparse deployments, size and location of darknets [21, 50]. Several players maintain darknet infrastructures, including the decades-old project run by CAIDA/UCSD [10], darknets operated by large network operators [7, 50], and other projects run by universities and security companies worldwide [5, 12, 23, 30, 41].

More recent work [36] profited from servers of Akamai's Content Delivery Network (CDN) to study unsolicited traffic. In contrast to a classic darknet, CDN nodes offer public services, thus receiving and processing production traffic.

[2]The repository is currently private to preserve authors' anonymity.

Yet, all TCP/UDP ports not hosting any production services may still be reached by unsolicited traffic. Authors show that the production servers attract unsolicited traffic that is far different from what is observed in an ordinary darknet. We here extend these findings, exposing and studying various deployment combinations and services for which we offer interaction. While our deployment is based on honeypots, thus missing components of a production environment, we show that the combination of services exposed in a host matters, shaping the mixture of attacks and noise that targets it.

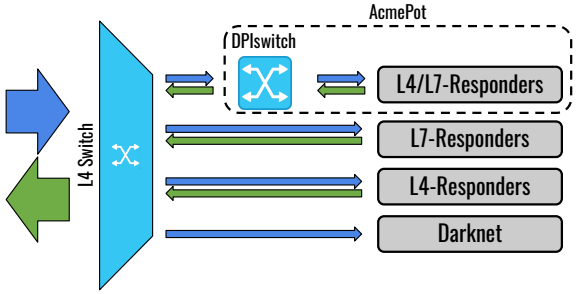


Figure 1: Infrastructure architecture overview.

2.2 Honeypot systems and analysis

We study the impact of deploying active services in the darknet address space using honeypots as responders. Honeypots have been used in security activities for years, with well-established projects such as the Honeynet Project [26] and TPot [45] providing multiple alternatives.

Previous efforts using honeypots have covered many aspects, such as (i) introducing new honeypots targeting particular protocols or services [16, 29], (ii) evaluating the effectiveness of different types of honeypots [24], and (iii) presenting techniques to uncover the presence of honeypots[32, 47]. Readers are invited to check this survey [33] showing a broad overview on honeypot research.

Some authors present general characterization of honeypot traffic, focusing on origin of attacks, targeted services and frequency of attacks (e.g., [2, 29, 44]). A recent work [43] compared the deployment of honeypots in different geographic locations. Another work [9] allocated unused addresses in a cloud to deploy honeypots. These works however evaluate honeypot deployments in isolation, without comparing measurements with what is observed in dark spaces.

A seminal work dating back to 2004 [51] introduced iSink, a monitoring system that can answer darknet traffic. The authors observed traffic peaks due to the presence of responders (e.g., NetBIOS and SMB) in the dark space. Later, another work [6] characterized honeypot traffic and discussed whether bots are sensitive to the "liveness" of hosts during scanning. The authors found that in around 16.3% of the scan events, there was evidence of liveness-awareness. Our work revisits these studies, reappraising the question on the deployment of active responders on darknets. Moreover, we verify how measurements from different active responders differ from (and influence) measurements collected on darknets.

Our AcmePot leverages deep packet inspection to select the best protocol used to answer incoming traffic. It is thus instrumental to identify attacks on non-standard ports. Several *meta-honeypots* allow flexible configuration of backends to handle traffic on non-standard ports [11, 22, 45]. Most of these systems act as proxies (at different layers), logging the traffic that is passed on to backends. They however lack AcmePot mechanisms to identify traffic on-the-fly, and a variety of supported protocols.

Honeytrap [27] is the honeypot closest to AcmePot. Honeytrap is a meta-honeypot that performs protocol identification. It however implements a limited number of protocol fingerprints. Honeytrap supports around 26 services, offering the possibility of extending the set of rules that identify protocols. AcmePot instead relies on a state-of-the-art DPI library [17] that is widely used on other networking applications, offering 100s of protocol fingerprints.

3 METHODOLOGY AND DATASETS

3.1 Infrastructure

Fig. 1 schematizes our measurement infrastructure. Unsolicited traffic that reaches our dedicated address space is routed to one of our four *deployments* that correspond to different levels of interactivity:

(1) Darknet: IP addresses that just receive traffic without replying to any packet;
(2) L4-Responders: responders that complete the TCP three-way handshake, capture eventual application requests from clients, but never reply to any message;
(3) L7-Responders: honeypots that mimic popular application-layer services. We deploy state-of-the-art honeypots to simulate well-known services; L7-Responders act as vertical responders that interact only on a limited set of ports and services;
(4) AcmePot, our novel responder that performs L7 switching of requests using deep packet inspection. It decides on-the-fly which application protocol to use, answering incoming TCP connections on all TCP ports. Differently from L7-Responders, AcmePot decouples standard TCP ports from application protocols.[3]

[3]To avoid resource starvation, both our L4-Responders and AcmePot implement active and inactive timeouts, dropping active (idle) connections after 60 s (10 s).

Table 1: Deployments, categories with their typical ports and applications. All numbers refer to traffic of a /29 network during a full month. For direct comparison, we report numbers only for the first /29 used as darknet. Numbers marked in bold represent clear anomalies.

Deployment	Category	Category: Port	Category: Application	Flows	Flows with L7 Payload	Dest. Ports	Sender Addr.
AcmePot	*All*	0:65535	*All below*	**1214 M**[1]	**683 M**[1]	**49 864**[2]	75 k
L7-Responders	*All*	*All below*	*All below*	17 M	12 M	**64 919**[2]	94 k
	Database	3306, 33060,* 1433, 4022,* 1434,* 5432,* 27017	mysql, mssql, postgres, mongodb	3 M	166 k	65 535	70 k
	Fileserver	135:139, 445	netbios, CIFS	5 M	2 M	65 535	68 k
	Mail	25, 110, 143, 465, 993, 995	pop(s), imap(s), smtp(s)	3 M	51 k	65 535	69 k
	Proxy	8080, 8000,* 3128	generic, squid	3 M	24 k	65 535	70 k
	Remote Desktop	3389, 5900, 5901, 5800,* 5801,* 5938,* 6568*	ms rd, vnc, teamviewer, anydesk	8 M	4 M	65 535	70 k
	Terminal	22, 2222,* 23, 2323*	ssh, telnet	6 M	3 M	65 535	82 k
	Web	80, 443	http(s)	3 M	39 k	65 535	73 k
L4-Responders	*All*	0:65535	-	8 M	3 M	**49 777**[2]	87 k
	Database	3306, 33060, 1433, 4022, 1434, 5432, 27017	-	3 M	294 k	65 535	71 k
	Fileserver	135:139, 445	-	3 M	742 k	65 535	69 k
	Mail	25, 110, 143, 465, 993, 995	-	3 M	24 k	65 535	71 k
	Proxy	8080, 8000, 3128	-	3 M	22 k	65,535	70 k
	Remote Desktop	3389, 5900, 5901, 5800, 5801, 5938, 6568	-	3 M	268 k	65 535	90 k
	Terminal	22, 2222, 23, 2323	-	4 M	292 k	65 535	71 k
	Web	80, 443	-	3 M	34 k	65 535	77 k
Darknet	*None*	0:65535	-	2 M	0	65 535	63 k

(∗) Ports that are forwarded to the L7-Responders, even if the backend (i.e., TPot) does not host any honeypot. The L7-Responders reset the connection in these cases, as opposed to the darknet (which never answers traffic) and the L4-Responders (which always try to open a connection request).
[1,2] Anomalies discussed in the following.

Note that none of our deployments answers to UDP or malformed TCP packets for preventing abuses (see ethics concerns in Section 3.4).

For the L7-Responders deployments, we rely on the honeypots organized and distributed by the TPot project [45] that act as backend. We activate honeypots to handle a range of popular application protocols. TPot offers a collection of third-party *low-interaction* honeypots, i.e., programs crafted to simulate a vulnerable service communicating over an L7 protocol. Most of our L7-Responders offer login interfaces only [25], registering the brute-force attempts against services (e.g., RDP, POP3 and IMAP). Some L7-Responders rely on more sophisticated honeypots, e.g., simulating a vulnerable server accessible via SSH/Telnet [11], or serving pages that mimic actual services accessible over the web [38]. We defer the reader to the documentation of TPot for details.

Our L7-Responders offer *vertical services* only: They are deployed behind the standard TCP ports of the given service, e.g., the HTTP honeypot is deployed on port TCP/80 whereas the Remote Desktop Protocol (RDP) honeypot responds to port TCP/3389. To study the impact of answering to traffic arriving on other ports, we implement and deploy AcmePot. AcmePot listens to *all* TCP ports. On receiving a new TCP connection request, it completes the three-way-handshake and waits for the first message from the client. It then analyzes the payload looking for the application-layer protocol. AcmePot relies on `nDPI` [17]. This choice lets us obtain a flexible system, supporting hundreds of protocols, which is far more than what is supported in previous projects [27]. If a known protocol is found and one of the L7-Responders can handle it, AcmePot steers traffic to such backend; otherwise it acts like L4-Responders. Note that AcmePot can only identify and steer traffic for cases that are *client initiated*, i.e., where the client sends the first application-layer message. Otherwise, it behaves like L4-Responders– e.g., in telnet or SMTP, where the client waits for the server banner before attempting to login.

Both our L4-Responders and AcmePot are implemented in Python using the Twisted framework [46]. Our architecture (see Fig. 1) is intrinsically distributed, and Twisted is scalable. However, as we will show later, the deployment of responders increases the traffic reaching the darknet by orders of magnitude. To prevent abuses, we thus intentionally limit the capacity of our infrastructure (see Section 3.4).

3.2 Deployments and categories

Inside a regular /16 network that hosts servers and clients, we isolate one /23 network to perform experiments with our multiple *deployments* – i.e., darknet, L4-Responders, L7-Responders or AcmePot– in equal conditions.[4] We split the /23 network into /29 networks, each with 8 IP addresses. We dedicate sixteen /29 networks (all belonging to the same /24 network and hosting 8 identical responders each) to L4-Responders and L7-Responders, answering to a specific service *category*. We configure 8 categories for L4-Responders and 8 for L7-Responders (see Table 1). The

[4]Further details about the darknet are omitted for keeping authors anonymous.

category defines which services the responder supports. We configure the responders to receive and handle only traffic that arrives to ports typically hosting services belonging to such category, silently dropping packets arriving on other ports (e.g., as done by some firewalls). We create categories for database, file, mail, proxy, remote desktop, terminal, and web services. We report all ports opened for each category on Table 1, together with some typical applications relying on such ports. We also create an *extra* category denoted as *All*, for which we accept all traffic going to any port. In the case of the *all* category in L4-Responders, we perform TCP handshake for flows arriving in any TCP port. For the L7-Responders category denoted as *All*, we pass all traffic to the TPot backend, regardless on whether there is a honeypot active on that port or not: if no honeypot is present, the backend explicitly resets the connection.

We devote a /29 network to host AcmePot, which answers to all ports by definition. It performs DPI on the arriving packets to identify the most appropriate responder based on the payload, and eventually forwards traffic to a honeypot offered by TPot. The remaining IP addresses in the /23 network operate as a classic darknet. In particular, we configure an entire /24 network as a darknet. Unless explicitly mentioned, all results using a darknet as baseline refer to such addresses. The second /24 network partly hosts responders and partly acts as a darknet – these latter addresses are used in Sec. 5 to understand the impact of active services on neighbour darknet addresses.

3.3 Data capture and processing

Our infrastructure captures all packets hitting the /23 darknet. We use `tcpdump` and store all traces on a high-end server, generating separate logs for each deployment.

We here characterize the traffic focusing on TCP flows, defined by the usual 5-tuple (client/server IP addresses, client/server ports and transport-layer protocol). A new flow starts when a `SYN` segment is received, and it terminates after the connection is closed (in case of the active responders) or an idle time. We annotate each flow with useful metadata and statistics, including the application protocol identified by `nDPI`, if any L7 payload is present. Table 1 shows an overall traffic breakdown for each of our deployments and categories.

According to the capabilities of each responder and the behavior of remote machines, we identify different *flow stages*:

- `SYN`: Flows for which we observe only the `SYN` message(s), eventually retransmitted by the client multiple times; This is the most common case on darknets, but it happens also on the blocked ports of other deployments or when a responder is unable to cope with the workload;
- `2WH`: Incomplete three-way handshake, where the client ignores (or resets) the `SYN/ACK` message, as in the case of stealth-`SYN` port scans;
- `3WH`: Client and server complete the TCP three-way handshake, but exchange no payload – this is expected in L4-Responders and AcmePot when clients wait for servers to initiate the conversation;
- `L7 payload`: Client and server open the TCP connection and exchange some application-layer messages.

In addition, we also record malformed TCP messages, e.g., `SYN/ACK` likely arriving due to backscattering or other packets with bogus TCP flags, as well as any other protocol (UDP, ICMP, etc). These cases are however not discussed in the paper, but included in the public traces we release to the community. Here, we analyze data captured over one month, from the 15^{th} of April to the 15^{th} of May 2021. In total we collected about 115 GB of traffic, corresponding to more than 1 300 millions flows coming from more than 600 000 unique IP addresses.

3.4 Ethics

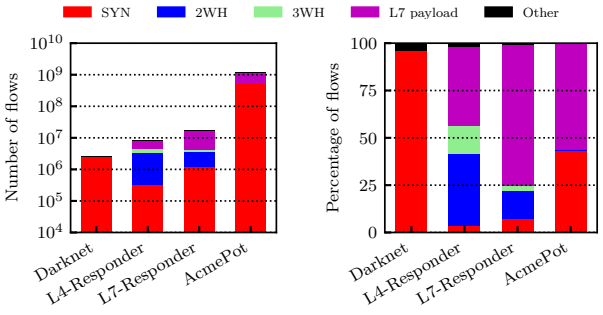
We take several countermeasures to restrict the impact of our measurements on third-party networks. First, and most important, we never answer packets if our packets may worsen the position of attack victims. In particular, we never answer UDP traffic, as it could make our infrastructure part of DDoS attacks relying on spoofed addresses and amplification techniques. For the same reason, we silently drop all TCP packets with `SYN/ACK` flags and other malformed flows, as they may arrive from victims of DDoS attacks with spoofed addresses. Answering such packets may help the attackers to overload the victims' networks.

The traffic that we answer may come from infected machines that are taking part in botnets. As previously said, we explicitly limit the capacity of our infrastructure to avoid creating too much traffic for the networks hosting such infected machines. The setup discussed in this paper can comprehensively sustain at most some few Mbps of traffic upstream, which is far insufficient to overload remote networks.

Finally, IP addresses sending traffic to our infrastructure may uncover vulnerable computers exploited by attackers [39]. We take all measures to protect such IP addresses. In the datasets we release, we anonymize addresses. We also collaborate with our security team and our upstream providers, actively notifying about novel attacks and senders.

4 MACROSCOPIC CHANGES IN TRAFFIC

In this section we report a high-level characterization of the different deployments to answer our first question, i.e., how much extra information one would get when starting to reply to unsolicited traffic. For the sake of comparability, we here



(a) Flows (notice the y-log scale) (b) Percentage

Figure 2: Flows reaching different deployments.

restrict our analysis to 8 addresses per deployment, focusing on those answering to the *all* category.

4.1 Breakdown per flow stage

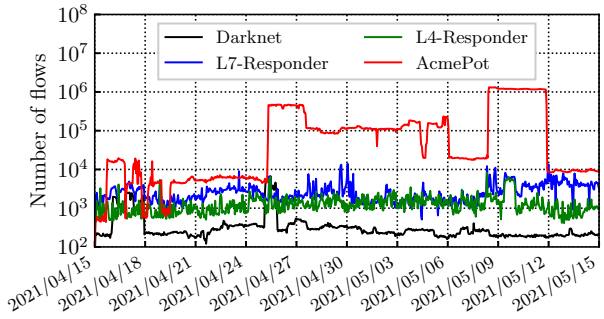
Fig. 2 reports the number of traffic flows received in each deployment, breaking it down per flow stage. The left plot details the number of flows (notice the *y*-log scale) while right plot details the share in each deployment.

The darknet observes a large majority of TCP SYN messages, with a few UDP and bogus TCP segments (about 8% of the total). As soon as we start replying with the L4-Responders, the number of flows grows by a factor of 4 compared to the darknet[5] (cfr. Table 1 too). Thus, although our deployment shall perform the full 3-way handshake, some flows remain in the SYN stage, i.e., connection requests to which our L4-Responders deployment cannot reply due to short-term congestion. Their share is marginal as visible in the percentages in the right plot. Interestingly, 35% of the flows terminate at the 2WH stage, most likely corresponding to "host discovery" scans with crafted SYN messages or SYN messages with spoofed IP addresses. About one forth of the open TCP flows carries no payload, i.e., likely host discovery actions performed with TCP-connect scan.

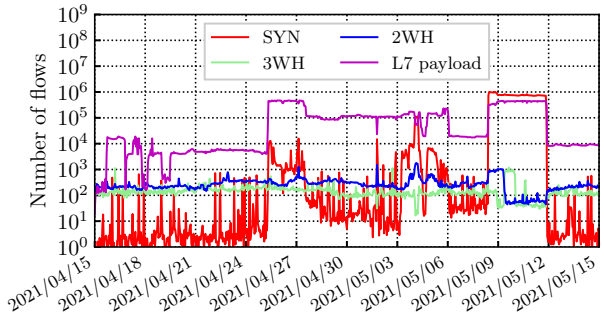
Moving to the L7-Responders, the number of flows doubles again. The SYN stage flows are now about 7%. Part of this traffic is again caused by the limits we impose on our infrastructure (Sec. 3.4). However, as we will zoom in later in Sec. 6, once we answer traffic in some ports, more scans are observed in other ports too. This effect increases the number of SYN-stage flows. Naturally, we observe a strong increase of `L7 payload` flows, which are now about 75% of the total.

Finally, moving to AcmePot, it attracts 3 and 2 orders of magnitude more flows than the darknet and the L7-Responders, respectively. The number of flows grows

[5]As we will show in Sec. 6, comparing two subsets of darknet addresses yields a stable number of flows



(a) Darknet, L4-Responders-All, L7-Responders-All, AcmePot



(b) AcmePot flow stage breakdown

Figure 3: Temporal evolution of the number of flows.

to billions – about 70 times more than in the L7-Responders, and 600 times more than in the darknet. Here we see around 40% of cases finishing on SYN stage, which correspond to periods in which our deployment hit the rate limiter. That is, the limits we impose on our infrastructure capacity play a crucial role, likely reducing the traffic attracted by AcmePot.

It is worth commenting that the share of `Other` traffic remains similar in all deployments. This suggests that answering TCP traffic as we do in our deployments does not stimulate senders to generate packets using UDP/ICMP.

4.2 Temporal evolution

Darknets and honeypots are known to receive variable numbers of flows over time. Our setup is not different: Fig. 3(a) reports the average per-hour number of flows received by each deployment (*All* category again). Notice again the y-log scale. As expected, the darknet is steadily the least contacted deployment with a few hundreds of flows per hour on average, except during sporadic massive scans hitting the address space [7, 28, 41]. Both L4-Responders and L7-Responders show a noisier pattern over time, again with small episodes of

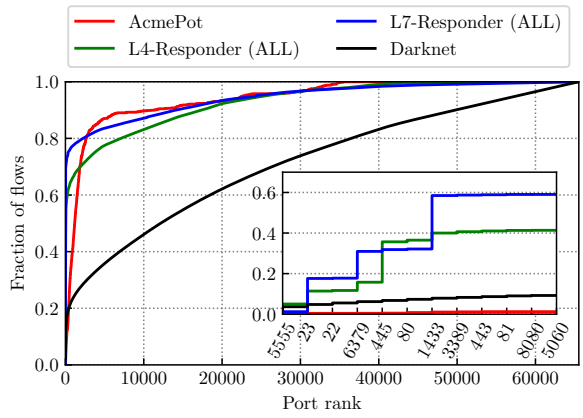


Figure 4: Flow distribution per destination port. Ports are ranked according to the received traffic volume in the darknet. Zoom on top-12 ports.

increases, sometimes affecting multiple deployments simultaneously (as in April 25th), sometimes being uncorrelated (as in May 7th).

The AcmePot registers much more variable figures. For instance, flows per hour top to more than 1 million on May 7th to suddenly vanish on May 12th. We will detail this case in Sec. 6.3. As said above, these episodes bring AcmePot to the limits we impose on the infrastructure. In Fig. 3(b) we break down the hourly number of AcmePot flows according to the flow stages. We see for example that the number of SYN stage flows is usually negligible. Yet, it increases together with the overall traffic. This is evident during the sudden growth in the May 7th – May 12th interval, where we see a plateau of around 350 000 flows per hour per IP address for `L7 payload` stage.

This explains the first anomaly in Table 1: AcmePot manages to answer only 683 million flows out of potentially 1 214 million flow requests.

Takeaway: The number of flows grows by orders of magnitude with increasingly interactive responders. Vertical honeypots attract 10 times more flows than darknets. AcmePot pushes this increase to 600 times, thanks to its ability to answer application traffic on non-standard ports. This growth creates temporal bursts of traffic that challenge the deployment itself and call for protection mechanisms to avoid collapsing the infrastructure and biasing the collected data.

5 PORTS, SENDERS AND NEIGHBORS

In the previous section, we have seen the effects of answering darknet traffic in terms of traffic volumes. We now assess changes on traffic patterns along two axes: the targeted services (i.e., probed ports) and traffic sources (i.e., IP addresses of senders). This section answers our second question, how the presence of active services changes ports and senders, and whether these deployments affect neighbouring darknet addresses. For this purpose, we drop the temporal dimension and analyze the entire aggregate of one-month of data.

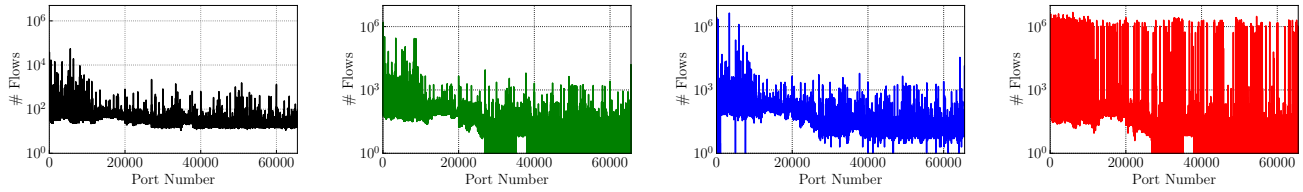
5.1 Changes on probed ports

The traffic volume is not evenly distributed over the exposed ports. Already in the darknet, well-known ports are heavily requested while the rest of the ports receive much less attention, likely part of horizontal scans. We assess how such a distribution changes as we increase the interactivity levels.

Fig. 4 reports the cumulative fraction of flows directed to each port, ranked by port popularity (in terms of traffic volume) in the darknet. Focus on the darknet (black curve). A handful ports receive 20% of flows. Then, the share grows almost linearly to cover the whole port range. This plot highlights how senders spend most of the traffic to perform horizontal scans against darknets, i.e., doing host-discovery.

When we start answering requests, the picture drastically changes. In L4-Responders, the top ports account for more than 60% of the flows. This percentage grows to more than 70% in L7-Responders. That is, once a target is discovered, senders activate the next stages of scans or attacks. This is clearly visible in the inset in Fig. 4, which details the share of flows in the top-12 ports. For instance, observe the increase at ports 23 (Telnet), 445 (SMB) and 3389 (Remote Desktop). These ports attract way more traffic once opened (L4-Responders), and even more if they expose actual services (L7-Responders, where we confirm that most activity is related to password brute-force attempts). Surprisingly, the pattern is not repeated for port 5555, which is often searched due to Android Debug Bridge services. This is the most popular port in the darknet, and we do offer a honeypot on the port [1]. Apparently, either the honeypot does not behave as expected by senders, or the service is not currently receiving attention.

Moving to AcmePot, we observe another different picture. Some hundreds of ports get most of the flows, but the remaining ports also gets some uneven distribution of traffic. Unlike the darknets and L7-Responders, more than 15 000 ports never received any flows in the one-month time period, for both AcmePot and L4-Responders. Recall that, for both cases, all ports result open during port scans. *This behaviour corresponds to the second type of anomalies reported in Table 1. We conjecture that either senders get busy performing other activities on the already found open ports, or they are more cautious and abort (or time out) scans after finding a high number of open ports.*



(a) Darknet (b) L4-Responders (c) L7-Responders (d) AcmePot

Figure 5: Number of flows per destination port.

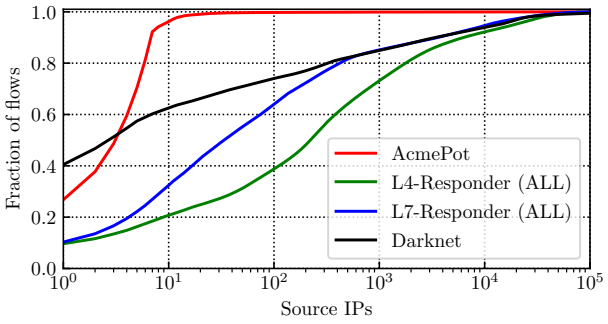


Figure 6: Fraction of flows per sender IP address.

Fig. 5 clearly depicts this behaviour. For each deployment, we report the number of flows per port, ordered by port number. L4-Responders and AcmePot miss some ports starting from port 27000 (number of flows goes below 1 in the y-log scale). Curiously, notice the continuous group of ports [35000 : 38000] where senders again check all ports. For the sake of completeness, note that few ports go unchecked also in L7-Responders.

Fig. 5 illustrates also the frequency in which ports are visited. As expected, senders usually concentrate their interest on well-known ports below 1024 for darknet, L4-Responders and L7-Responders. On the contrary, AcmePot attracts way more flows on very uncommon ports. Investigating the `L7 payload`, these flows carry Remote Desktop Protocol (RDP), hinting for a specific attack. We analyze this case in details in Sec. 6.

Takeaway: Answering to some ports engages senders, which activate the next stage in their scans or attacks. This increases the traffic and sometimes challenges the monitoring infrastructure. Answering all ports traps senders in some activities, possibly limiting/biasing their activity.

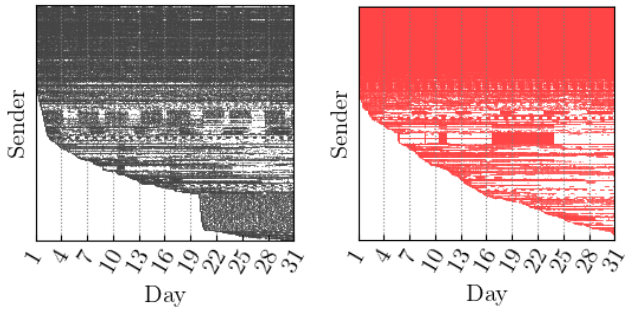
5.2 Changes on traffic senders

We now investigate changes seen in the set of senders contacting the deployments. Fig. 6 reports the total fraction of flows from each sender IP. The x-axis reports with a log scale each sender IP ranked according to the volume, for each deployment.

In the darknet, the most three active sender generates 40% of flows. These are well know scanners reported multiple times in blocklists. The top-10 most active responsible of 63% of total flows. Some of them are sending some hundreds probes per hour for the whole time. Some come, probe at high rate (hitting 20 000 flows per hour) and disappear after completing their job. These are also reported multiple times. At last, we observe a tail of more than 63 000 IP addresses. This is inline with previous work [7] that shows darknet traffic is dominated by bugs, misconfiguration and backscatter, with only a minority of senders actually performing scans and attacks. Comparing to L4-Responders and L7-Responders, we observe that flows are more distributed across senders, with the top-10 of them accounting for 20% and 32% of traffic only. Still, the total number of IP addresses is comparable, reaching 87 000 and 94 000. Looking at the tail of the distribution, we observe a lot of senders which exchange few flows with our deployments.

The figure is completely different in AcmePot where the top-10 most active senders account for 95% of all flows. By checking, these senders are involved in the RDP abuse observed on non-standard ports. They generate millions of flows per hour, triggering the rate limiting countermeasures we implemented in AcmePot (cfr. Fig. 3(b)).

Fig. 7 offers a visual representation of the activity of the top-1000 most active senders over time. Each row corresponds to a single IP address. A dot is present if that IP address is active in that hour. We register the presence of senders that are active most of the time; And the continuous arrival of new senders. For instance, the group of 200 new senders appears in the day 20 in the darknet. These are likely bots that perform a coordinated scan reaching our address space. In general, we can distinguish different patterns: some senders are persistent, while others keep coming back periodically. A few interact only for some time before disappearing and never coming back. The latter is more visible in Fig. 7(b) for AcmePot rather than the darknet in Fig. 7(a)



(a) Darknet (b) AcmePot

Figure 7: Activity pattern of top-1000 sender IP addresses. Each row corresponds to a sender IP address.

Table 2: Jaccard Index of senders hitting different deployments.

	Dark/same	L4-Resp	L7-Resp	AcmePot
Dark/other	*0.128*	*0.142*	*0.137*	*0.143*
Dark/same	–	0.366	0.341	0.420
L4-Resp	–	–	0.429	0.396
L7-Resp	–	–	–	0.372

This sudden and bursty activity patterns are typical of being part of botnets that perform their scan and attacks to then move on the following victims.

Next, we compute the Jaccard indexes to check if senders we observe in different deployments are the same. Given two deployments, we extract the set of IP addresses seen in each of them for the whole period, and compute the ratio between the intersection of the sets over the union. Table 2 details the results, where we further distinguish between the darknets in the *same* /24 address space as the deployments, and the *other* /24 network. The latter distinction allows us to answer the question of whether the presence of responders impacts the neighboring addresses.

It turns out that activating responders in the same /24 address space "pollutes" the darknet addresses, which get contacted more frequently by the same senders that reach our responders. Indeed, the Jaccard index shows that 34-43% of IP addresses are in common between the responders and the darknet sharing the same address space. Conversely, less than 15% of senders seen in the *other* darknet are also seen in the responders. This suggests that senders involved in the initial scan phase are not the same senders involved in the subsequent phases. Activating responders thus increases the visibility on senders. The low value of the Jaccard index is due to occasional senders hitting our address space - see the tail in Fig. 6 - due to backscattering, misconfigurations, and bugs [7].

5.3 Who does what?

At last, given the diversity of the destination ports and the sender IP addresses, we quickly investigate what attack or scan patterns the top-100 most active senders perform in each deployment. For the sake of brevity, we report only the simplest and the most complex and diverse scenario of the darknet and the AcmePot. Fig. 8 plots the activity pattern. The x-axis reports the destination port of flows, sorted by value. Each row refers to a single sender IP address, ordered by IP address, i.e., putting nearby addresses starting with the same initial octet. A dot is present if that IP address sends a flow to such port. The darker the color, the larger the amount.

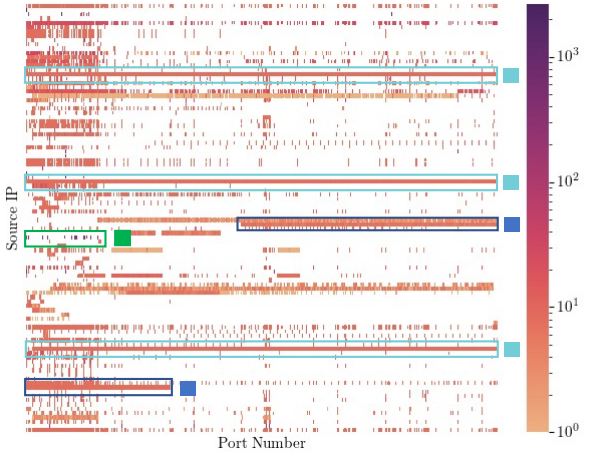
Check first Fig. 8(a) for the darknet. Clear patterns emerge - some we highlight using colors. First, some few horizontal scans are visible (cyan), with senders checking all ports. Second, we observe some vertical scanners (green) - i.e., addresses that target only few ports. Third, some senders target a large subset of ports, stopping the scan after some thousands of probes (dark blue).

Move now to Fig. 8(b) for AcmePot. More evident patterns emerge. Besides the horizontal scanners (cyan) (which however do not probe all ports anymore - cfr. Fig. 5(d)), very targeted scans are visible on very few ports with up to thousands of flows (green). These are vertical attacks that target well-known ports and services. Not shown here, they are present in L7-Responders too. A large number of coordinated scanners emerge too, i.e., senders that target the same few thousands ports (most of which in the [1:1024] range), and fewer selected ports higher than 1024 (black). These are those senders involved in the RDP exploit on non standard ports. At last, some smaller group of scanners appear (dark blue); each scanner probes a different subset of ports, splitting the port range in a coordinated effort.

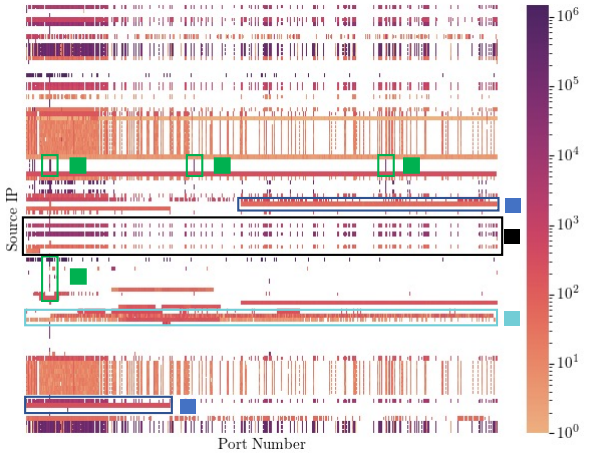
Takeaway: Most of traffic comes from few thousands senders that are involved either with vertical or horizontal scans and attacks. IP addresses involved in darknet scans are typically different from those seen in the later attack stages. And backscattering generates occasional senders that pollute senders sets. Unlike vertical honeypots, AcmePot lets us observe new scan patters where also non-standard ports get the attention of attackers.

6 AMPLIFICATION OF SERVICE-SPECIFIC DEPLOYMENTS

Here, we focus on the extra information, L4-Responders, L7-Responders and AcmePot offer compared to darknets. By evaluating the traffic amplification of the several deployments, we answer our last two questions, i.e., whether the presence of specific services attracts traffic to other services, and what happens when one starts answering to non-standard ports.



(a) Darknet



(b) AcmePot

Figure 8: Flows from top-100 sender to destination port. Addresses are sorted numerically.

6.1 Service amplification

To quantify the extra traffic per deployment, we define the *amplification factor* as the ratio between the number of flows seen on the 8 IP addresses of a specific port(s), and the number of flows directed to the same port(s) on the 8 IP addresses belonging to the darknet.

First, we run a preliminary test to verify whether the amplification factor changes when comparing IP addresses belonging to the same deployment. For this, we take all /29 darknets in the single /24 address space, and compute – for each destination port – the amplification factor for each darknet pair. Not reported here for brevity – the distribution of the amplification factors is centered between 0.9 and 1.1.

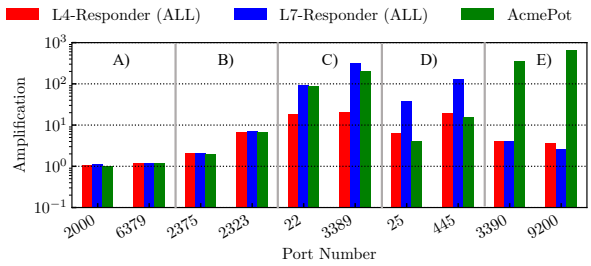
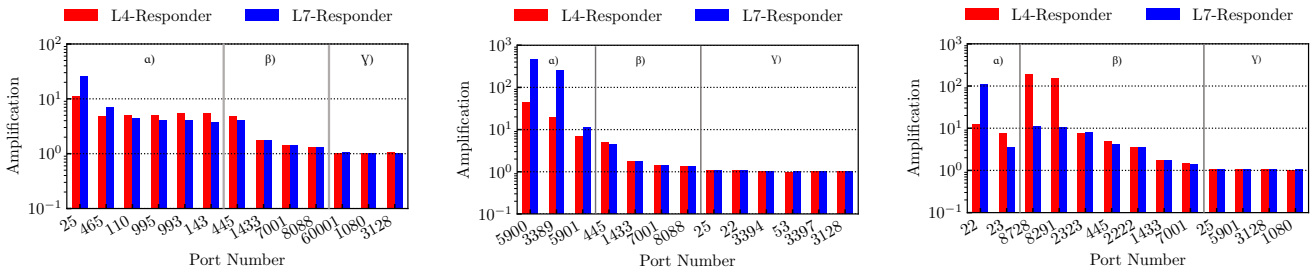


Figure 9: Amplification factor for most targeted ports.

We therefore consider significant any amplification factor outside this range.

Fig. 9 shows the amplification factor for some selected ports. We identify five major categories of behavior, which are labeled with capital letters. We provide two examples per category:

A) Invariant (around 50 000 ports): the traffic reaching these ports does not change significantly from the darknet to the other deployments. Ports like 2 000 and 6 379 receive only *port scan* attempts, whose volume does not change when responders are present;
B) Homogeneous (around 13 000 ports): senders find possible services on some open ports. These open ports trigger senders to contact several other ports on the host – e.g., ports 2 375 and 2 323 in the figure. However, for these ports, senders do not send any L7 payload - e.g., because waiting for servers to initiate the exchange. Here, L7-Responders and AcmePot behave just like L4-Responders;
C) L7 client-initiated (around 500 ports): these are clear cases of open services on default ports with client-initiated protocols, e.g., SSH and RDP on ports 22 and 3 389. Both L7-Responders and AcmePot are effective to engage with the senders. Since frequent attacks are present, we observe very large amplification factors. L4-Responders are less interesting for the senders, with reduced amplification factor;
D) L7 server-initiated (around 10 ports): open services on default ports for which the senders expect the server to initiate the L7 exchange. In this case, the L7-Responders vertical honeypots are more effective, while AcmePot behaves as the L4-Responders, e.g., on SMTP and SMB on ports 25 and 445, respectively;
E) Large-scale attacks on non-standard ports (around 1 500 ports): Senders discover particular services on non-standard ports and perform large attacks. Only AcmePot, which identifies the L7-protocol, spots such behavior. In particular we have witnessed an extensive



(a) Mail (b) Remote Desktop (c) Terminal

Figure 10: Amplification factor for selected deployments. β marks cases of *Side-Scans*.

Table 3: Traffic amplification for L4-Responders and L7-Responders. Cases in which no amplification is observed is marked with a hyphen.

L4-Responders								
	DB	*File*	*Mail*	*Proxy*	*RD*	*Terminal*	*Web*	*Others*
DB	**15.4**	4.3	–	–	–	–	–	–
File	1.6	**42.0**	–	–	–	–	–	–
Mail	1.5	4.1	**6.5**	–	–	–	–	–
Proxy	1.5	4.2	–	**2.7**	–	1.2	–	1.2
RD	1.5	4.2	–	–	**21.2**	1.6	–	–
Terminal	1.5	4.1	–	–	–	**9.3**	–	1.3
Web	1.5	4.2	1.4	1.2	–	1.3	**8.1**	–

L7-Responders								
	DB	*File*	*Mail*	*Proxy*	*RD*	*Terminal*	*Web*	*Others*
DB	**9.3**	3.9	–	–	–	–	–	–
File	1.6	**116.3**	–	–	–	–	–	–
Mail	1.5	3.6	**9.6**	–	–	–	–	–
Proxy	1.5	3.8	–	**2.8**	–	–	–	1.2
RD	1.5	3.8	–	–	**254.9**	–	–	–
Terminal	1.5	3.6	–	–	–	**46.6**	–	1.2
Web	1.5	3.8	1.2	1.2	–	1.2	**5.3**	–

RDP attack on multiple non-standard ports, resulting in around 1 500 ports for which AcmePot amplification grows to almost 1, 000.

Takeaway: Different deployments tend to amplify different behaviors, hinting to a complementarity between them. Overall, the obtained information clearly grows from case A) to case E). The latter is particularly interesting, showing benefits of performing traffic analysis on non-standard ports to push senders to exchange data. AcmePot proves more flexible in this case, concretely leading to the discovery of active attacks on non-standard ports, otherwise unseen with other deployments.

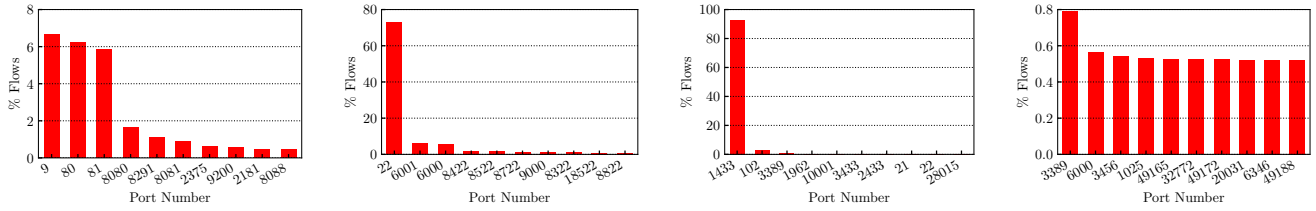
6.2 Targeted services and *Side-Scans*

So far we focused our attention on the responders that support *All* services at the same time. We now check what happens if we partition responders so that they behave as vertical services. For this, we consider L4-Responders and L7-Responders, with categories/ports/applications defined in Table 1. For each category, we compute the amplification factor with respect to the corresponding categories/ports/applications in darknet addresses.

Table 3 summarizes results. For each vertical deployment, we report the amplification factor only when significant. Rows report the category of the deployment while columns report the corresponding categories in the darknet as reference. As expected, activating specific services attracts the attention of senders on them (see main diagonal, in bold). L4-Responders suffice to observe more traffic, but L7-Responders clearly generate much more interaction. Exceptionally, L4-Responders see higher amplification factor than L7-Responders in some cases (e.g., *DB*). This is likely a consequence of our lack of honeypots for some ports of these categories in the L7 backends (see Table 1). In this case, while L7-Responders reply with an uninteresting response (reset the connections), the limitation to TCP handshake offered by L4-Responders further engages the senders.

Very surprisingly, we observe also significant amplification factors on services for which the deployment does *not* answer, i.e., where we drop the SYN packets. Regardless of the deployment, once senders find an IP address that is alive (i.e., answering a popular service), they target other ports in the *DB* and *File* categories. The case of the *Web* category is particularly interesting: When a service is found active on ports typically hosting HTTP services, senders apparently start targeting multiple other services/ports on the same host. We refer to this phenomenon as *Side-Scan* activity.

Fig. 10 reports some of the most relevant *Side-Scan* examples. α) marks the well-known (and open) ports for the category. Here, we get as expected significant amplification factors, with L7-Responders getting significantly more traffic than L4-Responders for some honeypots. β) marks those *Side-Scan* ports that suddenly get targeted - despite being blocked for the particular deployment. These are the ports



(a) HTTP (b) SSH (c) MsSQL-TDS (d) RDP

Figure 11: Flows percentages on top-10 ports for AcmePot and different L7 protocols.

senders target in vertical attacks/scans triggered by a different category. Finally, γ) exemplifies some ports that remain invariant, i.e., they are neither the initial target, nor reached in *Side-Scans*. Most ports fall in this class.

These cases clearly show benefits of activating some services on darknets. The vertical deployments allow us to discover ongoing *Side-Scan* for services, which otherwise would have remained unnoticed in the darknet. When opening ports of the *Mail* category (plot in the left-hand side), we observe significant amplification factors on ports (445, 1 433), which are usually used in *File* and *DB* services. We see curious *Side-Scans* also on ports (7 001,8 088). Similar effects can be seen for the RD category.

Less unexpected, ports (2 222, 2 323) are often used as alternative ports for terminal services – and senders *Side-Scan* these ports when finding standard terminal ports open (see right most plot). Ports (8 728, 8 291) are known to be vulnerable services in old versions of software routers. In particular, we observe frequent "door-knocking" attempts: the sender checks port 22 first; if open but no banner is offered, they check ports (8 728, 8 291). L7-Responders do offer a banner on port 22. Thus, flows on ports (8 728, 8 291) are smaller than in L4-Responders that offers no banner on port 22 [37].

Takeaway: We observe extremely high amplification factors in both L4-Responders and L7-Responders. Deployments targeting a particular service uncover *Side-Scan* activity, which varies according to the main service exposed on the deployment. *Side-Scans* clearly call for more investigations and automated means to check how they work, and who is involved with. We are sure this will trigger future works.

6.3 AcmePot additional visibility

Given the *Side-Scan* phenomena, we dig into AcmePot data to see if its ability to reply to L7 requests on any port increases visibility of this effect. We expect AcmePot to trigger even more *Side-Scans*.

Table 4: Top-5 protocols recognized in AcmePot.

Protocol	Flows	Sender Addr.	Dest. Ports	% of Flows on Standard Ports
RDP	329 652 678	1 415	28 333	0.8
HTTP	444 715	13 705	9 381	6.2
TLS	221 565	2 806	11 999	4.6
SSH	119 698	1 097	187	72.9
MsSQL-TDS	31 596	3 193	448	92.6

Table 4 reports a summary of the most common application protocols detected by AcmePot. We observe a vast majority of RDP flows - with 1 415 senders generating more than 330 M flows in one month. These senders target more than 28 thousand ports, with the standard port 3 389 accounting for only 0.8% of flows. That is, the majority of this attack would go unnoticed on darknets. We have already seen this behaviour in Fig. 5 and Fig. 6 where the IP addresses involved in this attack dominate the traffic AcmePot collects.

AcmePot lets us observe also other popular protocols like HTTP, TLS and SSH, where thousands of senders target thousands of ports. Some of these attacks focus mostly on the default port - like SSH or MsSQL-TDS where 72.9% and 92.6% of the flows goes to the default ports.

To check how senders choose the port to probe for a given protocol, Figure 11 details the most popular ports target of some L7 protocols. Start from the HTTP case. Port 9 results as the most popular port to receive HTTP requests. This is a *Side-Scan* performed by an Internet mapping project of the University of Michigan, which targets port 9 (about 30 000 flows) and 7 (about 50 flows only), sending bogus HTTP requests[34]. This scan activity would likely go unnoticed on traditional honeypots. Besides this curious scan, AcmePot recognizes HTTP requests on non-standard ports that it correctly handles. Given the popularity of solutions based on HTTP protocol, it is not surprising to see senders probe open ports with HTTP requests.

Move to SSH now. Here, most flows target port 22. Yet, the senders do check other ports where system administrators may move the SSH service, e.g., by adding some digits to the

22 ports as in (8 422, 8 522, 18 522). This behavior suggests a targeted *Side-Scan* where senders generate the port to target with some domain-driven algorithm.

The *Side-Scan* using the MsSQL-TDS protocol is even more vertical. Most of the attacks are directed to the default port 1 443, but some few requests go to port 102, likely trying to abuse some Microsoft Exchange service.

At last, the RDP case is worth more details. RDP has become a viable solution for malicious hosts for installing ransomware [49] via attacks that start with password brute-force [8] as well as a common backdoor [4].

Thanks to AcmePot, we observe 1 415 senders performing a password brute-force attack. The attackers however execute the brute-force in almost any port announcing RDP support! Figure 12 shows the targeted ports, ranked per number of received flows. Notice the log-log scale. The step-wise behavior of the figure suggests the presence of a group of 1 000 ports that gets most requests, followed by a second group of ports which are contacted less frequently. This second group may be due to an initial discovery horizontal scan, after which senders come back to perform the brute force password attack. The inner plot shows that there is also a clear pattern for the top-300 ports. Checking which ports each sender targets, we recognize three macro-categories:

- Senders (around 700) that vertically probe only standard RDP port 3 389 and the immediately adjacent ones;
- Senders that focus on a small group of selected ports (e.g., ports 1 289, 23 390, 1 025, 3 418, 50 000, 554, 3 336) - likely chosen via domain knowledge. The four IP addresses involved in this attack belong to the same network and have never been reported at the time of writing. They generate 3.5 million flows;
- Senders that scan thousand of ports (16 IP addresses). These addresses have been reported as heavy scanners [48] and perform very similar activity. This suggests they are part of the same botnet.

Takeaway: AcmePot unveils unexpected *Side-Scan* attacks and scans where senders target non-standard ports. It also triggers activity that L7-Responders in the standard ports do not observe. Senders may become very aggressive, calling for some ingenuity to avoid spending too much resources of the monitoring infrastructure.

7 CONCLUSION

In this paper we systematically analyzed the impact of deploying more and more interactive responders on the darknet address space. Our results show the potential of systematically engaging with senders. We confirm some already known patterns, such as the expected increase in traffic when active services are deployed on the darknet. We also shed lights on new events, such as the *Side-Scans* attracted by opening different services both on standards and non-standard ports. Such patterns remain otherwise unnoticed on darknets or classic honeypots alone.

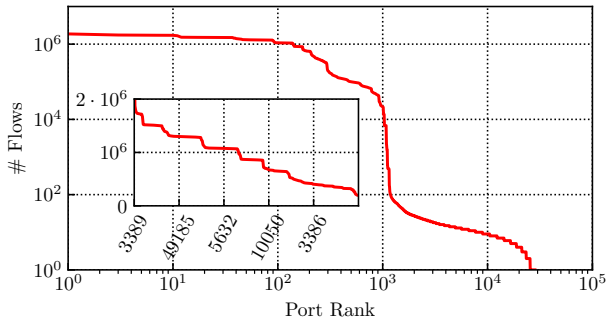


Figure 12: Number of flows per port for RDP traffic. Zoom on first 300 ports in inner axis.

We show that each deployment has its own benefits, unveiling different activities and bringing new perspectives. Combining the several interaction levels augments visibility. However, deployments may impact each other (e.g., polluting neighboring addresses) and may foster traffic increase to the point of saturating the monitoring infrastructure.

Beyond our findings, several challenges are waiting ahead of such hybrid infrastructures. For example, the large amount of collected information calls for automatic methods for analyzing the data, uncovering correlation between deployments, fingerprinting senders and, ultimately, identifying the rise of novel scans and cyberthreats.

REFERENCES

[1] ADBHoney. 2021. Low interaction honeypot designed for Android Debug Bridge over TCP/IP. https://github.com/huuck/ADBHoney

[2] Eric Alata, Vincent Nicomette, Mohamed Kaâniche, Marc Dacier, and Matthieu Herrb. 2006. Lessons learned from the deployment of a high-interaction honeypot. In *2006 Sixth European Dependable Computing Conference*. IEEE, 39–46.

[3] M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J. Halderman, L. Invernizzi, M. Kallitsis, D. Kumar, C. Lever, Z. Ma, J. Mason, D. Menscher, C. Seaman, N. Sullivan, K. Thomas, and Y. Zhou. 2017. Understanding the Mirai Botnet. In *Proceedings of the 26th USENIX Security Symposium (USENIX Security'17)*. 1093–1110. https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/antonakakis

[4] Tim Bai, Haibo Bian, Mohammad A Salahuddin, Abbas Abou Daya, Noura Limam, and Raouf Boutaba. 2021. RDP-Based Lateral Movement Detection using Machine Learning. *Computer Communications* 165 (2021), 9–19.

[5] M. Bailey, E. Cooke, F. Jahanian, A. Myrick, and S. Sinha. 2006. Practical Darknet Measurement. In *Proceedings of the 40th Annual Conference on Information Sciences and Systems (CISS'06)*. 1496–1501. http://ieeexplore.ieee.org/document/4068042/

[6] Paul Barford, Yan Chen, Anup Goyal, Zhichun Li, Vern Paxson, and Vinod Yegneswaran. 2010. Employing honeynets for network situational awareness. In *Cyber situational awareness*. Springer, 71–102.

[7] K. Benson, A. Dainotti, K. Claffy, A. Snoeren, and M. Kallitsis. 2015. Leveraging Internet Background Radiation for Opportunistic Network Analysis. In *Proceedings of the ACM Internet Measurement Conference (IMC'15)*. 423–436. http://dl.acm.org/citation.cfm?doid=2815675.2815702

[8] Matt Boddy, Ben Jones, and Mark Stockley. 2019. RDP Exposed-The Threat That's Already at Your Door. *Sophos, Inc, Sophos White Paper* (2019).

[9] A. Brzeczko, A. S. Uluagac, R. Beyah, and J. Copeland. 2014. Active Deception Model for Securing Cloud Infrastructure. In *Proceedings of the IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. 535–540.

[10] CAIDA/UCSD. 2021. The UCSD Network Telescope. https://www.caida.org/projects/network$_t$elescope/

[11] Cowrie. 2021. SSH/Telnet Honeypot. https://github.com/cowrie/cowrie

[12] J. Czyz, K. Lady, S. Miller, M. Bailey, M. Kallitsis, and M. Karir. 2013. Understanding IPv6 Internet Background Radiation. In *Proceedings of the 13th ACM Internet Measurement Conference (IMC'13)*. 105–118. http://dl.acm.org/citation.cfm?doid=2504730.2504732

[13] A. Dainotti, K. Benson, A. King, B. Huffaker, E. Glatz, X. Dimitropoulos, P. Richter, A. Finamore, and A. Snoeren. 2016. Lost in Space: Improving Inference of IPv4 Address Space Utilization. *IEEE Journal on Selected Areas in Communications* 34, 6 (2016), 1862–1876.

[14] A. Dainotti, A. King, K. Claffy, F. Papale, and A. Pescape. 2015. Analysis of a "/0" Stealth Scan From a Botnet. *IEEE/ACM Trans. Netw.* 23, 2 (2015), 341–354.

[15] A. Dainotti, C. Squarcella, E. Aben, K. Claffy, M. Chiesa, M. Russo, and A. Pescape. 2014. Analysis of Country-Wide Internet Outages Caused by Censorship. *IEEE/ACM Trans. Netw.* 22, 6 (2014), 1964–1977.

[16] Emiliano De Cristofaro, Arik Friedman, Guillaume Jourjon, Mohamed Ali Kaafar, and M. Zubair Shafiq. 2014. Paying for Likes? Understanding Facebook Like Fraud Using Honeypots. In *Proceedings of the 2014 Conference on Internet Measurement Conference (IMC '14)*. New York, NY, USA, 129–136.

[17] L. Deri, M. Martinelli, T. Bujlow, and A. Cardigliano. 2014. nDPI: Open-source High-speed Deep Packet Inspection. In *Proceedings of the International Wireless Communications and Mobile Computing Conference (IWCMC)*. 617–622.

[18] Dionaea. 2021. Generic Low Interaction Honeypot. https://github.com/DinoTools/dionaea

[19] Z. Durumeric, M. Bailey, and J. Halderman. 2014. An Internet-Wide View of Internet-Wide Scanning. In *Proceedings of the 23rd USENIX Conference on Security Symposiu (SEC'14)*. 65–78. http://dl.acm.org/citation.cfm?id=2671225.2671230

[20] C. Fachkha, E. Bou-Harb, and M. Debbabi. 2015. Inferring Distributed Reflection Denial of Service Attacks from Darknet. *Comput. Commun.* 62, C (2015), 59–71.

[21] C. Fachkha and M. Debbabi. 2016. Darknet as a Source of Cyber Intelligence: Survey, Taxonomy, and Characterization. *Commun. Surveys Tuts.* 18, 2 (2016), 1197–1227.

[22] Glutton. 2021. Generic Low Interaction Honeypot. https://github.com/mushorg/glutton

[23] GreyNoise. 2021. https://greynoise.io/

[24] Wonkyu Han, Ziming Zhao, Adam Doupé, and Gail-Joon Ahn. 2016. HoneyMix: Toward SDN-Based Intelligent Honeynet. In *Proceedings of the 2016 ACM International Workshop on Security in Software Defined Networks & Network Function Virtualization (SDN-NFV Security'16)*. New York, NY, USA, 1–6.

[25] Heralding. 2021. Credentials Catching Honeypot. https://github.com/johnnykv/heralding

[26] Honeynet. 2021. The Honeynet Project. https://www.honeynet.org/

[27] Honeytrap. 2021. Advanced Honeypot Framework. https://github.com/honeytrap/honeytrap

[28] M. Jonker, A. King, J. Krupp, C. Rossow, A. Sperotto, and A. Dainotti. 2017. Millions of Targets Under Attack: A Macroscopic Characterization of the DoS Ecosystem. In *Proceedings of the ACM SIGCOMM Internet Measurement Conference (IMC'17)*. 100–113. http://dl.acm.org/citation.cfm?doid=3131365.3131383

[29] Steffen Liebergeld, Matthias Lange, and Ravishankar Borgaonkar. 2014. Cellpot: A Concept for Next Generation Cellular Network Honeypots. *Internet Society* (2014), 1–6.

[30] L. Metongnon and R. Sadre. 2018. Beyond Telnet: Prevalence of IoT Protocols in Telescope and Honeypot Measurements. In *Proceedings of the 2018 Workshop on Traffic Measurements for Cybersecurity (WTMC'18)*. 21–26. http://dl.acm.org/citation.cfm?doid=3229598.3229604

[31] D. Moore, C. Shannon, D. Brown, G. Voelker, and S. Savage. 2006. Inferring Internet Denial-of-Service Activity. *ACM Trans. Comput. Syst.* 24, 2 (2006), 115–139.

[32] S. Morishita, T. Hoizumi, W. Ueno, R. Tanabe, C. Gañán, M. J. G. van Eeten, K. Yoshioka, and T. Matsumoto. 2019. Detect Me If You... Oh Wait. An Internet-Wide View of Self-Revealing Honeypots. In *Proceedings of the IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*. 134–143.

[33] M. Nawrocki, M. Wählisch, T. Schmidt, C. Keil, and J. Schönfelder. 2016. A Survey on Honeypot Software and Data Analysis. *arXiv:1608.06249* (2016).

[34] University of Michigan. 2013. Why am I receiving connection attempts from the University of Michigan? https://cse.engin.umich.edu/about/resources/connection-attempts/

[35] E. Raftopoulos, E. Glatz, X. Dimitropoulos, and A. Dainotti. 2015. How Dangerous Is Internet Scanning? A Measurement Study of the Aftermath of an Internet-Wide Scan. In *Proceedings of the 7th Workshop on Traffic Monitoring and Analysis (TMA'15)*. 158–172. http://link.springer.com/10.1007/978-3-319-17172-2$_1$1

[36] P. Richter and A. Berger. 2019. Scanning the Scanners: Sensing the Internet from a Massively Distributed Network Telescope. In *Proceedings of the Internet Measurement Conference (IMC'19)*. 144–157. http://dl.acm.org/doi/10.1145/3355369.3355595

[37] V. Riyadi. 2018. Securing Mikrotik Router. https://mum.mikrotik.com/presentations/ID18/presentation$_5$554$_1$540255240.pdf

[38] SNARE/TANNER. 2021. Web Application Honeypot Sensor. http://mushmush.org/

[39] Pavol Sokol, Jakub Míšek, and Martin Husák. [n. d.]. Honeypots and Honeynets: Issues of Privacy. 2017, 1 ([n. d.]), 4. https://doi.org/10.1186/s13635-017-0057-4

[40] F. Soro, M. Allegretta, M. Mellia, I. Drago, and L. Bertholdo. 2020. Sensing the Noise: Uncovering Communities in Darknet Traffic. In *Proceedings of the Mediterranean Communication and Computer Networking Conference (MedComNet)*. 1–8. https://ieeexplore.ieee.org/document/9191555/

[41] F. Soro, I. Drago, M. Trevisan, M. Mellia, J. Ceron, and J. J. Santanna. 2019. Are Darknets All The Same? On Darknet Visibility for Security Monitoring. In *Proceedings of the IEEE International Symposium on Local and Metropolitan Area Networks (LANMAN)*. 1–6. https://ieeexplore.ieee.org/document/8847113/

[42] S. Staniford, D. Moore, V. Paxson, and N. Weaver. 2004. The Top Speed of Flash Worms. In *Proceedings of the ACM Workshop on Rapid Malcode (WORM'04)*. http://portal.acm.org/citation.cfm?doid=1029618.1029624

[43] Jay Thom, Yash Shah, and Shamik Sengupta. [n. d.]. Correlation of Cyber Threat Intelligence Data Across Global Honeypots. In *Proceedings of the IEEE 11th Annual Computing and Communication Workshop and Conference* (2021) *(CCWC)*. 0766–0772. https://doi.org/10.1109/CCWC51732.2021.9376038

[44] Christof Ferreira Torres, Mathis Steichen, and Radu State. 2019. The Art of The Scam: Demystifying Honeypots in Ethereum Smart Contracts. In *28th USENIX Security Symposium (USENIX Security 19)*. Santa Clara, CA, 1591–1607. https://www.usenix.org/conference/usenixsecurity19/presentation/ferreira

[45] TPot. 2021. The All In One Honeypot Platform. https://github.com/telekom-security/tpotce

[46] Twisted. 2021. Event-driven Networking Engine Written in Python. https://twistedmatrix.com/trac/

[47] Alexander Vetterl and Richard Clayton. 2018. Bitter Harvest: Systematically Fingerprinting Low- and Medium-interaction Honeypots at Internet Scale. In *Proceedings of the 12th USENIX Workshop on Offensive Technologies (WOOT 18)*. USENIX Association, Baltimore, MD, USA. https://www.usenix.org/conference/woot18/presentation/vetterl

[48] VirusTotal. 2021. https://www.virustotal.com/

[49] ZiHan Wang, ChaoGe Liu, Jing Qiu, ZhiHong Tian, Xiang Cui, and Shen Su. 2018. Automatically Traceback RDP-based Targeted Ransomware Attacks. *Wireless Communications and Mobile Computing* 2018 (2018).

[50] E. Wustrow, M. Karir, M. Bailey, F. Jahanian, and G. Huston. 2010. Internet Background Radiation Revisited. In *Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement (IMC'10)*. 62–74. http://portal.acm.org/citation.cfm?doid=1879141.1879149

[51] Vinod Yegneswaran, Paul Barford, and Dave Plonka. 2004. On the design and use of Internet sinks for network abuse monitoring. In *International Workshop on Recent Advances in Intrusion Detection*. Springer, 146–165.