

Towards Identifying Human Actions, Intent, and Severity of APT Attacks Applying Deception Techniques - An Experiment

Joel Chacon
Eigen Ltd
Surrey, England
joel.chacon@eigen.co

Sean McKeown
School of Computing
Edinburgh Napier University
Edinburgh, Scotland
s.mckeown@napier.ac.uk

Richard Macfarlane
School of Computing
Edinburgh Napier University
Edinburgh, Scotland
r.macfarlane@napier.ac.uk

Abstract—Attacks by Advanced Persistent Threats (APTs) have been shown to be difficult to detect using traditional signature- and anomaly-based intrusion detection approaches. Deception techniques such as decoy objects, often called honey items, may be deployed for intrusion detection and attack analysis, providing an alternative to detect APT behaviours. This work explores the use of honey items to classify intrusion interactions, differentiating automated attacks from those which need some human reasoning and interaction towards APT detection. Multiple decoy items are deployed on honeypots in a virtual honey network, some as breadcrumbs to detect indications of a structured manual attack. Monitoring functionality was created around Elastic Stack with a Kibana dashboard created to display interactions with various honey items. APT type manual intrusions are simulated by an experienced pentesting practitioner carrying out simulated attacks. Interactions with honey items are evaluated in order to determine their suitability for discriminating between automated tools and direct human intervention. The results show that it is possible to differentiate automatic attacks from manual structured attacks; from the nature of the interactions with the honey items. The use of honey items found in the honeypot, such as in later parts of a structured attack, have been shown to be successful in classification of manual attacks, as well as towards providing an indication of severity of the attacks

Index Terms—deception, honeypots, honeynets, honeytokens, APT, early intrusion detection, human actions, severity, intent

I. INTRODUCTION

Ahmad et al. [1] have recently defined APTs as “An entity that engages in a malicious, organized, and highly sophisticated long-term or reiterated network intrusion and exploitation operation to obtain information from a target organization, sabotage its operations, or both”. APTs conduct stealthy operations on specifically selected target organisations and their level of sophistication includes the exploitation and even harvesting zero-day vulnerabilities. Intellectual property or productivity losses after APT attacks can have devastating consequences for both public and private organisations.

Because of these characteristics, APTs remain a major challenge for security professionals. There is evidence in literature that both signature-based and anomaly-based intrusion detection techniques may be ineffective against APTs due to

either a high rate of false positives, false negatives or both [2], [3]. Regular server patching will also not defend against an attack employing zero-day vulnerabilities; and persistent spear-phishing means that it only takes one click for the attacker to set foothold inside the network of an organisation and deploy their custom-coded back-door tools [4].

Therefore, there is still a need for new and effective APT attack detection paradigms and techniques applicable in practice, over a wide range of organisations and industries. In this context, the use of deception techniques has gained recent popularity [5], [6] and also recently explored by the MITRE Corporation [7], and so has been selected as the technique to be explored and investigated in this work.

The original deception device: the honeypot, was developed over 30 years ago with the purpose of attracting attackers to reveal their intentions, strategies and toolset. Most recently the use of other honey items, known more generally as “deception techniques” operate on the basis that any interaction with these resources is malicious in nature and can be detected by traditional monitoring systems.

Deception has developed over the years to include complex constructs such as honeynets and simplified elements known as honeytokens. The latter is a very general concept and is normally defined as a honeypot that is not a computer, but rather a sort of digital entity that has no legitimate purpose [8]. As the role of deception techniques in cyber security increases, so will the need for an ethical framework. Early research suggests that defensive deception, such as proposed in this work, is generally ethical [9].

This paper seeks to experimentally verify that it is possible to detect and classify intrusions according to the level of human reasoning and interaction required to execute or set them up, making use of honeytokens within a self contained system. This is in contrast to prior work which does not experimentally evaluate the use of honeytokens for intrusion detection [8], [10]–[12]. Insights gained from this work are also useful when reasoning whether actions are carried out by a human actor, as honeytokens are designed to require human reasoning, while prior work makes use of statistical heuristics

to identify automated tools [13], [14].

The objective is achieved by setting up an experimental honeypot system in a real private cloud environment to test its ability to detect intrusions and classify them as stated above.

II. BACKGROUND

A big challenge in preventing APT attacks lies in the effectiveness of social engineering, targeting the human element, which is notoriously difficult to mitigate technically. APTs are also inherently stealthy in their operation and constantly adapt to countermeasures developed to uncover them, as has been evidenced by efforts to catalogue and classify their techniques, tactics and procedures (TTPs) [15], [16].

But before an APT launches a social engineering attack, they must collect as much information about the target as possible, and this presents an opportunity to detect their activity, as has been demonstrated in prior work [12], [17].

However, once the initial attack vector has been successful by means of the exploitation of a vulnerability, the attack has successfully compromised a host inside the company intranet. At this stage it can be said that the attacker has gained control of a host “inside the defence perimeter”, and the next stage of infection can be initiated, such as privilege escalation and lateral movements. These steps, even though aided by hacking tools, are human driven as they requires analysis of the findings and decision of where to go next. In this sense, an APT attack may share characteristics with insider threats; because the actions taken after the initial foothold are driven by human actors from compromised user accounts [12].

There are two distinct traditional intrusion detection approaches: signature-based Intrusion Detection Systems (IDS), and anomaly-based IDS: Signature-based IDS are designed to detect specific pre-defined patterns of malicious behaviour and are therefore ineffective against APTs due to the innovative, polymorphic and stealthy nature of these types of attacks. This fact is well known in the research community and therefore the focus has shifted towards anomaly-based detection techniques [12], which focus on detecting intrusions by attempting to recognise patterns of unusual behaviour as reflected in host logs and network traces, generally by using machine learning or hybrid techniques combining deterministic, signature-based and artificial intelligence approaches.

However, some authors [12], [18] claim that anomaly-based detection systems are flawed at the root of their assumption to detect APT actors: while the machine learning paradigm assumes training based on known malicious behaviour patterns, it cannot be expected to be effective against new or unknown attack patterns. It has also been reported that anomaly-based detection systems generate a large amount of log data that needs to be examined by human analysts [10], which includes a large number of false positives. The investigation of ways to reduce the number of false positives continues to be the subject of recent research by Krishna et al. [3]. Likewise, “normal” behaviour data-sets for training are difficult to obtain [18]. Nicho, Oluwasegun and Kamoun [19] also concur, pointing

out that APTs exploit unknown attack patterns or have carefully studied how to evade well-known security mechanisms.

Due to the APTs level of sophistication designed to avoid detection by conventional means, researchers widely agree that these type of attacks are virtually impossible to prevent [12], that an effective defence cannot rely on prevention alone [20], and that the base assumption should be that the malicious actor has already breached the perimeter [21]. Industry analyst Gartner predicted that by 2020, prevention-centred strategies will be completely ineffective against targeted attacks [22], and this is certainly the consensus of the research community.

A shift in emphasis from prevention to response [23] necessarily requires an effective detection mechanism. A key element in countering these types of actors need to be in the early detection of their activity so that countermeasures and escalation prevention actions can be initiated as early as possible in order to limit damages.

FireEye [24] reported that, on average, adversaries inside a North American network remain undetected for 56 days, a figure which is even greater in other regions. CrowdStrike [25] found that, on average in 2019, an adversary takes 9 hours to move laterally and compromise other hosts within an organisation; all of which emphasises the need for fast and early detection of breaches.

Two key advantage of deception techniques is that their false-positive rate can be significantly lower compared with other intrusion detection techniques, as any interaction is by definition not normal; and at the same time have a higher probability of reducing the false-negative rate, as any new, unknown pattern of attack is likely to be successful against deployed honeypots [8] and therefore effectively detected [26].

While honeypots are essentially computer hosts with a CPU at its core; honeytokens are not and take many forms. Files with decoy content can be honeytokens if they are planted throughout the network’s resources and given interesting names, such as “master-passwords”, and monitored for access, as suggested by Kim and Spafford [27]. Their work is the earliest suggestion of the possible usefulness of a honeytokens even before the term was coined.

False records can be honeytokens; such as fake user or user-name entries in databases: If the record is used to attempt to access additional information, it would signal an unauthorised breach [8]. As Spitzner [28] notes, the concept of a honeytokens can be extended beyond database entries: file, Web or email servers can all have embedded honeytokens such as unique tags detectable by traditional signature-based IDS system.

Applied to Web Servers, Fraunholz and Schotten [26] identified six deception techniques: Fake Web server banners; false entries in the robot.txt file; fake error code responses; honey or decoy files, with information leading to intrusion detection or to distract or delay any attacker, respectively; introducing adaptive delays to IP addresses making repetitive requests; false comments within the HTML code: “honey-comments” containing any misleading information such as fake credentials and fake links. These researchers applied several of these techniques to existing production Web servers, to mislead

attackers and increase the security of Web servers. Several of these techniques had already been proposed by Virvilis [12].

The term "Breadcrumbs" has recently appeared in commercial literature [5], [29], referring to a type of honeypot planted in real production systems in order to lure attackers towards honeypots and other decoys.

Various authors claim that the use of deception techniques and honeypots in particular can be effective against cyber-attacks, however, none of these authors claims to have demonstrated this in practice: Spitzner [8] "feels" that honeypots can be effective against actors seeking information, such as intelligence seeking or industrial espionage. Almeshekah & Spafford [10] conclude that deception techniques have been shown to be effective in many contexts because they take advantage of human biases making honey-x devices seem plausible, but do not demonstrate this experimentally. Zhao & Mannan [11] claim that their fake authentication and session handler *Uvauth* can "effectively deceive an attacker assuming fake sessions can be efficiently generated", but do not report results of any experimental trials and in their conclusions admit that the system has not been fully evaluated. Finally, Virvilis [12], who proposes a number of simple and attractive honeypot techniques directed at detecting APT and insider-attacks does not claim to have tested them for effectiveness.

On the other hand, other researchers working with low or medium interaction honeypots exposing an SSH service to the internet have explored the topic of segregating human vs bot (i.e. automated) interactions. It is well known that hosts exposed to the internet will be attacked by opportunistic actors dedicated at harvesting compute boxes in order to launch other attacks; and so these researchers logged and analysed breach attempts in their thousands. Using medium interaction honeypots that enabled fake login sessions, Kemppainen and Kovanen [13] identified bots through measurable host logged features: command timestamp consistency and delay between commands (almost always 4 seconds), command execution despite errors in the previous command, and detecting the same command executed over separate sessions. Unsurprisingly, they found that only 37 interactions were executed by humans (2.3% of the total).

Udnahi et al. [14] used a low interaction honeypot (only to capture login attempts), simulating a server farm of 65,000 hosts and analysed over 5 million connection requests. They characterised human actions based heuristics determined from an observation of the data: the number of requests per minute (≤ 10 requests/min), the number of targets attempted (≤ 2 per day), rate of characters in each request (≤ 3 per sec) and the speed at which the password guess is typed (≤ 3 characters/sec). However, this methodology did not consider the initiation of a brute-force attack with particular username and password dictionaries which may be derived from earlier Open-Source Intelligence (OSINT) or reconnaissance activities on the target and would be classed as actions by a bot.

III. METHODOLOGY

Given the lack of experimental data in the identification of human interactions in the context of an insider threat or an APT actor with a foothold inside the organisation, the research questions to be investigated are:

RQ1: Can the influence of human reasoning and interactions be deduced with the use of honeypots in honeypots located within the security perimeter?

RQ2: Can such an arrangement of honeypots be used to determine the level of intent or severity of the attack?

To answer these questions, the experimental method focused on the deployment and evaluation of various honeypot techniques in a high interaction Linux-based honeypot deployed in a private cloud environment. The honeypot will be connected and communicated to via a purposely deployed virtual network segment which also has no legitimate purpose, and is an integral part of the deception (ie a honey sub-net).

The principle of all honeypots for intrusion detection is that any interaction or actions resulting from the observation of honeypots represent an intrusion attempt; therefore, the experiment also required the setup of a network traffic, honeypot and honeypot monitoring infrastructure. The environment and experimental design followed the steps below:

- 1) Selection of a set of honeypot techniques, suitable to address the research questions.
- 2) Selection of a honeypot where the selected honeypot techniques can be deployed, so that human interactions, intent and severity of the intrusion attempt can be determined.
- 3) Deployment of the experimental honeypot and honeypot arrangement in a company virtual private infrastructure within a virtual sub-net.
- 4) Design and deployment of an intrusion detection harness to monitor honeypot logs and network traffic indicative of interaction with the deployed honeypots and honeypots.
- 5) Setup of a VPN with direct access to the virtual sub-net where the honeypot is located.
- 6) The design of the experiment to simulate an insider or APT attack. This was done by means of a penetration testing (pentest) engagement via the VPN.

The experimental design includes inviting pentesters and ask they behave as APT attackers would in the second stage of the attack; after the initial foothold within the company firewall. The hypothesis is that they would find and try to use the honeypots as means of breaching the security of the honeynet systems thereby triggering the detection of intrusions, and human interaction in particular. The authors recognise that the experimental design may differ in many aspects from the behaviour of a real APT actor, e.g. the pentester is directed at a particular sub-net and given a specific pentest brief, while real attackers can decide where what and where to probe; however, it is believed that the human behaviour of an attacker upon observing the honey items may be similar to that of a pentester, which is at the core of this research.

Several honeypot techniques were selected from the literature to help evaluate the interactions and for comparison to previous work:

- 1) Use of “disallowed” entries with the robot.txt file [12], [26]
- 2) An invisible link (e.g. white font over white background) in a webpage [12], and
- 3) Fake credentials added as comments within the HTML source. In the case of the fake credentials [12], [26]. This honeypot had an additional original added feature: the username planted was an incomplete email; i.e lacking the full domain name.

The selection of these honeypot techniques is justified based on their potential to detect human interactions and be self-contained within the honeypot. Indeed, honeypot 1) and 3) may also be considered breadcrumbs, because their discovery may invite a human actor to investigate further into the deception: explore the disallowed folder or try the fake credentials as a login. The selection of technique 3) also aligns with MITRE’s technique T1081 “Credentials in Files” [15].

The monitoring function outputs and honeypot interactions are categorised into three types of intrusion indicator, depending on the severity of the intrusion detected:

- **Low priority**, if there is any level of network detected interaction with the honeypot or the use of automatic reconnaissance tools with minimal human reasoning. These are indications of attacker reconnaissance activity or the start of an attack on the honeypot.
- **Medium priority**, if the results of the reconnaissance were used to take the intrusion further and attempt a breach, with some possible human actions and if they indicate sustained attack activity using automated tools.
- **High priority**, if honeypots were discovered by a human actor and further reasoning was applied to expand the efforts leading to a breach, for example if the information found in one honeypot (a breadcrumb) was used in a subsequent step, indicating a high level of intent.

Sub-classifications of these indicators are also possible to indicate degrees of severity of each type of attack detected.

If intrusions are detected, which would have to have had human associated level of prior reasoning, which were not possible or unlikely to be executed by a bot, would point towards fulfilling RQ1. If different intrusion level indicators (low, medium and high) were detected in sequence from low to high, that would indicate some success in terms of RQ2, and a high level of intent in breaching the system.

Since all these honeypots are part of a webserver, a fake client user login application was implemented using PHP code on the honeypot, which never leads to a session and simulates variable response times to the user.

Figure 1 represents the experimental set-up and monitoring infrastructure. The Elastic Stack, by Elastic, composed of Elasticsearch, Logstash and Kibana was used for the monitoring of the honeypot logs and network traffic. Filebeat, an open source log file shipper was used to monitor various system

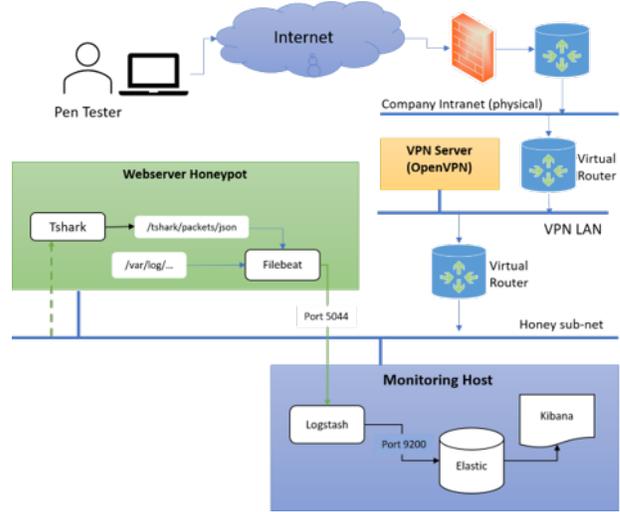


Fig. 1. Experiment Set Up and Network Diagram

logs and TShark was used to monitor network traffic in and out of the honeypot. TShark can directly output a JSON file format directly compatible with Logstash [30].

Filebeat was installed in the honeypot webserver to monitor webserver (Nginx) and PHP logs, which were then ingested by Logstash on the monitoring server. The Nginx logs record access to different served pages as standard. Within the webpage PHP code, any string message can be programmatically directed to the PHP log. This feature was used to create log messages when the index.php page was accessed and also the user email and password entered in the login page. Filebeat was also used to monitor the packet capture JSON log file created by TShark.

The following filter (see below) was added to the TShark command line to output ICMP and TCP SYN packets to or from the honeypot, as a means to detect incoming network activity (attack) or outgoing traffic (honeypot compromise):

```
tshark -i eth0 -f "icmp || (tcp[tcpflags] & (tcp-syn) != 0 and tcp[tcpflags] & (tcp-ack) = 0)" -T ek > /home/tshark/packets.json
```

Logstash was configured to create indices from these data sources and push the result into the Elasticsearch database. Kibana was used to set up specific monitors to detect events of interest in a single dashboard.

A custom Kibana dashboard (Figure 2) was configured to display (from left to right, top to bottom): a) ICMP packets, b) Source IP and timestamp of Index.php access, c) No. of clicks on hidden URL, d) Timestamp, Source and Destination IP of any TCP SYN packets, e) Timestamp and credentials used in any login attempt and f) No. of accesses to disallowed folder.

Because this setup detects traffic on the honey sub-net and interactions with the honeypot, any event detected will have a malicious or unauthorised implication; but with varying levels of severity. Section V (specifically Table I) analyses the types of intrusions that can be detected in the system as designed, their suitability to be attacked by an automated process and their implied intent and severity.

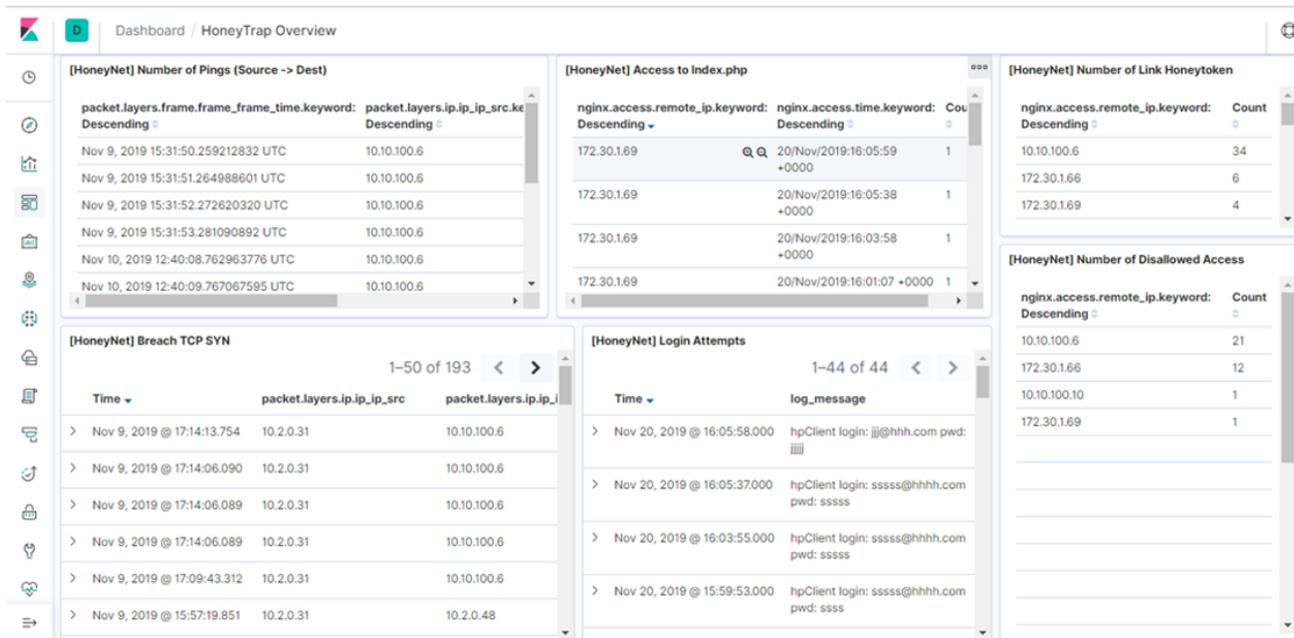


Fig. 2. Kibana Dashboard

IV. PENETRATION TEST PROCESS SUMMARY

An experiment was carried out in January 2020, with a duration of approximately 1.5 hours. The test was conducted by an experienced industry professional pentester.

The sequence of events of the pentest was as follows, as described by the tester:

- 1) Initial brief reconnaissance of available hosts using Nmap and including ping sweeps. Pings to the webserver (honeypot) host were captured by the monitoring harness.
- 2) Version scan and OS detection using Nmap.
- 3) Analysis of HTTP requests and responses. Modifying with special characters looking for SQLi and other code injections, using Burp suite.
- 4) Vulnerability scan of HTTP servers, using Nikto.
- 5) Directory fuzzing of HTTP servers, using dirb.
- 6) Attempt to brute force login of webserver, using Hydra.

The pentest attack followed several stages as evidenced in the Kibana dashboard and the sequence of events in the integrated logs. The elapsed time after the test start time is indicated between brackets.

Stage 1 – Reconnaissance

[+ 8 mins] During this stage, after a single ping; a flood of TCP SYN packets was detected; as shown in Figure 2.

Stage 2 – Burpsuite sweep of Web application

[+ 13 mins] The count of SYN packets reaches 173 while Burpsuite scans through the Web application pages; the main “Index.php” which presents the fake login is accessed a few times and non-authorized /admin folder is attempted 10 times. Recall the /admin folder is included in robot.txt file as disallowed; however, at this stage, it is deduced that it

was attempted to be accessed by Burpsuite irrespective of its inclusion in robot.txt.

Stage 3 – Exploring the fake login page (Index.php)

[+ 14 mins] The count of SYN packets reaches 216 while Burpsuite continues to scan through the Web application pages; the main “Index.php” which presents the fake login is accessed 35 times and a blank login email/password combination is detected. This is clearly the continuation of the automated scan stage; given the number of SYN packets, the number of detected separate accesses to Index.php and the number of attempts to reach /admin (47 times).

Stage 4 – Login credentials, initial test

[+ 14 mins] While the automatic scanning continues; at this stage is the first sign of human intervention with the attempted login with credentials that could be guessed from the context of the test.

Stage 5 – Further automated scans

[+ 17 mins] While the login attempts temporarily stop at 3; the number of SYN packets rises again, the number of accesses to Index.php rises to more than 20 every second and attempts to access /admin shoots to more than 1300. This indicates a return to automatic scanning tools, possibly dirb¹. This stage continues until with 783 SYN packets, 11 blank login attempts and 12,004 attempts to access the /admin folder, as shown below. As determined during the analysis of these results; this sharp increase was the result of the use of the tool dirb, fuzzing sub-folders inside /admin.

Stage 6 – Use of Fake Credential Honeytoken

[+ 36 mins] At this stage the attacker has examined the source

¹A Web content scanner. See <https://tools.kali.org/web-applications/dirb> for more information

code of the Index.php page and identified the fake credentials added as a comment:

```
<!-- test login user: eigentest1@eigen psw:
e1Ars3nal -->
```

And tried to use these credentials to gain access in three opportunities: The first time he makes a mistake (password “e1Ars3nal” instead of “e1Ars3nal”), then he corrects this mistake; and finally completing the incorrect email reference by adding .co to the user email, as per the company’s domain (eigen.co). This stage confirms human intervention and reasoning impossible to be duplicated by a robot, at least with the current state of technology. It also demonstrates clear intent to breach the security; as the attacker followed a series of clues to try to breach the security of the Web page.

Stage 7 – Access to hidden link Honeytoken

[+43 mins] At this stage the hidden link was discovered and accessed. The pentester reported that this link was discovered with dirb; but dirb uses a fuzzing technique to try to guess hidden folders. A wide search across all of dirb wordlist files (command line: “cat * | grep testlogin”) revealed that “testlogin” is not one of the directories that dirb could guess, unless a very customised wordlist is used.

On the other hand, when Burpsuite was directed at the root of the Web application in order to verify this result; it immediately detected the link /testlogin/index.php. Since Burpsuite was already used in Stage 2 of the pentest the implication is that the repeated run to discover and investigate this hidden link demonstrates a higher level of commitment and intention to attack.

Stage 8 – Login Brute Force

[+ 66 mins] In this stage a login brute force attack was attempted, pre-configured with variations on the original login honeytoken; and common simple passwords; e.g. “123456”; and usernames such as admin@eigen.co, administrator@eigen.co and ROOT@eigen.co. An earlier login attempt with email ‘@gmail indicates a type of SQL injection.

[+ 94 mins] the credentials “eigentest1@eigen” with password “Liverpool” (another football team) was attempted; again, a variation on the planted honeytoken. By +97 mins a total of 1412 email/password combinations had been tested; many focusing on the username of “eigentest1@eigen” planted in the honeytoken; with a large number of different passwords; which indicates a premeditated configuration of the brute force attack using the fake credential username discovered prior.

[+ 98 mins] the Test concluded.

V. ANALYSIS

The results of the experiment reveal some interesting findings from the outputs of the monitoring functions and the interactions with the honeytokens. Some of the monitoring functions served to indicate an attack is being launched without revealing the threat level or severity of the intrusion, while others revealed clear intent and human interaction.

In what follows all these cases will be examined. Figure 3, shows a state diagram of the attack pattern indicating the indicator priority of each type of event detected.

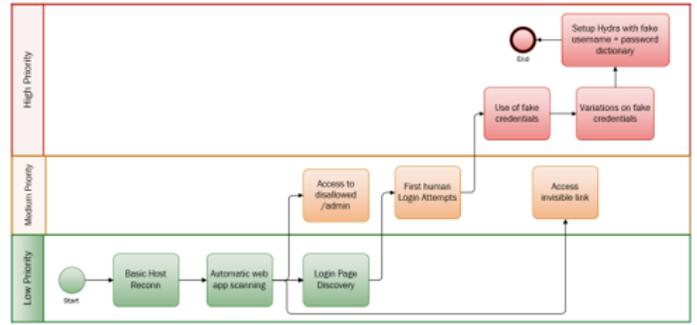


Fig. 3. Attack State Diagram

Low Priority Indicators: Any detected network or Web application interaction with the honeypot is indicative of reconnaissance activity by a potential attacker, but in some cases, it may be an opportunistic or an interaction by mistake by an insider with no malicious intent.

The detection of ICMP and TCP SYN packets directed at the honeypot both served to indicate this level of intrusion; they were one of the earliest signs of attack activity. There is nothing malicious about TCP SYN packets in company networks; but in this case, they indicate possible recon or TCP communication initiation to/from the honeypot; which is deemed unauthorised activity. Outgoing TCP SYN packets, would indicate a honeypot breach; a higher level of intrusion.

Signs of automatic Web app scanning and the discovery of the fake login Index.php page is also an early indication as can be seen from the profile of events.

Medium Priority Indicators: Initiation of a Web content scan implies that the host was identified as serving a Web page.

Access to the /admin folder which was included in the robot.txt file is a medium priority indicator, because this folder was detected directly by the initial Web scan process (Burpsuite) irrespective of its inclusion as a disallowed access within the robot.txt file.

Also, the sheer number of hits (a total of 56,114) indicated the use of an automated Web scanning tool, as any normal user would not likely be accessing this folder so many times; however, the attack can still be opportunistic because it used a standard tool and required little human configuration or effort.

Tests on the Web application carried out by the author using dirb revealed that the /admin folder was one of the first folders found by the tool, which then focused on the discovery of sub-folders within the /admin folder: This helps to explain why so many hits were detected.

A better implementation of this honeytoken might be the inclusion of less commonly named disallowed folders in robot.txt, one that is not part of dirb’s dictionaries or commonly used by Web developers, so that it would not have been scanned directly by common Web scanning tools using fuzzing techniques; but rather indicate an analysis of the contents of the robot.txt file.

Another medium priority indicator can be the access to the hidden link “/testlogin/index.php”; since it was not detected

Event detected	Implication	Severity Level (Design)	Auto/Human	Actual Result and Severity
ICMP pings or incoming SYN traffic on Honeypot 1	Initial stages of reconnaissance, opportunistic attacker or involuntary action by insider.	Very Low	Auto	Verified Correct. Very Low. (packet capture)
Access to Index.php of the Webserver	Initial stages of webserver reconnaissance.	Low	Auto	Verified Correct. Low. (Nginx log)
Click on hidden url	Web application crawling	Medium-Low	Auto or Human	Verified Correct. The link was most probably discovered by Burpsuite; towards the end of the pentest. (hidden /testlogin/index.php)
Login to Honeypots 1 attempt via SSH	Attempt to breach security via online brute force	Medium-Low	Auto or Human	Not observed.
Login attempt in Web portal with any username/password	Attempt to breach security via online brute force	Medium-High	Auto or Human	Verified Correct. Medium-High. (Log of usr/psw from PHP code)
Access attempt to disallowed entry in robot.txt	Deliberate malicious attempt to expand reconnaissance scope	Medium-High	Auto	Disallowed entry was not a factor in browsing the folder Low. (Access to /admin)
Login attempt in Web portal with the honeytokens credentials hidden as HTML comments	Attempt to breach security via online brute force after examining the comments in the HTML. Indicative of human action.	High	Human	Verified Correct. High. (Log of usr/psw from PHP code)
Login attempts in Web portal with variations of the theme of the honeytokens credentials hidden as HTML comments	Attempt to breach security via online brute force after examining the comments in the HTML and reasoning logical variations. Indicative of human reasoning and action.	High-High	Human	Verified Correct. High-High. (Log of usr/psw from PHP code)
TCP SYN initiated from any honeypot (breach of honeypot shell)	Successful security breach. If honeypots have been patched to latest version, implies possible exploitation of zero-day vulnerability	High-High (APT breach)	Human	Not Observed.

TABLE I
HONEYTOKEN INTERACTIONS AND SEVERITY OF THE DETECTED EVENTS

until stage 7 of the pentest; via the use of an automatic tool. The main advantage of this honeytokens is the fact that we can be almost certain that it was detected by automatic tool, because the link is invisible to the human eye.

High Priority Indicators: The honeytokens that resulted in the most effective high priority indicator was the fake credentials within the HTML page comments. The use of these credentials indicated deliberate analysis of the contents of the HTML source file and the introduction of variations in the credentials that were tested suggests the presence of a committed human actor, intent on breaching the security of the Web server.

The three login attempts with variations on the fake credentials were detected, including correcting a typo and adding the correct domain name to the fake user email; which could only be human attempts based on prior acquired knowledge.

This manual, low intensity, breach attempt was followed by an automated brute force attack towards the end of the pentest. This brute force attack is also a high priority indicator in that it takes human effort to configure and launch the attacking tool (in this case Hydra) pre-configured with the username previously obtained from the HTML comment honeytokens.

The brute force attack is easily detectable by the large number of login attempts (1406) vs the prior manual attempts or by other Web scanning tools such as Burpsuite (45).

The results of the experiment indicate that it is indeed possible to differentiate between automated attacks and those with human involvement, by means of the monitoring and honeytokens techniques deployed.

Table I summarises the results of the observations and their implication in terms of automated vs human interactions, level of intent and severity. The experimental results verify that it was possible to segregate initial reconnaissance activities and automatic Web application scanning operations from human reasoned and initiated attacks within the security perimeter (RQ1). It was also possible to infer the level of intent of the attacker using the technique of planting credentials that hinted at possible variations, such as completing a correct domain name (in this case adding a “.co” at the end), which in itself implies previous knowledge of the owner of the site being attacked (RQ2). This was also verified by the evidence of an automated brute force attack which was configured to use the fake username credentials and a dictionary of passwords. The fact that the attacker was led to try a further attack based on the honeytokens shows that the concept of breadcrumbs could be effective in detecting a structured attack. Indeed, this simple honeytokens technique (planted fake credentials) can be used, not only within HTML comments, but also within files, configured within domains, and as email accounts, as shown in previous work in the area of honey items.

These results demonstrate that it is possible to assign a priority or severity level to intrusion detection events by use of simple honeytokens and breadcrumbs. The concepts explored here could be improved and extended by: a) Using non-common folder names that cannot be fuzzed; b) Extending the use of breadcrumb honeytokens in production servers, giving clues that point to the honeypots or invite a login

to servers; c) Allowing some of the fake credentials planted to be valid logins within honeypots, which can lead to the discovery of other honeytokens; thereby leading the attacker into a false trail of discovery, all the while being observed; and d) designing honeytokens that can be discovered by the documented MITRE ATT&CK TTPs [15].

VI. CONCLUSIONS

The results of the experiment reveal an increment in the level of human reasoning and actions: While at the start, the human actor is launching automatic reconnaissance tools (Low priority), very soon this leads to the more detailed investigation of findings and even attempting a few manual attempts to login with any generic context-related credentials (Medium priority). The detection of these Low and Medium priority indicators starts to give a positive answer to RQ1. Finally, the discovery of the fake credential breadcrumb leads to manually testing them directly, creating plausible variations and setting up a brute force attack based on them, all of which demonstrate a high degree of human reasoning and a high level of intent (High priority), demonstrating both RQ1 and RQ2. Prior research focused on detecting automated attacks on login attempts due to their regularity or predictability and considering all other events as human driven, but the present work focuses directly on capturing those events which are very likely executed by a human showing that honeytokens are a very simple but effective technique in achieving this.

If this type of deception is embedded within an organisation's intranet, it has the potential to detect the start of an APT or insider threat attack with very low false positives (as the attack intent and reality would be verified by the detection of ever higher priority indicators that demonstrate human actions) and low false negatives, as the honeytokens would be discoverable by any new unknown type of attack. Future work can focus on extending the use of the most effective honeytokens techniques across production server and client hosts and allowing some of the fake credentials to initiate controlled sessions within honeypots to continue to observe the behaviour of the attacker. Future ambitious experiments that could reach more general conclusions could focus on setting up a completely fake company network on a public cloud, with fake user avatars that allow themselves to be hacked via phishing emails, in order to study the path of real APT actors in the wild from initial foothold onward.

REFERENCES

- [1] A. Ahmad, J. Webb, K. C. Desouza, and J. Boorman, "Strategically-motivated advanced persistent threat: Definition, process, tactics and a disinformation model of counterattack," *Computers and Security*, vol. 86, pp. 402–418, 2019.
- [2] D. Fraunholz, S. D. Anton, C. Lipps, D. Reti, D. Krohmer, F. Pohl, M. Tammen, and H. D. Schotten, "Demystifying Deception Technology: A Survey," 2018.
- [3] C. R. Krishna *et al.*, "A hybrid approach to mitigate false positive alarms in intrusion detection system," in *International Conference on Computer Networks and Communication Technologies*. Springer, 2019, pp. 837–848.
- [4] Trend Micro, "Spear-phishing email: most favored APT attack bait, Trend Micro incorporated research paper," *Research paper*, pp. 1–8, 2012.
- [5] B. Y. P. Musich, "A Definitive Market Guide to Deception Technology A Definitive Market Guide to Deception Technology," no. August, 2019.
- [6] K. Dunn, "Illusive Networks, Deception Management System V3.1.105," *SCMagazine.com*, no. August, 2019.
- [7] D. L. Schuh, "'The Cyberspace Advantage : Inviting Them In' ! How Cyber Deception Enables Better Resilience," no. 19, 2019.
- [8] L. Spitzner, "Honeypots: Catching the insider threat," *Proceedings - Annual Computer Security Applications Conference, ACSAC*, vol. 2003-Janua, pp. 170–179, 2003.
- [9] N. C. Rowe, "The Ethics of Deception in Cyberspace," *Calhoun: The NPS Institutional Archive*, p. 13, 2008.
- [10] M. H. Almeshekah and E. H. Spafford, "Planning and integrating deception into computer security defenses," in *Proceedings of the 2014 New Security Paradigms Workshop*, 2014, pp. 127–138.
- [11] L. Zhao and M. Mannan, "Explicit authentication response considered harmful," in *Proceedings of the 2013 New Security Paradigms Workshop*, 2013, pp. 77–86.
- [12] N. Virvilis, "Changing the game : The art of deceiving sophisticated attackers," *2014 6th International Conference On Cyber Conflict (CyCon 2014)*, pp. 87–97, 2014.
- [13] S. Kemppainen and T. Kovanen, "Honeypot utilization for network intrusion detection," *Intelligent Systems, Control and Automation: Science and Engineering*, vol. 93, pp. 249–270, 2018.
- [14] S. Udhani, A. Withers, and M. Bashir, "Human vs bots: Detecting human attacks in a honeypot environment," *7th International Symposium on Digital Forensics and Security, ISDFS 2019*, pp. 1–6, 2019.
- [15] MITRE, "MITRE ATT&CK," 2020. [Online]. Available: <https://attack.mitre.org/>
- [16] A. Lemay, J. Calvet, F. Menet, and J. M. Fernandez, "Survey of publicly available reports on advanced persistent threat actors," *Computers and Security*, vol. 72, pp. 26–59, 2018.
- [17] A. Paradise, A. Shabtai, R. Puzis, A. Elyashar, Y. Elovici, M. Roshandel, and C. Peylo, "Creation and Management of Social Network Honeypots for Detecting Targeted Cyber Attacks," *IEEE Transactions on Computational Social Systems*, vol. 4, no. 3, pp. 65–79, 2017.
- [18] R. Sommer and V. Paxson, "Outside the Closed World : On Using Machine Learning For Network Intrusion Detection," *2010 IEEE Symposium on Security and Privacy*, pp. 305–316, 2010.
- [19] M. Nicho, A. Oluwasegun, and F. Kamoun, "Identifying vulnerabilities in apt attacks: A simulated approach," in *2018 9th IFIP International Conference on New Technologies, Mobility and Security (NTMS)*. IEEE, 2018, pp. 1–4.
- [20] S. Piper, *Advanced Threat Protection for Dummies, Blue Coat Systems Special Edition*. Hoboken, NJ: John Wiley & Sons, Inc., 2013, vol. 91.
- [21] V. E. Urias, W. M. Stout, and C. Loverro, "Computer network deception as a Moving Target Defense," *Proceedings - International Carnahan Conference on Security Technology*, vol. 2015-Janua, pp. 1–6, 2016.
- [22] R. Brewer, "Advanced persistent threats: Minimising the damage," *Network Security*, vol. 2014, no. 4, pp. 5–9, 2014.
- [23] R. Baskerville, P. Spagnoletti, and J. Kim, "Incident-centered information security: Managing a strategic balance between prevention and response," *Information and Management*, vol. 51, no. 1, pp. 138–151, 2014.
- [24] Fireeye, "M-Trends 2020," p. 76, 2019. [Online]. Available: <https://content.fireeye.com/m-trends/rpt-m-trends-2020>
- [25] CrowdStrike, "CrowdStrike Global Threat Report," pp. 1–32, 2020. [Online]. Available: [papers3://publication/uuid/DCFC62BA-BD06-4255-9A01-BAD5DE94D643](https://www.crowdstrike.com/publication/uuid/DCFC62BA-BD06-4255-9A01-BAD5DE94D643)
- [26] D. Fraunholz and H. D. Schotten, "Defending Web Servers with Feints, Distraction and Obfuscation," *2018 International Conference on Computing, Networking and Communications, ICNC 2018*, no. November, pp. 21–25, 2018.
- [27] G. Kim and E. Spafford, "Experiences with tripwire: Using integrity checkers for intrusion detection," 1994. [Online]. Available: <http://docs.lib.purdue.edu/cgi/viewcontent.cgi?article=2114&context=cstech>
- [28] L. Spitzner, "The honeynet project: Trapping the hackers," *IEEE Security & Privacy*, vol. 1, no. 2, pp. 15–23, 2003.
- [29] A. Bushby, "How deception can change cyber security defences," *Computer Fraud and Security*, vol. 2019, no. 1, pp. 12–14, 2019.
- [30] C. Wurm, "Analyzing Network Packets with Wireshark, Elasticsearch, and Kibana," p. 1, 2017. [Online]. Available: <https://www.elastic.co/blog/analyzing-network-packets-with-wireshark-elasticsearch-and-kibana>