Fidelis™ Cybersecurity

Applying Deception Mechanisms for Detecting Sophisticated Cyber Attacks

www.fidelissecurity.com

Contents

Executive Summary

Overview

A new and deadly generation of remotely controlled targeted corporate network attacks is challenging core network security assumptions, making prevention-centric strategies obsolete.[1]

While network security teams are starting to shift their focus from perimeter defense to post-breach detection, traditional detection tools fall short of the mark, either generating far too many false-positives or altogether failing to detect attacks in real time. These shortcomings are discussed in a study published by Mandiant Consulting,[2] which found that 53 percent of all data breaches are discovered by an internal notification, not through external detection efforts. The study further mentions that the average time between infection and detection by an external source was 99 days.

Deception — the use of decoys, traps, lures and other mechanisms that will be discussed in this paper — is quickly gaining the attention of organizations seeking an efficient post-breach detection technology.

The research department at Fidelis Cybersecurity conducted an experiment to investigate the performance of deception technologies in a simulated corporate environment in which more than 50 professional hackers and security experts used their knowledge and skills to try to extract a pre-defined piece of data and stay undetected. The experiment was conducted as a Capture the Flag (CTF) challenge; and in addition the environment was tested against a variety of malware programs.

The experiment sought to answer a number of questions, including:

- What kind of attacker will be attracted to what different type of resources (traps)?
- What deception mechanisms should the defending organization employ?
- Where should they be placed?
- What kind of traps should be used?

Every attack pattern was carefully monitored and upon completion the data logged was analyzed and aggregated. Trends, attack patterns and statistics were derived from the data logged.

Key Findings

1. 100 percent of attackers were detected using one or more of the mechanisms planted
2. 66 percent of the attackers were lured to and detected by the decoys; the rest were detected by other deception mechanisms such as data traps and beacon traps
3. Deception increased the attacker's knowledge gap and led to detection in very early stages of the attack, including highly sophisticated attackers
4. The more tailored traps and decoys were to the specific environment, the more effective they were
5. Diversity was key to effective detection-by-deception. Different attackers were drawn to different traps. The research found that a diverse range of traps and decoy types increased the defender's detection capability
6. Non-sophisticated network intel-gathering attempts were quickly and accurately detected by decoys
7. The more time attackers spent within the network, the more silent and harder to detect they became. The deployment of mini-traps helped to overcome this issue.

[1] Gartner, Prevention Is Futile in 2020, January 2016

[2] Mandiant Consulting, M-Trends 2016, February 2016

Root Cause

The variance in exploitation methods and attack vectors — email usage, downloading, and smartphones connected to Wi-Fi and social networks to name just a few — has made security professionals realize that attackers cannot be prevented from entering the corporate network. Today it is accepted that determined attackers will penetrate the network sooner or later, it's just a matter of time. This paradigm change dictates a shift from perimeter security to post-breach detection and resolution.

Attackers and Defenders — Advantages and Disadvantages of Both

Cyberattacks are no different from military and homeland security situations in that the attacker has an immediate advantage of being the initiator. Unlike conventional combat and even espionage, cyber attackers have the added advantages of anonymity and knowing their goal, which might not be obvious to the defender. In comparison to something tangible and shiny such as valuable works of art or jewels, sensitive information in an organization tends to be stored with little or no high-level safeguards and may get forgotten altogether.

Yet another advantage enjoyed by cyber attackers is that they have time to keep trying without being noticed, repeatedly until they succeed. Defenders have been spread thin and rely on early detection systems. These detection systems tend to produce a high number of false positives causing unnecessary 'noise' and prohibiting defenders from knowing whether or not they are under attack. Moreover, many existing detection solutions can be evaded by simply changing malware signatures or hosts, leaving the organization vulnerable.

There is however, one important card in the hand of the defenders, and that is the control of the information in their network. To be more exact, they have the ability to manipulate information in advance of the attackers' arrival. Control and manipulation in this case mean setting up the network in such a way that all is not necessarily what it seems. The organization can manipulate the information the attacker sees by planting false assets and data in the network. This approach is dubbed deception — and its intention is to cause confusion, waste the attackers' time and deflect the attack by sending them down the wrong path. Deception helps keep the attacker away from the real data and additionally facilitates the employment of detection mechanisms that can identify an attack in progress with a high degree of certainty.

Defining Deception

Decoys

The building blocks of internal networks are the assets. These include workstations, servers, laptops, routers, switches, mobile, and other network devices; and the connections and interactions between them. The deception is based on the integration of similar network entities into the real network whose purpose is to mimic the "real" assets and have the same network behavior. These fake assets are called decoys. Decoys are for all intents and purposes no different from any other asset; they will appear to have the same operating systems, the same applications running on the same ports, the same protocols and even similar data in some cases, all depending on the level of interactivity the decoy is given.

The difference between the decoys and the real asset is that legitimate users of the network have no reason to access them, so any access to a decoy should be an indication that an intruder is at work. Like the real assets, the decoy contains data. It looks genuine but is in fact fabricated or even randomly generated and therefore does not increase the attack surface for the organization.

Furthermore, decoys are an excellent means of observing the attacker and learning how it interacts with assets. This way defenders can gather valuable intelligence and forensics about the attack, including methods, purpose, source, etc. By occupying the attacker as much as possible through interaction with the decoy, defenders can also delay the attacker from fulfilling the attack's real purpose.

Whereas older generations of deception used the concept of honeypots, next-generation solutions focus on more than simply making the decoys more attractive, more believable and more strategically located within the network. For next-generation deception to be

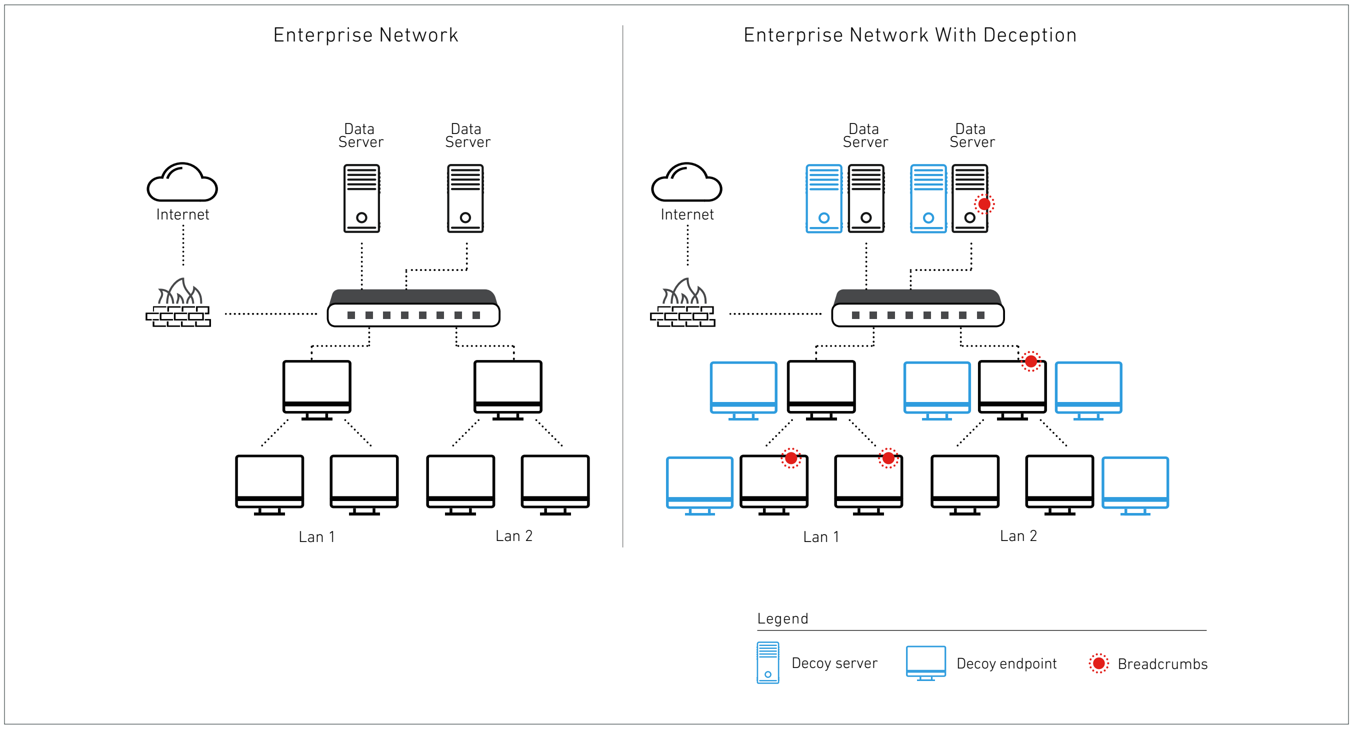


Figure 1: Deception, before and after

effective there needs to be a way to drive the attacker to decoy — making deception deterministic rather than statistic.

Breadcrumbs

Sophisticated attackers track user activities and use information they find on infected machines to get to other resources and assets that might contain valuable data. They look for information that contains references to assets, such as user sessions, credentials, file networking tables and emails. This is something defenders can use to their advantage.

Just as the assets need to be cloned into decoys, so also must these trails and pointers be cloned into their own look-alikes. These false pointers are referred to as "traps" or "breadcrumbs." If the decoys are the false assets, then the breadcrumbs are the false pointers. It is important to understand the different roles between these two deception components — the breadcrumbs serve the crucial role in the deception plan of directing the attackers to detection mechanisms, which may include, for example, decoys, beacon traps and traffic analysis mechanisms, all of which will be discussed in more detail further on in this paper. Breadcrumbs are diverse; they can be files, documents, email messages and system resources — in fact anything on a system or on the network an attacker might look at.

Detection Mechanisms

As the title of this paper suggests, deception is a tool (and a powerful one) in the hands of cyber defenders — but the end-game is quick and accurate detection of advanced persistent attacks, unauthorized access to network assets and data exfiltration. With the concept of deception discussed; now it's time to present the detection mechanisms that deception leads to:

- **Decoy Access** — The first and obvious detection method is decoy access. As shown in figure 2, when an attacker, be it human or machine (malware) taps a decoy, is detected the system sounds the alarm, alerting defenders about the threat.
- **Beacon Traps** — Beacons are a mechanism built into data (files, email, cloud-based accounts, bit-coins, etc.) which send a signal to a pre-defined server every time the data is used. Beacon traps are placed amongst real data in order to aid in the early detection of unauthorized access, copying or modification. Once opened, a beacon trap

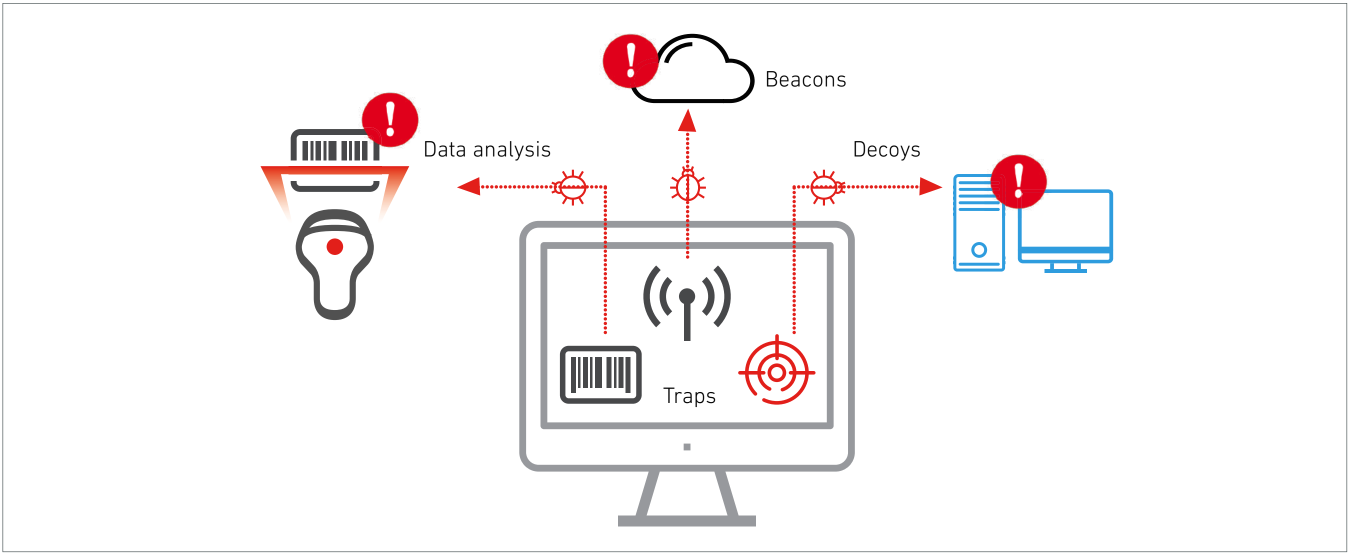


Figure 2: Active deception and detection mechanisms

automatically sends an alert to a pre-defined server (e.g. HTTP request) which indicates that it has been tapped. An example of a beacon can be a file, that once opened, alerts a remote web server.

- **Data Analysis** — Deception is more than luring attackers into decoys. As will be shown further on in this document, deception also includes the use of "poisoned data" that attackers consume and utilize. When the attacker attempts to use the data (be it a username and password, access to an FTP site, etc.) it is detected, alerting the system administrator to things that would otherwise go unnoticed.

Putting Theory to Practice

While understanding deception in theory is a necessary first step, moving to the deployment of decoys and breadcrumbs in real network environments requires further thought, planning and analysis of the current configuration of the network. Questions that will need to be answered in order to deploy decoys effectively include:

- What kind of attacker will be attracted to what different types of traps?
- What decoys should be employed (i.e. which assets, operating systems, applications, etc., in the network need to be mirrored)?
- Where should the decoys be placed?
- What kind of traps should be used?

Due to the wide variation in attacker tactics, there is not one answer. The possibilities are virtually endless and the path to adequately or at least reasonably address different attack styles and preferences is not immediately obvious.

An In-Depth Research of Deception Techniques

For the purpose of the research, three primary questions were formulated:

1. Which deception-based techniques are most effective in real life attack scenarios?
2. Which types of bait (decoys, traps) attract different types of attackers and in what scenarios?
3. Which deployment techniques of the deception layer yield the best security coverage of organizational assets?

These questions are answered by deploying an advanced deception layer in a real-life network scenario. The deception layer was then tested against real attackers such as pen-testers, white-hat hackers and other individuals who are involved in real-world hacking activities, as well as against a wide range of malware types. The following section elaborates more about the methodology of the research.

Methodology

The Environment

The first stage was to build a comprehensive organizational network consisting of servers and workstations running both Linux and Microsoft Windows. The network was connected to a domain with an Microsoft Active Directory server, DNS server, web server, database servers and more.

The next step was to populate the network with "live" content. This included adding users, corporate applications, documents and relevant emails, corporate web applications databases and log files. By the end of this step, a snapshot of a real-world corporate network had been created, complete with assets, users, services and data.

The environment contained:

- 29 Users
- 1,491 Documents
- 5,532 Emails
- 31 application installed
- 3 Full Browser profiles (Chrome, IE, FF)
- 2 Corporate web applications
- 2 Databases
- 1 DC
- 1 DNS Server
- 1 Private cloud service

Creating the Deception

Once the "real" corporate network was ready, the next task was adding the deception layer. This consisted of additional assets based on the structure of the network that were to serve as decoys, and traps to be tested that were put on the real assets and the decoys. On top of the traps and decoys that had proven successful in field deployments, others were added that were in development in the lab, together with several POCs and some promising open source projects.

Relying on prior experience from multiple customer installations, the deception infrastructure was built into the environment taking care to:

- Make it blend in seamlessly so it wouldn't arouse the attackers' suspicion
- Ensure it would be non-intrusive to users so they wouldn't interfere with the deception mechanisms, and that the deception layer wouldn't interfere with their work
- Keep a low attack surface to not create new vulnerabilities into the network

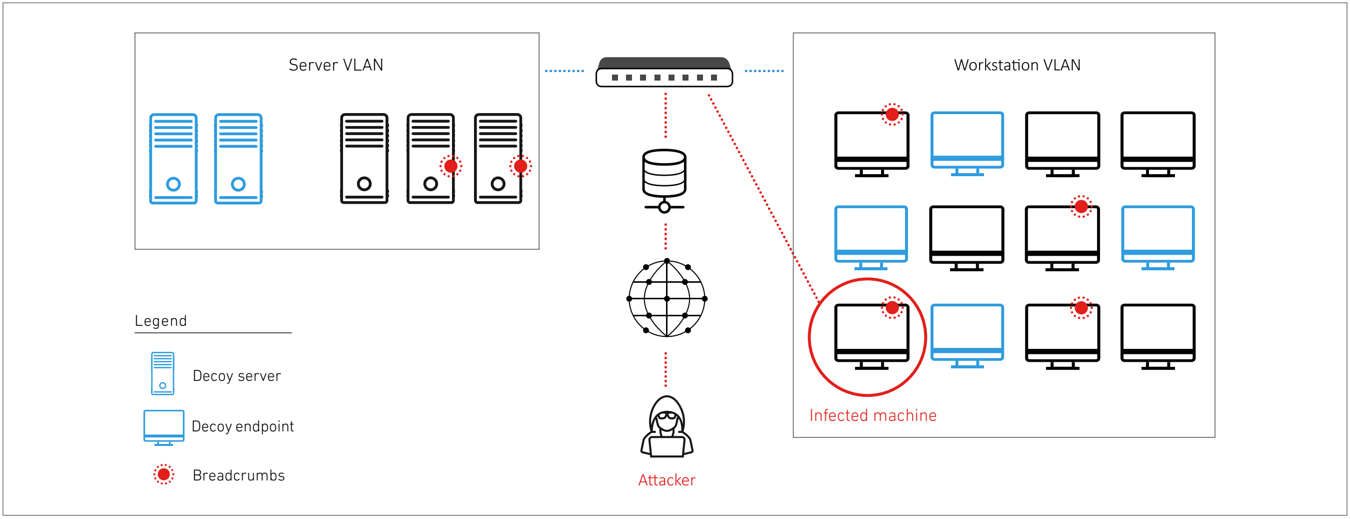


Figure 3: The research network environment

The task was split primarily into building the traps and the decoys, and distributing them throughout the network. In total, the deception layer contained:

- 11 decoys
  - 7 Workstations (user and dev machines running Windows 7)
  - 2 Windows Servers (running Windows 2012 and Windows 2008)
  - 1 Ubuntu Linux server
- 95 decoy services
- Traps including:
  - 61 files
  - 39 beacon traps
  - 27 emails
  - 26 credentials
  - 12 applications
  - 10 IoTs
  - 2 network traps

These are described in more detail below.

Building the Decoys

Decoys are entities designed to mirror the appearance of assets in the organization. To this end the decoys were made to look like and behave like servers, workstations, mobile devices and network devices. From the outside, the decoys appeared exactly like the asset they were mimicking, including the same OS, services and applications. The main difference between the decoys and real assets was that the information inside was not valuable. Since the decoys are not real assets, there is no reason for a legitimate user to access them. Hence, any access to the decoy, especially a highly interactive one, should arouse suspicion.

Decoy Engagement Levels

The decoys were defined with a variation of interactive capabilities. Some decoy services appeared only as open ports, while others were full-blown services, appearing to run real applications. Among the services made available were TCP, UDP, SMB, HTTP, ICMP, RDP, FTP, MYSQL, SMTP and SSH. Decoy services with higher levels of interaction increased the possibility of engagement with the attacker, revealing the attacker's intentions. This in turn increases the probability of accurate identification, slows the attacker down and minimizes false positives.

Building the Traps*

The mini-traps were categorized into 4 distinct types, as shown in Table 1.

Table 1: Traps according to four main types

Files • Documents (.txt, .doc, .xls,.pdf etc.) • Beacon traps • Emails • Logs • Databases • Recent/deleted documents	**Network** • Network table caches poisoning (ARP, DNS, NetBios etc.) • Mounted devices (printers, cameras etc.) • (half) open connection to decoys • Host and ImHost files
Applications • Session apps (SSH, FTD, RDP, clients etc.) • Browsers (history, passwords, bookmarks etc.) • App uninstall information	**Credentials** • Passwords and Hash injections • Windows Credentials Manager • Password Managers

While all categories shared the common function of directing the attacker to a relevant decoy, each one was disguised as or placed in different types of system components.

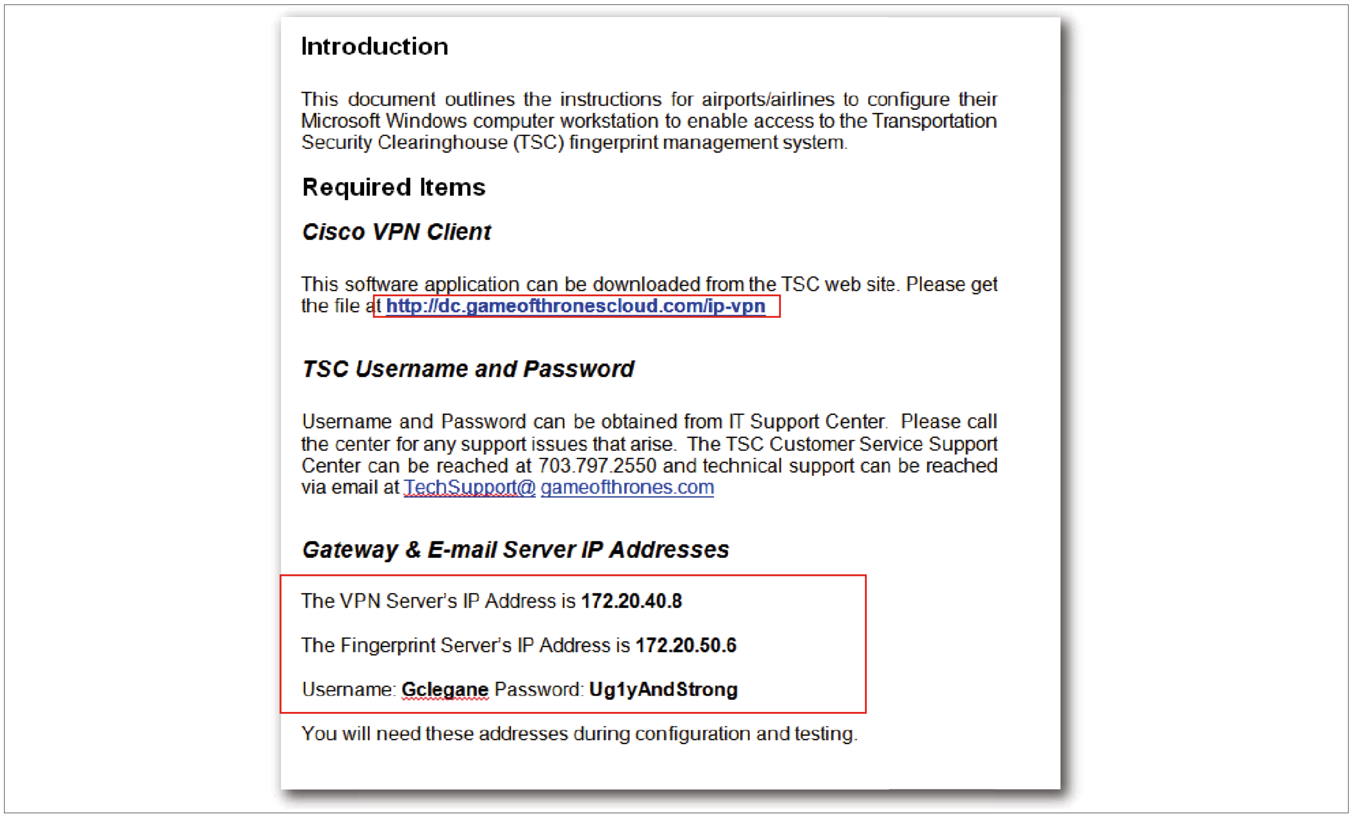
* A few of the traps listed below that were used in this exercise were included to test the efficacy of the solution. These traps may not be currently available in the production version of the product but are in research and development. If you need additional information on them or would like to implement them, please reach out to your sales rep and we can connect you with our product management team.

File-Based Traps

The file trap is the simplest yet most versatile trap out there. Common examples include:

- A text file of some application configuration that contains a username and password
- A technical document common to every organization, such as instructions of how to connect to the corporate VPN
- A personal document of an employee that social engineers might want to use for extortion purposes.
- IT/Corporate documents (txt, doc, xls pdf, etc.)
- Beacon traps within files/emails (both as individual files and within PST files)
- Logs
- Databases
- Recent file lists in Windows, Office and other programs

Figures 4 and 5, below show examples of files that were used in the environment.



Introduction

This document outlines the instructions for airports/airlines to configure their Microsoft Windows computer workstation to enable access to the Transportation Security Clearinghouse (TSC) fingerprint management system.

Required Items

Cisco VPN Client

This software application can be downloaded from the TSC web site. Please get the file at **http://dc.gameofthronescloud.com/ip-vpn**

TSC Username and Password

Username and Password can be obtained from IT Support Center. Please call the center for any support issues that arise. The TSC Customer Service Support Center can be reached at 703.797.2550 and technical support can be reached via email at TechSupport@ gameofthrones.com

Gateway & E-mail Server IP Addresses

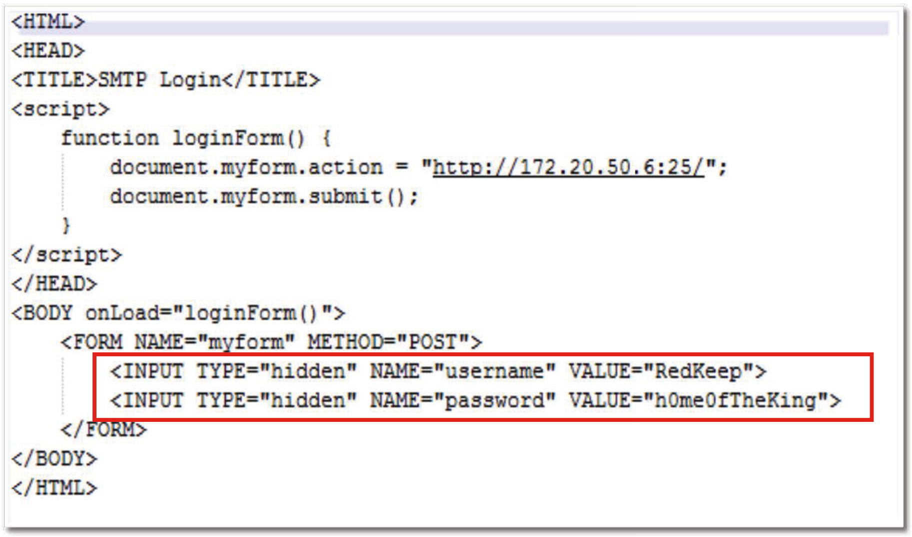
The VPN Server's IP Address is **172.20.40.8**

The Fingerprint Server's IP Address is **172.20.50.6**

Username: **Gclegane** Password: **Ug1yAndStrong**

You will need these addresses during configuration and testing.

Figure 4: A WORD document containing instructions for connecting to a VPN



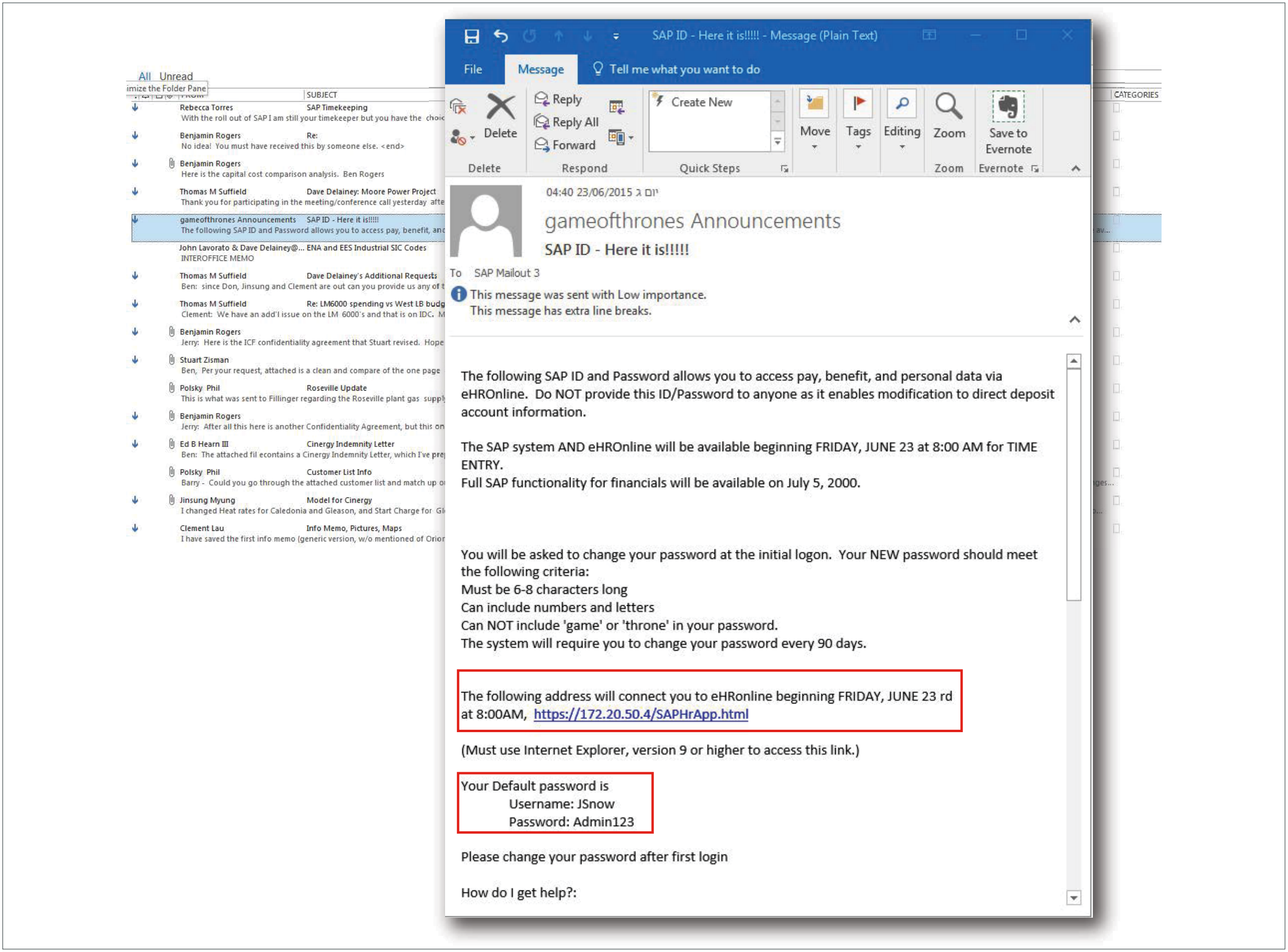
```
<HTML>
<HEAD>
<TITLE>SMTP Login</TITLE>
<script>
    function loginForm() {
        document.myform.action = "http://172.20.50.6:25/";
        document.myform.submit();
    }
</script>
</HEAD>
<BODY onLoad="loginForm()">
    <FORM NAME="myform" METHOD="POST">
        <INPUT TYPE="hidden" NAME="username" VALUE="RedKeep">
        <INPUT TYPE="hidden" NAME="password" VALUE="h0me0fTheKing">
    </FORM>
</BODY>
</HTML>
```

Figure 5: An HTML page containing login details masked from view of the user

Most of these traps contain a pointer of some kind to another location in the network, namely a decoy. Some also include an 'entry key' such as a user name and password; a breadcrumb can even point to a URL on a server outside of the organization. Once the attacker discovers and takes action based on it, the interaction can be monitored.

Email Messages

Email messages have an important role as traps. Despite the ease with which emails can be read, they are still used extensively to transmit sensitive data from one person to another. In other words, emails pose a major vulnerability, one which attackers are well aware of. This affords emails a high degree of credibility (with attackers) and makes them excellent breadcrumbs. The examples in figures 6 and 7 below show an email exchange seemingly shared internally (between employees) which contains what would seem like important information.



FROM	SUBJECT
Rebecca Torres	SAP Timekeeping
With the roll out of SAP I am still your timekeeper but you have the choic	
Benjamin Rogers	Re:
No idea! You must have received this by someone else. <end>	
Benjamin Rogers	
Here is the capital cost comparison analysis. Ben Rogers	
Thomas M Suffield	Dave Delainey: Moore Power Project
Thank you for participating in the meeting/conference call yesterday afte	
gameofthrones Announcements	SAP ID - Here it is!!!!!
The following SAP ID and Password allows you to access pay, benefit, and	
John Lavorato & Dave Delainey@...	ENA and EES Industrial SIC Codes
INTEROFFICE MEMO	
Thomas M Suffield	Dave Delainey's Additional Requests
Ben: since Don, Jinsung and Clement are out can you provide us any of t	
Thomas M Suffield	Re: LM6000 spending vs West LB budg
Clement: We have an add'l issue on the LM 6000's and that is on IDC. M	
Benjamin Rogers	
Jerry: Here is the ICF confidentiality agreement that Stuart revised. Hope	
Stuart Zisman	
Ben, Per your request, attached is a clean and compare of the one page	
Polsky Phil	Roseville Update
This is what was sent to Fillinger regarding the Roseville plant gas suppl	
Benjamin Rogers	
Jerry: After all this here is another Confidentiality Agreement, but this on	
Ed B Hearn III	Cinergy Indemnity Letter
Ben: The attached fil econtains a Cinergy Indemnity Letter, which I've pre	
Polsky Phil	Customer List Info
Barry - Could you go through the attached customer list and match up o	
Jinsung Myung	Model for Cinergy
I changed Heat rates for Caledonia and Gleason, and Start Charge for Gl	
Clement Lau	Info Memo, Pictures, Maps
I have saved the first info memo (generic version, w/o mentioned of Orion	

04:40 23/06/2015 יום ג

gameofthrones Announcements

SAP ID - Here it is!!!!!

To SAP Mailout 3

This message was sent with Low importance.
This message has extra line breaks.

The following SAP ID and Password allows you to access pay, benefit, and personal data via eHROnline. Do NOT provide this ID/Password to anyone as it enables modification to direct deposit account information.

The SAP system AND eHROnline will be available beginning FRIDAY, JUNE 23 at 8:00 AM for TIME ENTRY.
Full SAP functionality for financials will be available on July 5, 2000.

You will be asked to change your password at the initial logon. Your NEW password should meet the following criteria:
Must be 6-8 characters long
Can include numbers and letters
Can NOT include 'game' or 'throne' in your password.
The system will require you to change your password every 90 days.

The following address will connect you to eHRonline beginning FRIDAY, JUNE 23 rd at 8:00AM, https://172.20.50.4/SAPHrApp.html

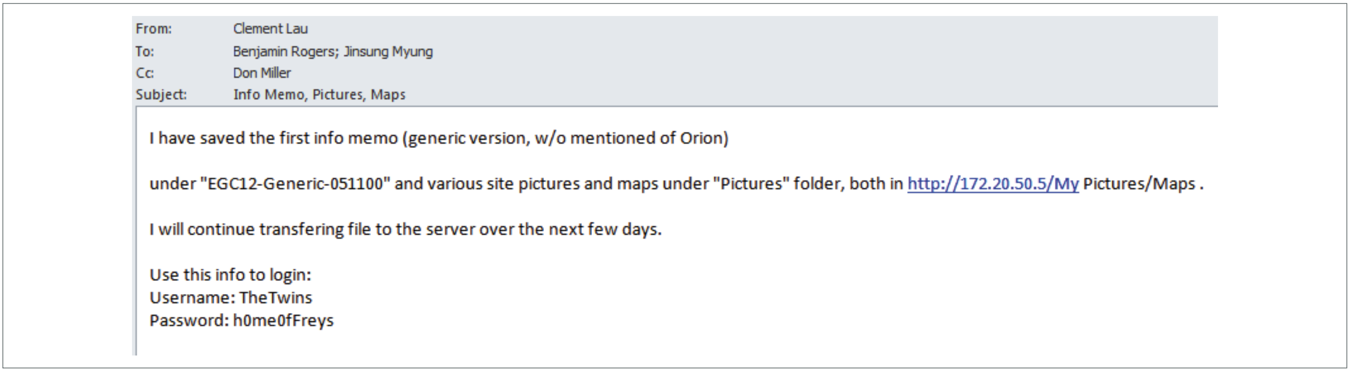
(Must use Internet Explorer, version 9 or higher to access this link.)

Your Default password is
Username: JSnow
Password: Admin123

Please change your password after first login

How do I get help?:

Figure 6: An email trap containing false credentials



From:	Clement Lau
To:	Benjamin Rogers; Jinsung Myung
Cc:	Don Miller
Subject:	Info Memo, Pictures, Maps

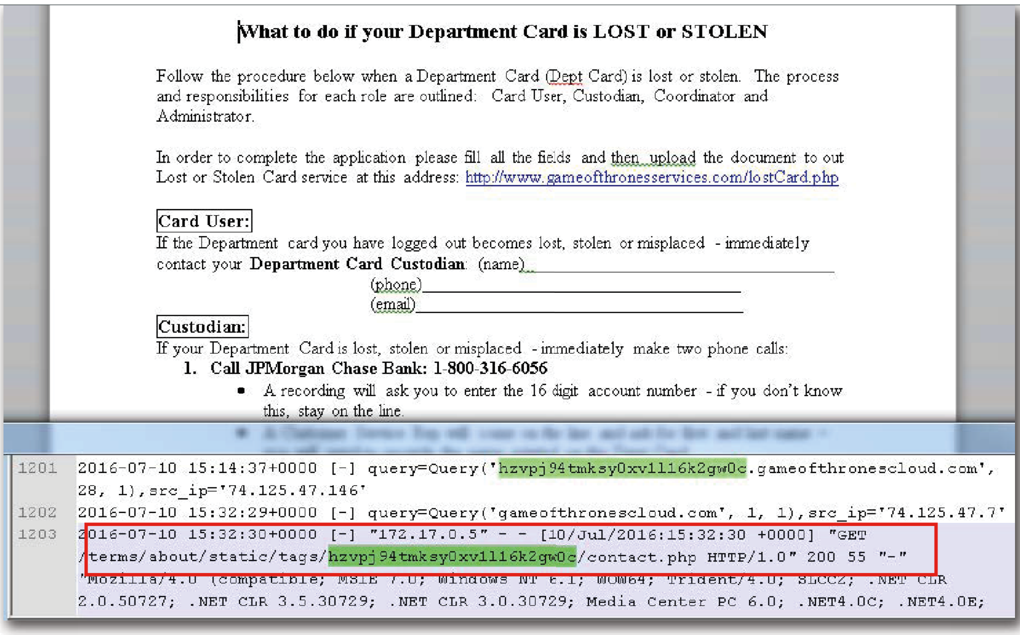
I have saved the first info memo (generic version, w/o mentioned of Orion)

under "EGC12-Generic-051100" and various site pictures and maps under "Pictures" folder, both in http://172.20.50.5/My Pictures/Maps .

I will continue transfering file to the server over the next few days.

Use this info to login:
Username: TheTwins
Password: h0me0fFreys

Figure 7: An email trap containing false credentials



What to do if your Department Card is LOST or STOLEN

Follow the procedure below when a Department Card (Dept Card) is lost or stolen. The process and responsibilities for each role are outlined: Card User, Custodian, Coordinator and Administrator.

In order to complete the application please fill all the fields and then upload the document to out Lost or Stolen Card service at this address: http://www.gameofthronesservices.com/lostCard.php

Card User:
If the Department card you have logged out becomes lost, stolen or misplaced - immediately contact your **Department Card Custodian** (name)____________________
(phone)____________________
(email)____________________

Custodian:
If your Department Card is lost, stolen or misplaced - immediately make two phone calls:

1. **Call JPMorgan Chase Bank: 1-800-316-6056**
   - A recording will ask you to enter the 16 digit account number - if you don't know this, stay on the line.

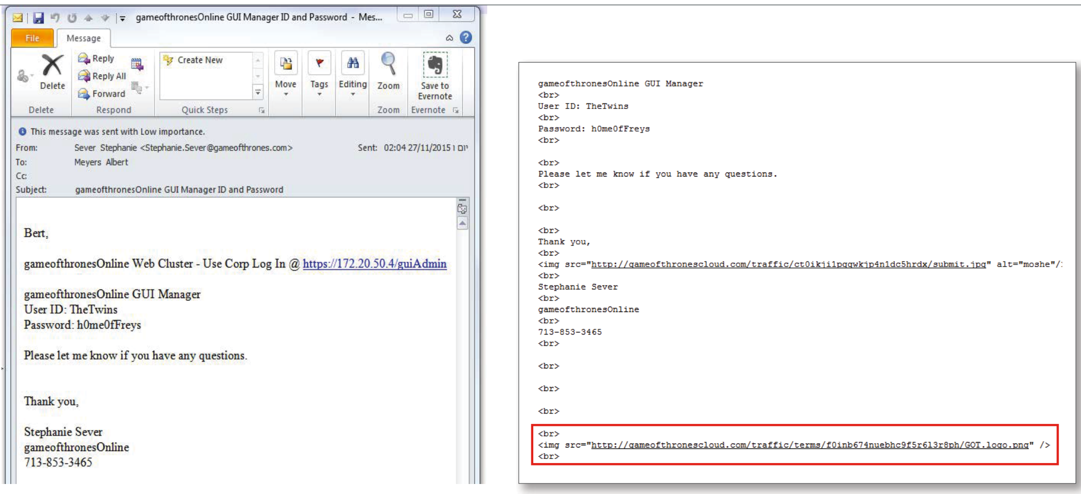
```
2016-07-10 15:14:37+0000 [-] query=Query('hzvpj94tmksy0xv1l16k2gw0c.gameofthronescloud.com',
28, 1),src_ip='74.125.47.146'
2016-07-10 15:32:29+0000 [-] query=Query('gameofthronescloud.com', 1, 1),src_ip='74.125.47.7'
2016-07-10 15:32:30+0000 [-] "172.17.0.5" - - [10/Jul/2016:15:32:30 +0000] "GET
/terms/about/static/tags/hzvpj94tmksy0xv1l16k2gw0c/contact.php HTTP/1.0" 200 55 "-"
"Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.1; WOW64; Trident/4.0; SLCC2; .NET CLR
2.0.50727; .NET CLR 3.5.30729; .NET CLR 3.0.30729; Media Center PC 6.0; .NET4.0C; .NET4.0E;
```

Figure 8: A WORD file document containing a Canary type beacon[3]

Beacon Traps

Beacon traps, such as Canary files, are one more method for trapping attackers. As already mentioned, a beacon is a document or email that once opened, automatically creates a network request to a pre-defined server. Each beacon trap contains a unique token, allowing defenders to know exactly which file was opened and from which asset it originated, even if the file was already exfiltrated from the organization.

Another implementation of Beacons is in an email message. Figure 9 below shows an example of an email with an embedded HTML img tag in it. The image is invisible to the user (1x1 white pixel) and is downloaded from the server. Each time the email is opened with an html supported client, it sends back a ping and raises an alert.



This message was sent with Low importance.
From: Sever Stephanie <Stephanie.Sever@gameofthrones.com>
Sent: 02:04 27/11/2015 ו יום
To: Meyers Albert
Cc:
Subject: gameofthronesOnline GUI Manager ID and Password

Bert,

gameofthronesOnline Web Cluster - Use Corp Log In @ https://172.20.50.4/guiAdmin

gameofthronesOnline GUI Manager
User ID: TheTwins
Password: h0me0fFreys

Please let me know if you have any questions.

Thank you,

Stephanie Sever
gameofthronesOnline
713-853-3465

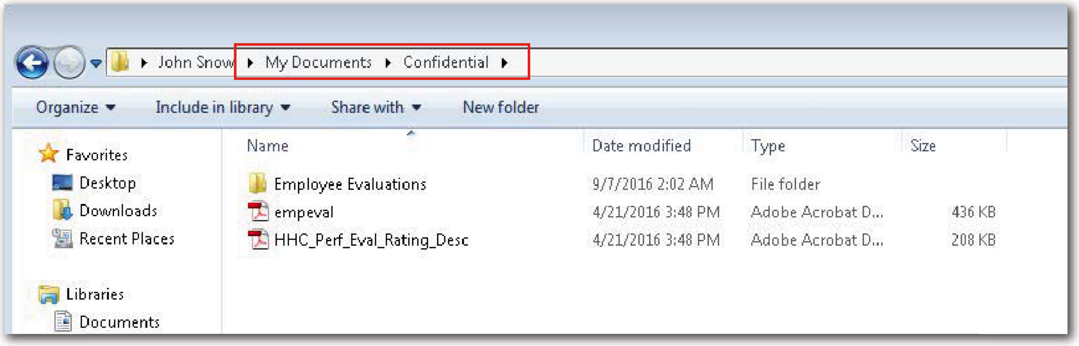
```
gameofthronesOnline GUI Manager
<br>
User ID: TheTwins
<br>
Password: h0me0fFreys
<br>

<br>
Please let me know if you have any questions.
<br>

<br>

<br>
Thank you,
<br>
<img src="http://gameofthronescloud.com/traffic/ct0ikji1pgqwkjp4n1dc5hrdx/submit.jpg" alt="moshe"/:
<br>
Stephanie Sever
<br>
gameofthronesOnline
<br>
713-853-3465
<br>

<br>

<br>

<br>

<br>
<img src="http://gameofthronescloud.com/traffic/terms/f0inb674nuebhc9f5r613r8ph/GOT.logo.png" />
<br>
```

Figure 9: An email message containing a beacon as it appears to the attacker and the HTML within the email message

[3] The research project provided an opportunity to test out one of the more exciting projects currently in progress, by thinkst OS (www.thinkst.com)

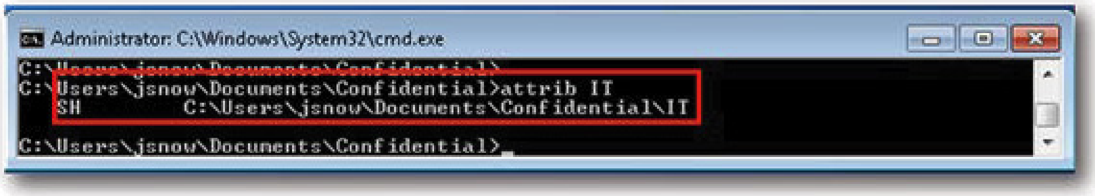
Permissions and System

One efficient way to utilize file-based traps is to take advantage of the operating system's file permission system. Using the file permission system gives two advantages: one, hiding the traps from everyday users who typically view the folder (in other words, limiting access to only privileged users), thus significantly reducing the chance of the user inadvertently opening the covert traps and triggering a false positive; the second, it is used as a powerful lure, whetting an attacker's appetite simply by being restricted, but more importantly — it immediately triggers a high alert as it signifies that the attacker has gained high privilege access rights.


John Snow ▸ My Documents ▸ Confidential ▸
Organize ▾ Include in library ▾ Share with ▾ New folder
Favorites
Desktop
Downloads
Recent Places
Libraries
Documents

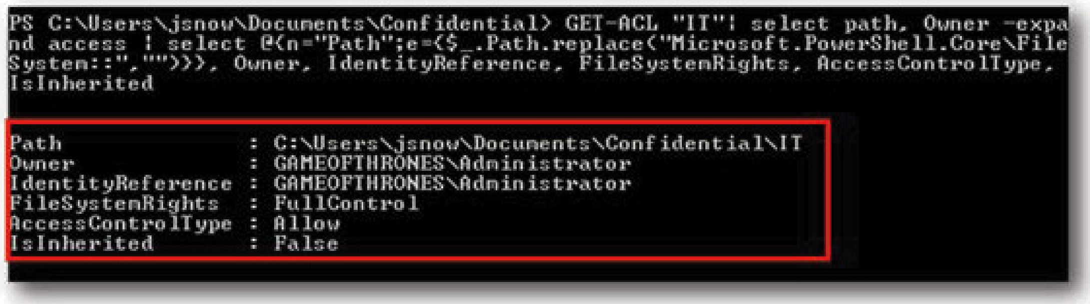
Name	Date modified	Type	Size
Employee Evaluations	9/7/2016 2:02 AM	File folder	
empeval	4/21/2016 3:48 PM	Adobe Acrobat D...	436 KB
HHC_Perf_Eval_Rating_Desc	4/21/2016 3:48 PM	Adobe Acrobat D...	208 KB

Figure 10a: The OS file permission system as the user sees it



```
C:\Users\jsnow\Documents\Confidential>
C:\Users\jsnow\Documents\Confidential>attrib IT
SH         C:\Users\jsnow\Documents\Confidential\IT

C:\Users\jsnow\Documents\Confidential>_
```

Figure 10b: The OS file permission system — the library is hidden



```
PS C:\Users\jsnow\Documents\Confidential> GET-ACL "IT"| select path, Owner -expa
nd access | select @{n="Path";e={$_.Path.replace("Microsoft.PowerShell.Core\File
System::","")}}, Owner, IdentityReference, FileSystemRights, AccessControlType,
IsInherited

Path              : C:\Users\jsnow\Documents\Confidential\IT
Owner             : GAMEOFTHRONES\Administrator
IdentityReference : GAMEOFTHRONES\Administrator
FileSystemRights  : FullControl
AccessControlType : Allow
IsInherited       : False
```

Figure 10c: The OS file permission system — the library is locked by permissions for domain admin only access

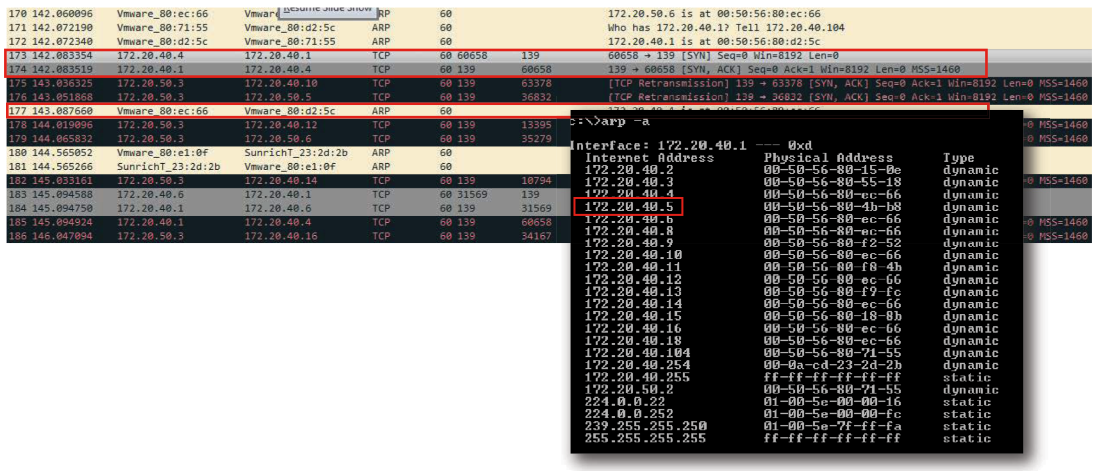
Network-Based Traps

The second category of traps centers around network-related content. This research focuses on four main types of network-based traps:

- Network table cache (ARP, DNS, Netbios)
- Mounted devices (network printers, cameras)
- Open connection to decoys
- Host and lmHost files

ARP Table Poisoning

One way deception can be added to the ARP table is by inserting entries into it that point to a decoy. Since static entries are not very authentic it is better to insert entries into the ARP table in a way that will seem as if the infected computer has actually been in contact with the destination computer. One way of doing that is by sending spoofed TCP SYN packets from the decoy to the asset. The spoofed packet causes the asset to return a SYN ACK (of RST if the port is closed), followed by an ARP request so the asset will know which MAC to return the packet to. An ARP reply is then sent with the decoy MAC back to the asset, triggering the asset to insert that MAC/IP into its ARP cache. The replacement of older entries in the cache by new ones necessitates repeating the process constantly to ensure the ARP table always contains the decoy's addresses.



```
170 142.060096   Vmware_80:ec:66    Vmware           RP    60                  172.20.50.6 is at 00:50:56:80:ec:66
171 142.072190   Vmware_80:71:55    Vmware_80:d2:5c   ARP   60                  Who has 172.20.40.1? Tell 172.20.40.104
172 142.072340   Vmware_80:d2:5c    Vmware_80:71:55   ARP   60                  172.20.40.1 is at 00:50:56:80:d2:5c
173 142.083354   172.20.40.4        172.20.40.1       TCP   60 60658   139      60658 → 139 [SYN] Seq=0 Win=8192 Len=0
174 142.083519   172.20.40.1        172.20.40.4       TCP   60 139     60658    139 → 60658 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1460
175 143.036325   172.20.50.3        172.20.40.10      TCP   60 139     63378    [TCP Retransmission] 139 → 63378 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1460
176 143.051868   172.20.50.3        172.20.50.5       TCP   60 139     36832    [TCP Retransmission] 139 → 36832 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1460
177 143.087660   Vmware_80:ec:66    Vmware_80:d2:5c   ARP   60
178 144.019096   172.20.50.3        172.20.40.12      TCP   60 139     13395                                                                     =0 MSS=1460
179 144.065832   172.20.50.3        172.20.50.6       TCP   60 139     35279                                                                     =0 MSS=1460
180 144.565052   Vmware_80:e1:0f    SunrichT_23:2d:2b ARP   60
181 144.565266   SunrichT_23:2d:2b  Vmware_80:e1:0f   ARP   60
182 145.033161   172.20.50.3        172.20.40.14      TCP   60 139     10794                                                                     =0 MSS=1460
183 145.094588   172.20.40.6        172.20.40.1       TCP   60 31569   139
184 145.094750   172.20.40.1        172.20.40.6       TCP   60 139     31569
185 145.094924   172.20.40.1        172.20.40.4       TCP   60 139     60658                                                                     =0 MSS=1460
186 146.047094   172.20.50.3        172.20.40.16      TCP   60 139     34167                                                                     =0 MSS=1460
```

```
c:\>arp -a

Interface: 172.20.40.1 --- 0xd
  Internet Address      Physical Address      Type
  172.20.40.2           00-50-56-80-15-0e     dynamic
  172.20.40.3           00-50-56-80-55-18     dynamic
  172.20.40.4           00-50-56-80-ec-66     dynamic
  172.20.40.5           00-50-56-80-4b-b8     dynamic
  172.20.40.6           00-50-56-80-ec-66     dynamic
  172.20.40.8           00-50-56-80-ec-66     dynamic
  172.20.40.9           00-50-56-80-f2-52     dynamic
  172.20.40.10          00-50-56-80-ec-66     dynamic
  172.20.40.11          00-50-56-80-f8-4b     dynamic
  172.20.40.12          00-50-56-80-ec-66     dynamic
  172.20.40.13          00-50-56-80-f9-fc     dynamic
  172.20.40.14          00-50-56-80-ec-66     dynamic
  172.20.40.15          00-50-56-80-18-8b     dynamic
  172.20.40.16          00-50-56-80-ec-66     dynamic
  172.20.40.18          00-50-56-80-ec-66     dynamic
  172.20.40.104         00-50-56-80-71-55     dynamic
  172.20.40.254         00-0a-cd-23-2d-2b     dynamic
  172.20.40.255         ff-ff-ff-ff-ff-ff     static
  172.20.50.2           00-50-56-80-71-55     dynamic
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.252           01-00-5e-00-00-fc     static
  239.255.255.250       01-00-5e-7f-ff-fa     static
  255.255.255.255       ff-ff-ff-ff-ff-ff     static
```

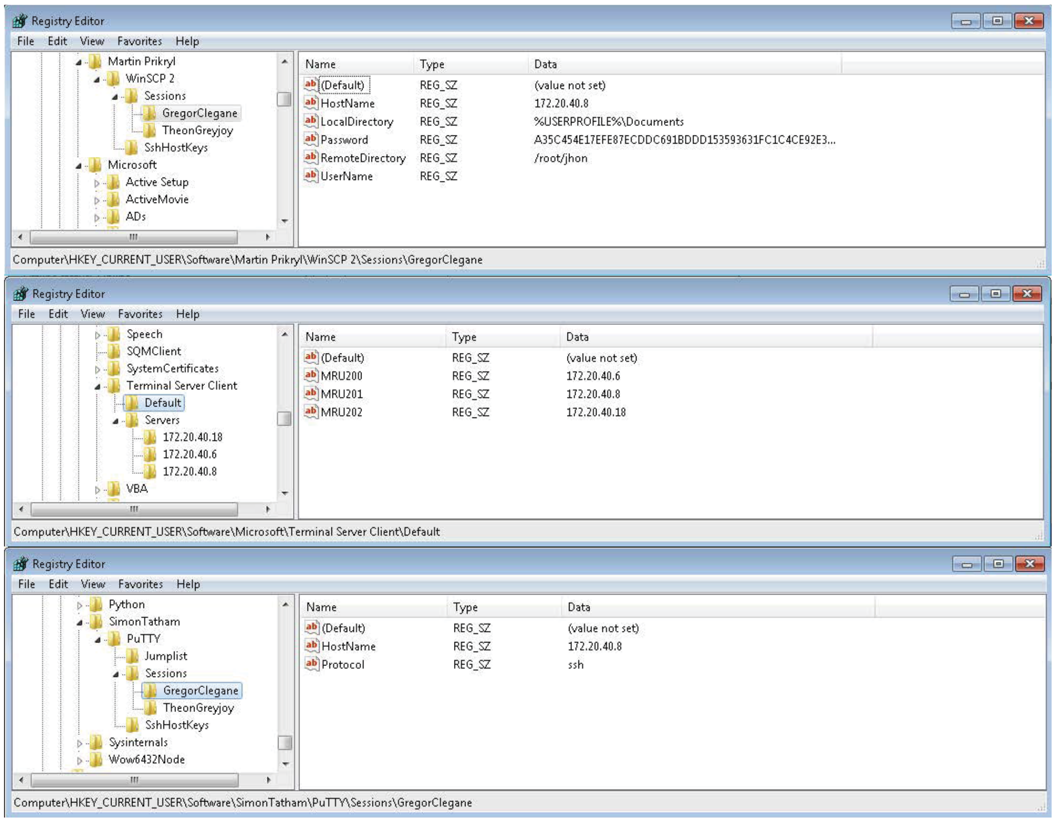
Figure 11: Using the ARP table as a lure to deceive attackers

Application-Based Traps

The third category of file traps centers around applications, or to be more specific, data related to those applications, such as:

- Session apps (SSH, FTP, RDP clients, etc.)
- Browsers (history, passwords, bookmarks)
- App uninstall information

Every application that saves some kind of user information can be used as a trap. The most obvious ones are those that access remote computers, such as FTP, RDP and SSH clients, password managers and browsers. Traps can be related to applications that are not currently installed, which helps hide them from the users to avoid false positives. Basing traps on uninstalled programs is valid as uninstalling leaves behind "digital residue" computer resources used by many applications. Nevertheless, the most authentic application trap is one related to a currently installed and working app. Below are some examples of application traps that were set in the registry for WinScp, Windows MSTC, and PUTTY.



Name	Type	Data
(Default)	REG_SZ	(value not set)
HostName	REG_SZ	172.20.40.8
LocalDirectory	REG_SZ	%USERPROFILE%\Documents
Password	REG_SZ	A35C454E17EFE87ECDDC691BDDD153593631FC1C4CE92E3...
RemoteDirectory	REG_SZ	/root/jhon
UserName	REG_SZ	

Computer\HKEY_CURRENT_USER\Software\Martin Prikryl\WinSCP 2\Sessions\GregorClegane

Name	Type	Data
(Default)	REG_SZ	(value not set)
MRU200	REG_SZ	172.20.40.6
MRU201	REG_SZ	172.20.40.8
MRU202	REG_SZ	172.20.40.18

Computer\HKEY_CURRENT_USER\Software\Microsoft\Terminal Server Client\Default

Name	Type	Data
(Default)	REG_SZ	(value not set)
HostName	REG_SZ	172.20.40.8
Protocol	REG_SZ	ssh

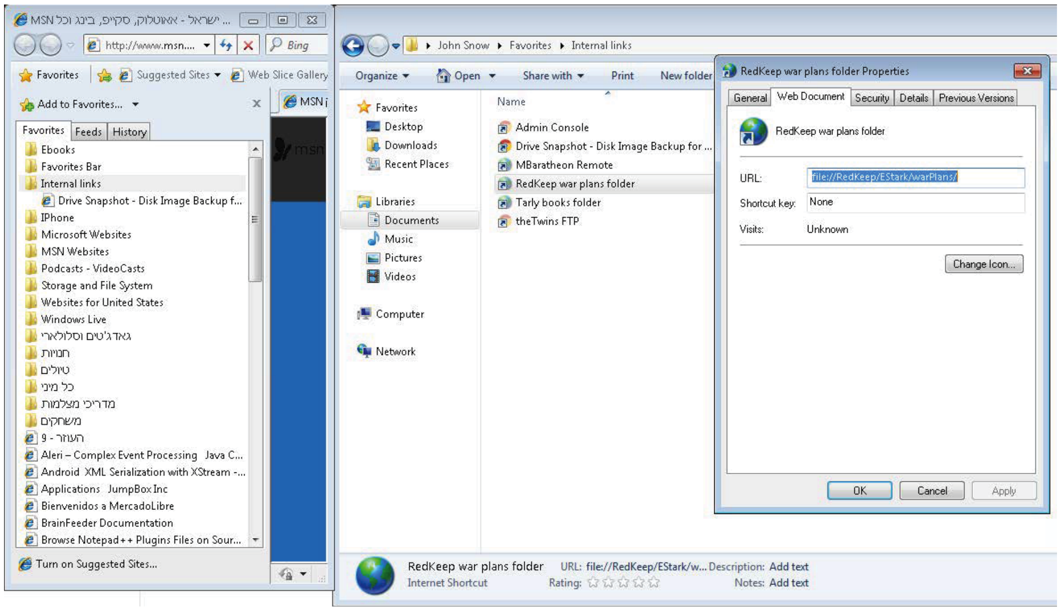
Computer\HKEY_CURRENT_USER\Software\SimonTatham\PuTTY\Sessions\GregorClegane

Figure 12: Examples of application traps

Browsers

Browser history and bookmarks can be used to plant some very specific information for attackers to find. For example, the favorites in Internet Explorer are kept as individual .url files in the folder documents/favorites. If the trap destination file is hidden, it will be hidden in the browser as well — invisible to innocent users to avoid false positives. These methods work well for automatic scanners and file scrapers.

Chrome and Firefox are slightly trickier. These browsers use SQLite DB to save both password data and browsing history. Chrome saves its history in the “urls” table. There is a column called “hidden” that does all the work, as shown below in figure 14.


John Snow ▸ Favorites ▸ Internal links

Name
Admin Console
Drive Snapshot - Disk Image Backup for ...
MBaratheon Remote
RedKeep war plans folder
Tarly books folder
theTwins FTP

RedKeep war plans folder Properties

RedKeep war plans folder

URL: file://RedKeep/EStark/warPlans/

Shortcut key: None

Visits: Unknown

RedKeep war plans folder URL: file://RedKeep/EStark/w... Description: Add text
Internet Shortcut Rating: Notes: Add text

Figure 13: Using Internet Explorer Favorites to set traps

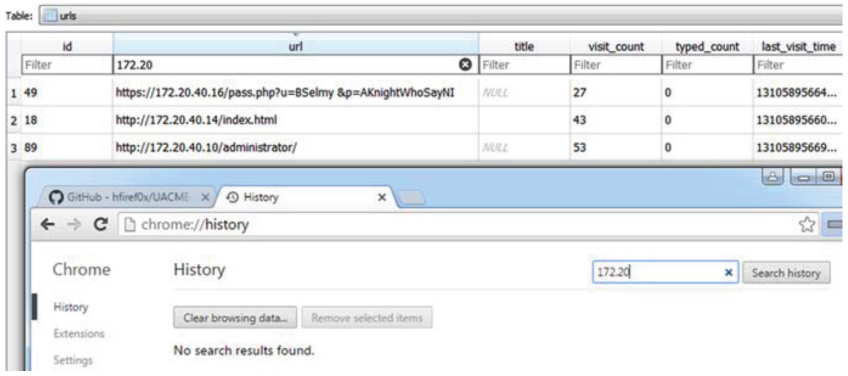


Table: urls

	id	url	title	visit_count	typed_count	last_visit_time
	Filter	172.20	Filter	Filter	Filter	Filter
1	49	https://172.20.40.16/pass.php?u=BSelmy &p=AKnightWhoSayNI	NULL	27	0	13105895664...
2	18	http://172.20.40.14/index.html		43	0	13105895660...
3	89	http://172.20.40.10/administrator/	NULL	53	0	13105895669...

chrome://history

History

172.20

No search results found.

Figure 14: Using Chrome Favorites to set traps

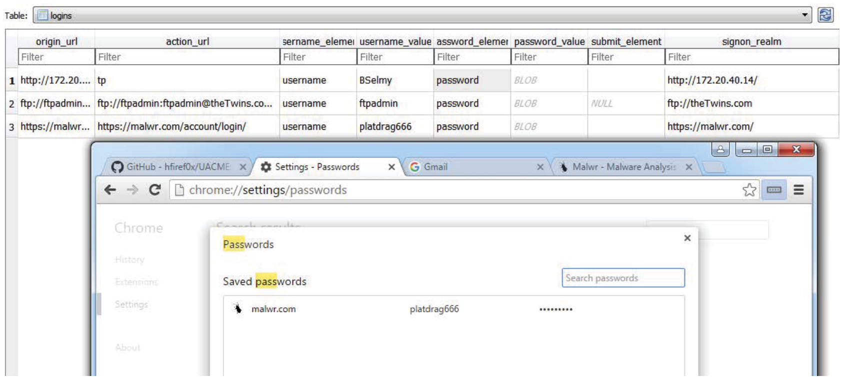


Table: logins

	origin_url	action_url	sername_elemer	username_value	assword_elemer	password_value	submit_element	signon_realm
	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	http://172.20....	tp	username	BSelmy	password	BLOB		http://172.20.40.14/
2	ftp://ftpadmin...	ftp://ftpadmin:ftpadmin@theTwins.co...	username	ftpadmin	password	BLOB	NULL	ftp://theTwins.com
3	https://malwr...	https://malwr.com/account/login/	username	platdrag666	password	BLOB		https://malwr.com/

Passwords

Saved passwords

Search passwords

malwr.com	platdrag666	••••••••••

Figure 15: Chrome Saved Passwords

Which Software Applications do Attackers Target?

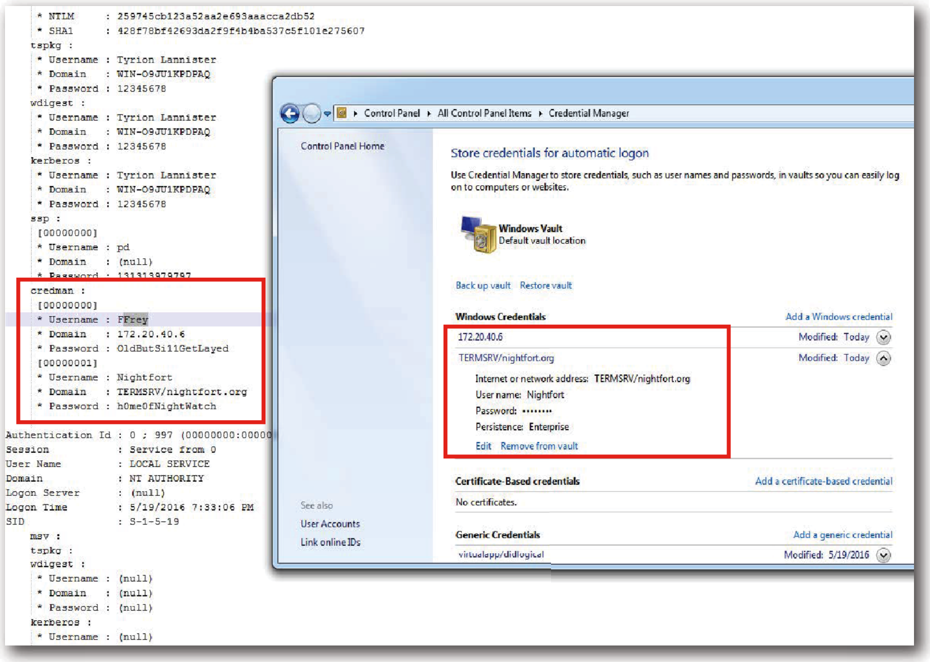
We know that different malware applications look for different data so we needed to diversify our resources to cover as large a scope as possible. Fortunately, assistance came from the unlikely source of the malware authors themselves. As it turns out, the source code for many Trojans is leaked to the internet. Perhaps the best known example is the PONY Trojan. There are more than 200 known applications that are repeatedly monitored by a host of malware programs.

Traps in Credentials

System Credentials are the fourth category of traps built into the test environment. Credentials can include:

- Passwords and hash injections
- Windows Credential Manager
- Password managers

Perhaps the most interesting of all are the key stores such as the windows lsass and Windows Credential Manager. Windows Credential Manager (credman) is by no means a secure method for saving passwords,



```
   * NTLM     : 259745cb123a52aa2e693aaacca2db52
   * SHA1     : 428f78bf42693da2f9f4b4ba537c5f101e275607
  tspkg :
   * Username : Tyrion Lannister
   * Domain   : WIN-O9JU1KPDPAQ
   * Password : 12345678
  wdigest :
   * Username : Tyrion Lannister
   * Domain   : WIN-O9JU1KPDPAQ
   * Password : 12345678
  kerberos :
   * Username : Tyrion Lannister
   * Domain   : WIN-O9JU1KPDPAQ
   * Password : 12345678
  ssp :
   [00000000]
   * Username : pd
   * Domain   : (null)
   * Password : 131313979797
  credman :
   [00000000]
   * Username : FFrey
   * Domain   : 172.20.40.6
   * Password : OldButSillGetLayed
   [00000001]
   * Username : Nightfort
   * Domain   : TERMSRV/nightfort.org
   * Password : h0me0fNightWatch
Authentication Id : 0 ; 997 (00000000:00000
Session           : Service from 0
User Name         : LOCAL SERVICE
Domain            : NT AUTHORITY
Logon Server      : (null)
Logon Time        : 5/19/2016 7:33:06 PM
SID               : S-1-5-19
  msv :
  tspkg :
  wdigest :
   * Username : (null)
   * Domain   : (null)
   * Password : (null)
  kerberos :
   * Username : (null)
```

Store credentials for automatic logon

Use Credential Manager to store credentials, such as user names and passwords, in vaults so you can easily log on to computers or websites.

Windows Credentials

172.20.40.6 — Modified: Today

TERMSRV/nightfort.org — Modified: Today

Internet or network address: TERMSRV/nightfort.org
User name: Nightfort
Password: ••••••••
Persistence: Enterprise

Certificate-Based credentials

No certificates.

Generic Credentials

virtualapp/didlogical — Modified: 5/19/2016

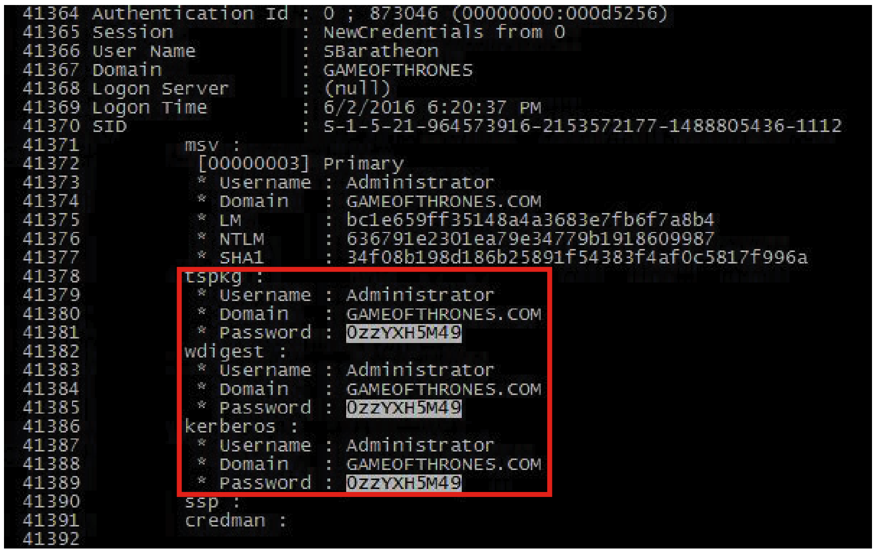
Figure 16: Using Windows Credential Manager for setting traps

but it offers excellent opportunities for setting up traps. It can contain any generic service with an IP and port along with a username and password — for example, RDP locations are very common in credman as they're being saved there by default. Attackers can then dump the credman passwords using a tool such as mimikatz. Since local administrator privileges are usually needed for commencing the attack, it requires the attacker to work a bit harder, which in turn makes the trap appear more lucrative. Figure 16 shows an example of a trap set within Windows Credentials Manager.

The lsass mechanism can also be adjusted to create a record of a user that has logged in to the asset. False credentials, tickets and hashes were injected into the lsass (shown in figure 17 below). For this DCEPT was used, an open source project that was recently released by DELL Secureworks.[4] DCEPT places honeytoken credentials into the memory by calling the CreateProcessWithLogonW Windows API to launch a suspended sub-process with the LOGON_NETCREDENTIALS_ONLY flag.

After successfully finding the domain admin password in plain text the attacker would attempt to log on to the system — something that goes through the domain controller if one exists.

It's important to note that each request was sent on the network to the domain controller (DC) and was either sniffed on the LAN or found in the DC logs (assuming the deception layer has such capabilities). The password hash was found in the network traffic and could be brute-forced quite easily as the plaintext was known, and this was enough to confirm that the attacker used the planted (fake) password. Each asset/user was given a unique password, making it simple for the research team to know exactly where it was from and ensure it was not already in use as a legitimate password.



```
Authentication Id : 0 ; 873046 (00000000:000d5256)
Session           : NewCredentials from 0
User Name         : SBaratheon
Domain            : GAMEOFTHRONES
Logon Server      : (null)
Logon Time        : 6/2/2016 6:20:37 PM
SID               : S-1-5-21-964573916-2153572177-1488805436-1112
        msv :
         [00000003] Primary
         * Username : Administrator
         * Domain   : GAMEOFTHRONES.COM
         * LM       : bc1e659ff35148a4a3683e7fb6f7a8b4
         * NTLM     : 636791e2301ea79e34779b1918609987
         * SHA1     : 34f08b198d186b25891f54383f4af0c5817f996a
        tspkg :
         * Username : Administrator
         * Domain   : GAMEOFTHRONES.COM
         * Password : 0zzYXH5M49
        wdigest :
         * Username : Administrator
         * Domain   : GAMEOFTHRONES.COM
         * Password : 0zzYXH5M49
        kerberos :
         * Username : Administrator
         * Domain   : GAMEOFTHRONES.COM
         * Password : 0zzYXH5M49
        ssp :
        credman :
```

Figure 17: Adjusting the logon mechanism

[4] See: www.secureworks.com/blog/dcept

Recruiting the Attackers

The objective of the research was to examine the effectiveness of deception techniques against a variety of attacks, in a contained, yet realistic environment. For this purpose, experienced hackers were recruited to test the deception layer against sophisticated human attacks. In addition, the same environment was observed with the introduction of different kinds of malware including crimeware, RATs, and Trojans.

The CTF Challenge

Once the environment and the challenge were in place (as described in the chapters above), the research team posted a public invitation in popular hacker forums, groups and websites. The invitation called for the best people in the field — red teams, pen testers and security researchers — to take part in the research. After filtering the respondents, those that fitted the required profile received a written brief of the mission and a clear definition of the goal.

For the purpose of the research a "capture the flag" (CTF) challenge was created. This included placing valuable pieces of data on various assets in the network environment that served as hints for the participants, without which they could not complete the challenge. Participants were given access to one asset through a RAT (remote access Trojan). From that starting point, they had to reach their objective which required piecing together 5 items of information that were spread across different assets in the organization. The task required the attacker to gather intelligence, collect information and move laterally in the network, knowing that traps and decoys were built into the system but not knowing what they were.

Each attacker or team worked exclusively on a freshly initialized instance of the environment to avoid interference. In all, 52 information security professionals from across the world took part in the challenge, spending an average of 6-7 hours each in an attempt to solve the task. The attack challenge ran for more than a month, while in parallel, the environment was tested against several dozen malware types.

Findings

The Knowledge Gap

"Knowledge gap" is a term given to a phenomenon observed in nearly all of the attacks that took place in the framework of the research. It refers to the difference between what the attacker thinks he or she knows and reality. This knowledge — or lack thereof — includes the current location, the network structure, the nature of the assets, the target destination and other characteristics.

After participants, or attackers, were given access to one asset through a RAT, they were informed that the first of 5 file hashes needed to be found and extracted in order to solve the quest, and it was on the initial infected machine (i.e. on the asset) they were currently on.

While it was a relatively simple first task, requiring no special technical skills, participants needed an average of 600 shell commands to find and take the hash for the first file. In comparison, the subsequent phases were considerably more technical and required progressively higher levels of expertise to pass, but participants used progressively fewer commands as they advanced. On average participants needed only 370 commands to solve the 2nd phase, and 120 commands for the 3rd phase. There was a slight relapse on phase 4, which averaged 132 commands, but it was a considerably greater technical challenge.

The participants who reached phase 5, used an average of just 14 shell commands. This is depicted in figure 18 below.

In other words, the more attackers progressed in their mission, the more they knew about the environment, including current location, where the targets are, what

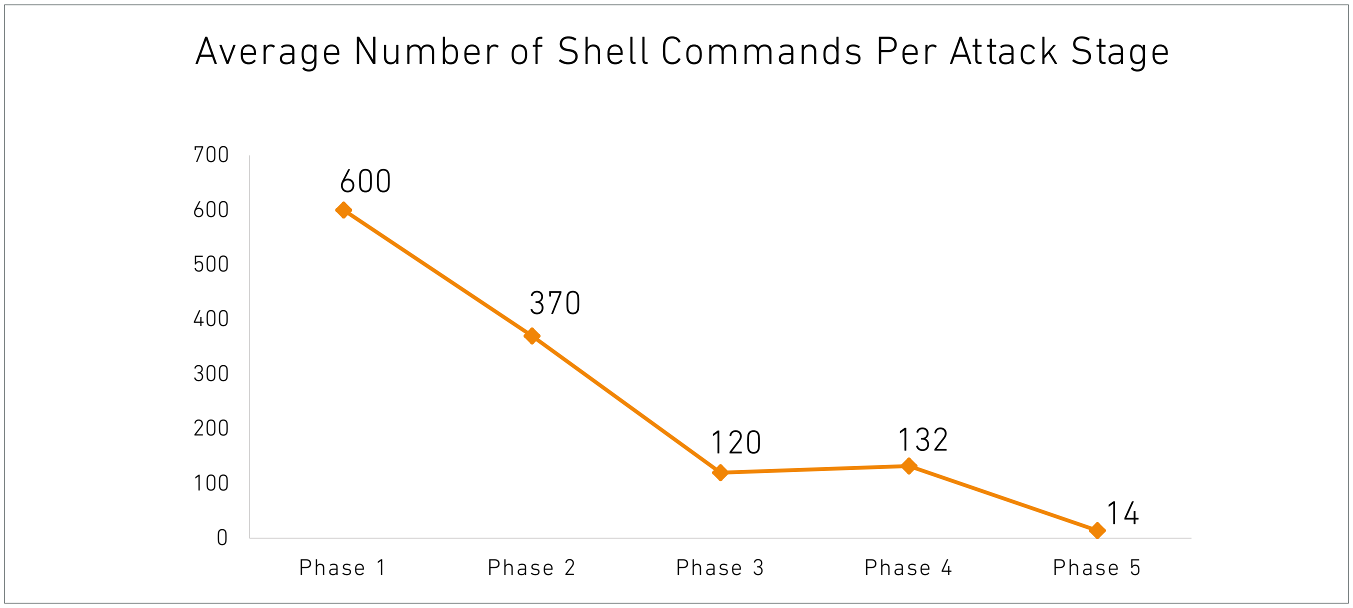


Figure 18: The Knowledge Gap

needs to be done in order to reach them, etc. The gap between reality and current perception got progressively smaller.

To illustrate the point, in the first phase, a kind of "spray and pray" behavior was observed, with participants shooting all over the place to try to find clues, or valuable pieces of information. This pattern was one of many displaying the lack of knowledge attackers had of their environment. The noise they created gave the first clear indication that these were not regular users. The activities that were observed included everything from scanning to exploits. During the initial phase, participants tapped multiple traps, decoys and beacons. In one particular case, a participant downloaded every file in the file system. This behavior pattern was dramatically reduced in subsequent stages.

The Significance of Attacker Intel

For the purpose of the research, participants were given a set of instructions and hints to simulate an attacker's pre-attack intel. It was found that a large number of them paid scant attention to the instructions, which contained a strong hint regarding where to find the first of the 5 hashed files. When analyzing the results, distinct differences were found in the performance of attackers with intel (i.e. those who had read the instructions) and those without. On average, participants who read the instructions and got the hint managed to solve the first phase of the challenge in less than 100 shell command attempts. This group of participants showed how good intel and prior attack knowledge can really affect the effectiveness of the attack.

The findings above are an important example of what the power of knowledge gives to attackers. The more information attackers have prior to entering the network, the quieter (and harder to detect) they can be. As attackers spend more time within the network, they learn more about it and this in turn will allow them to be more silent — and harder to detect. This is another indication of the critical value of early detection.

One way for defenders to deal with attacker intel is to increase the attacker's knowledge gap. This is where deception techniques can be particularly effective. Understanding where attackers go to gather their intel is a powerful tool for defenders as it allows them to strategically deploy their deception components.

Distribution of Traps Consumed

The participants' activity around the traps was tracked and every read, download and trip-up was logged. More than 1.9 million log lines were collected, cross referenced and analyzed to find out what trap

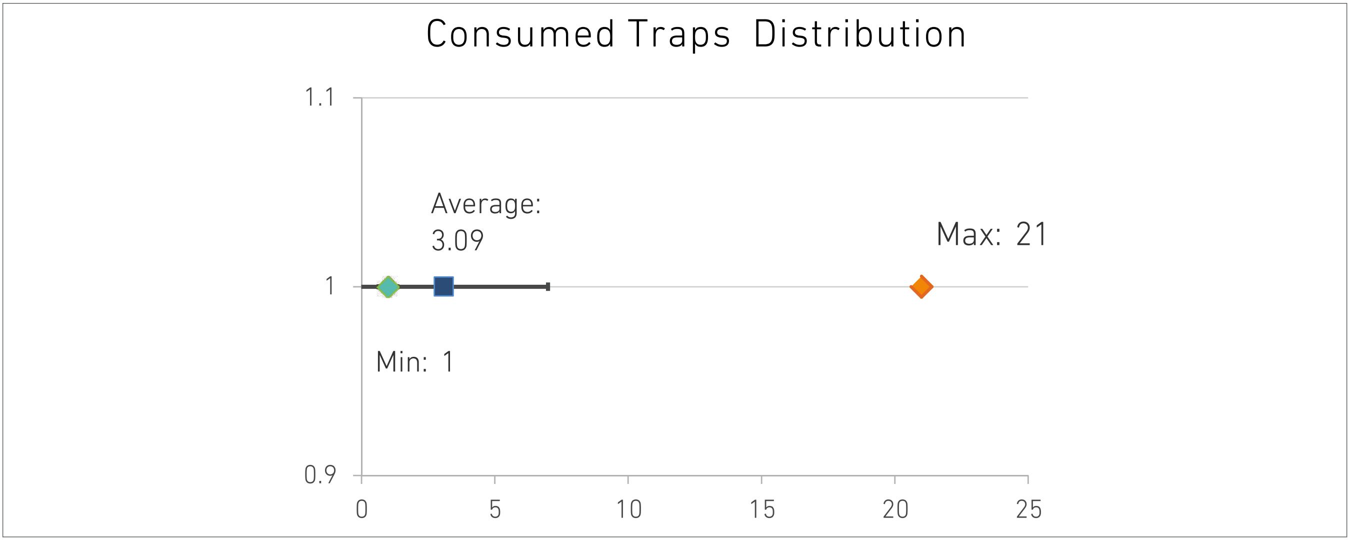


Figure 19: Traps consumption distribution

information was actually viewed. Participants' interest in traps proved to be very diverse and traps were consumed in a variety of ways.

Top Trap Statistics

- Logged 340 occurrences of a trap being consumed, i.e. the attacker took the bait
- On average each trap was consumed three times
- The most popular trap, the PUTTY application trap, was consumed 21 times
- Document trap consumption was logged 120 times
- 62 percent of all traps laid were discovered

Total accumulated touches per top trap type

- 119 document traps
- 98 application traps
- 68 email traps

Percentage of touches per top trap type

- 90% application traps
- 70% email traps
- 64% document traps

Trap consumption is depicted in figure 20.

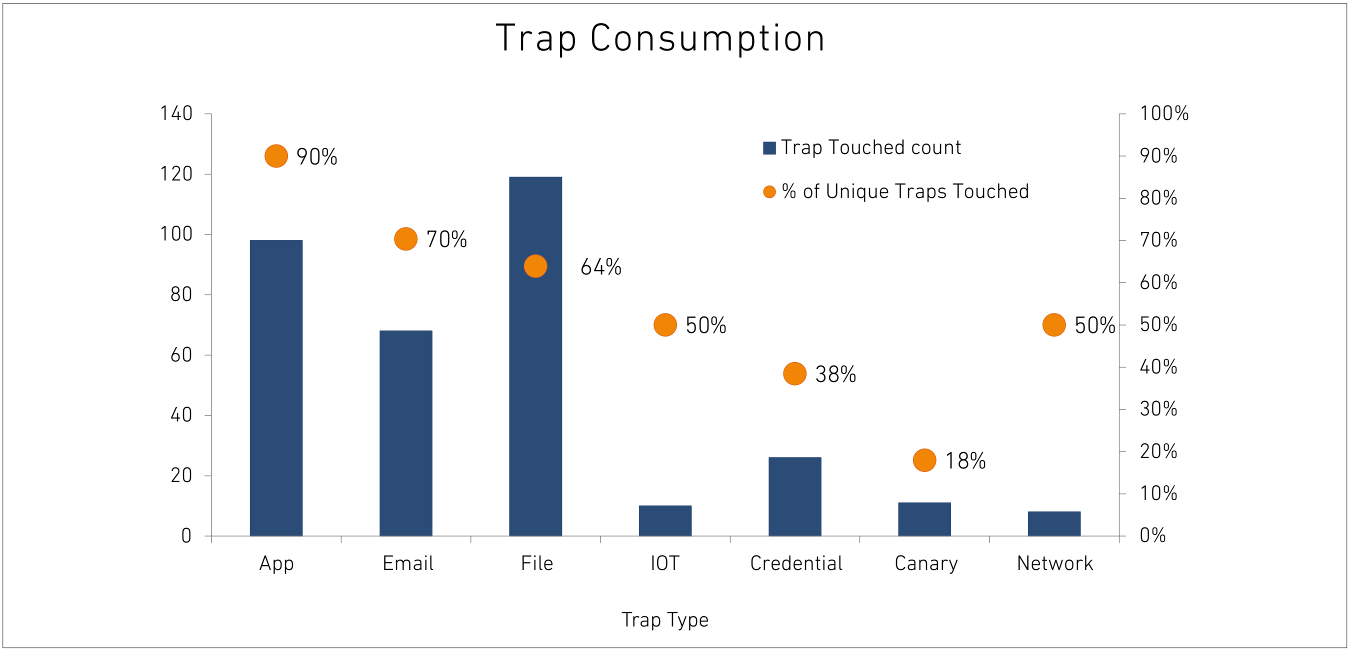


Figure 20: Trap consumption count

Man vs. Machine — Distribution of Attacker Trap Appetite

Access to traps was reviewed also from the perspective of human attacker vs automatic (malware) attacks. Analysis shows a difference in behavior patterns between human and machine-based attacks.

Malware, being mostly automatic, looks for highly structured data in applications and browsers. Some more sophisticated types of malware also look at document. Humans on the other hand, are much more diverse, but with a clear preference to documents (77 percent). Beacon traps were triggered equally in percentage by human and machine attackers at 25 percent.

No one trap type covered all attackers and it was seen that if enough trap types are placed, observing the attack patterns can indicate what kind of attacker is on the network (i.e. human or machine).

On average, human attackers triggered 10.5 traps during their attempt. In addition, human attackers discovered quite a few passwords, network locations and services which they later used in subsequent steps — as described further in this paper.

Top Traps Triggered by Human Attackers

- Document traps: 77% of attackers
- Credential traps: 45% of attackers
- Email traps: 36% of attackers

In addition, human attackers also reached network and application based traps.

Top Traps Triggered by Malware

- Application traps: 88%
- Beacon traps: 25%
- Document traps: 13%

None of the malware attacks touched email or network-based traps.

The traps clearly succeeded in getting the participants' attention and also in deceiving them with the fake information they contained. Whether specific or generic, they achieved their primary purpose of making the attacker think there is an asset that is interesting enough to visit.

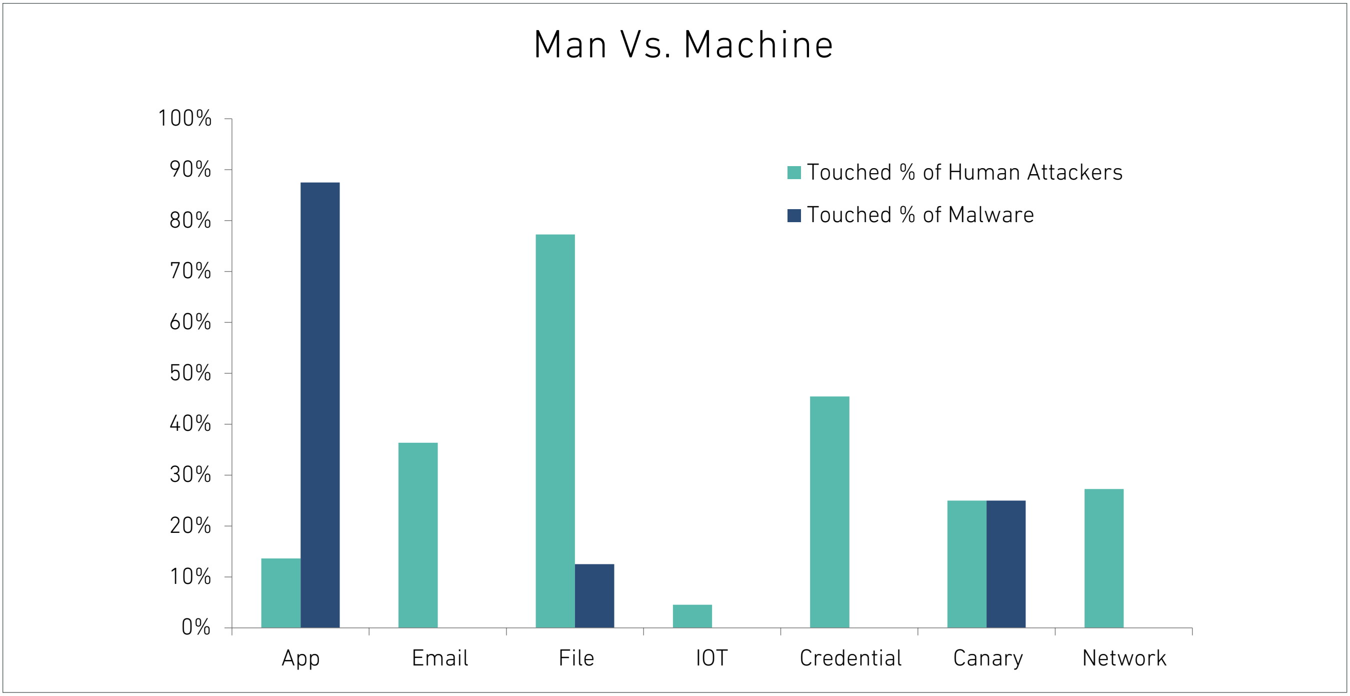


Figure 21: Type of trap consumed: human attackers vs. malware

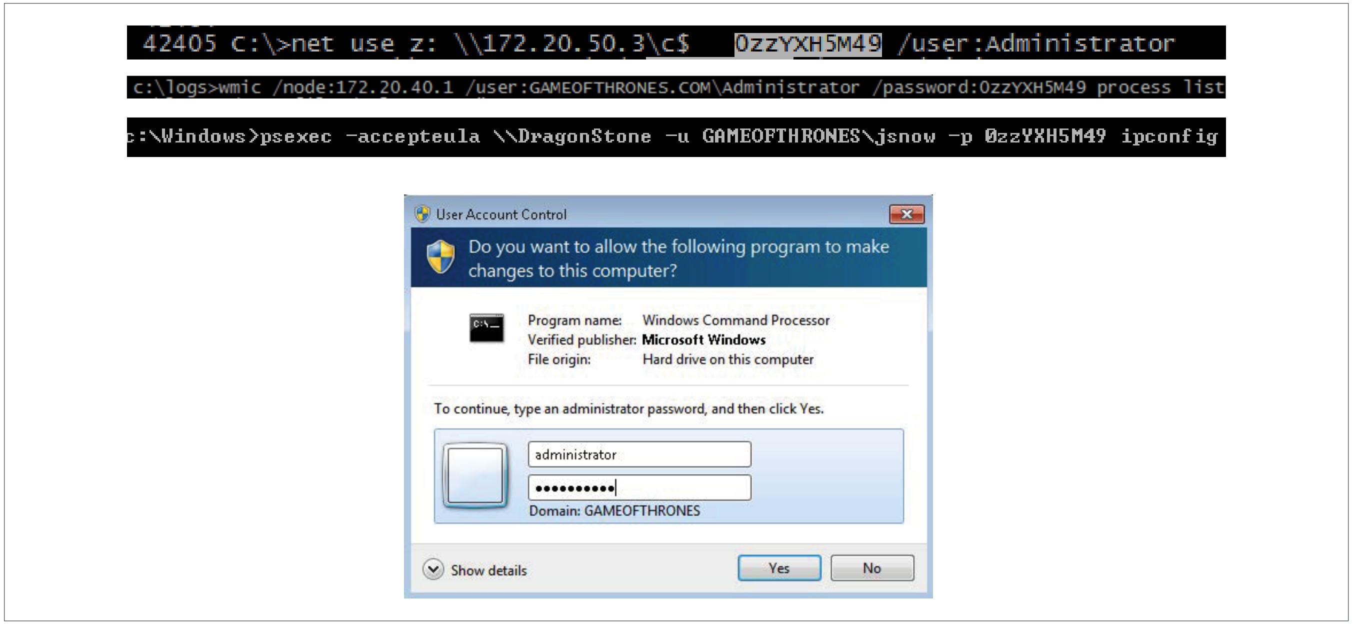
Passwords and Credentials

Passwords are the Holy Grail as far as attackers are concerned. Our research showed that attackers not only picked up passwords regardless of their source — email, credman, lsass — or format — cleartext, hashed, session ticket, etc. — they also used passwords multiple times in a variety of locations. For example:

- Attackers found an average of two credentials each
- Every password that was found was used 2.5 times on average
- Maximum times of password reuse: 11 times in 11 different places

Passwords and credentials make very efficient traps. Users tend to use the same credentials on different services and attackers are well aware of this. Thus placing credentials in the vicinity of a resource (asset, application) was a big enough hint for attackers and made the trap almost irresistible.

A typical attack pattern used by attackers in the research involved finding a password. This became more frequent in the 2nd and 3rd phases. After attackers gained admin access to one of the machines, they attempted to move laterally in the network. The figure below illustrates several such attempts.



```
42405 c:\>net use z: \\172.20.50.3\c$   0zzYXH5M49 /user:Administrator
```

```
c:\logs>wmic /node:172.20.40.1 /user:GAMEOFTHRONES.COM\Administrator /password:0zzYXH5M49 process list
```

```
c:\Windows>psexec -accepteula \\DragonStone -u GAMEOFTHRONES\jsnow -p 0zzYXH5M49 ipconfig
```

User Account Control

Do you want to allow the following program to make changes to this computer?

Program name: Windows Command Processor
Verified publisher: **Microsoft Windows**
File origin: Hard drive on this computer

To continue, type an administrator password, and then click Yes.

administrator
••••••••••
Domain: GAMEOFTHRONES

Figure 22: A typical use of passwords

ARP "Poisoning"

The ARP table was used by a number of participants who subsequently fell for the traps that had been planted. Interaction with traps built into ARP increased the likelihood of stepping into a decoy by 27 percent, bringing it from 52 percent to 66 percent.

It is important to remember that one of the key components of success when using deception is diversity (i.e. the use of many types of traps and decoys each tailoring to different "tastes" of attackers). Hence while ARP traps were not consumed by all participants, they still contributed a fair share of "catches."

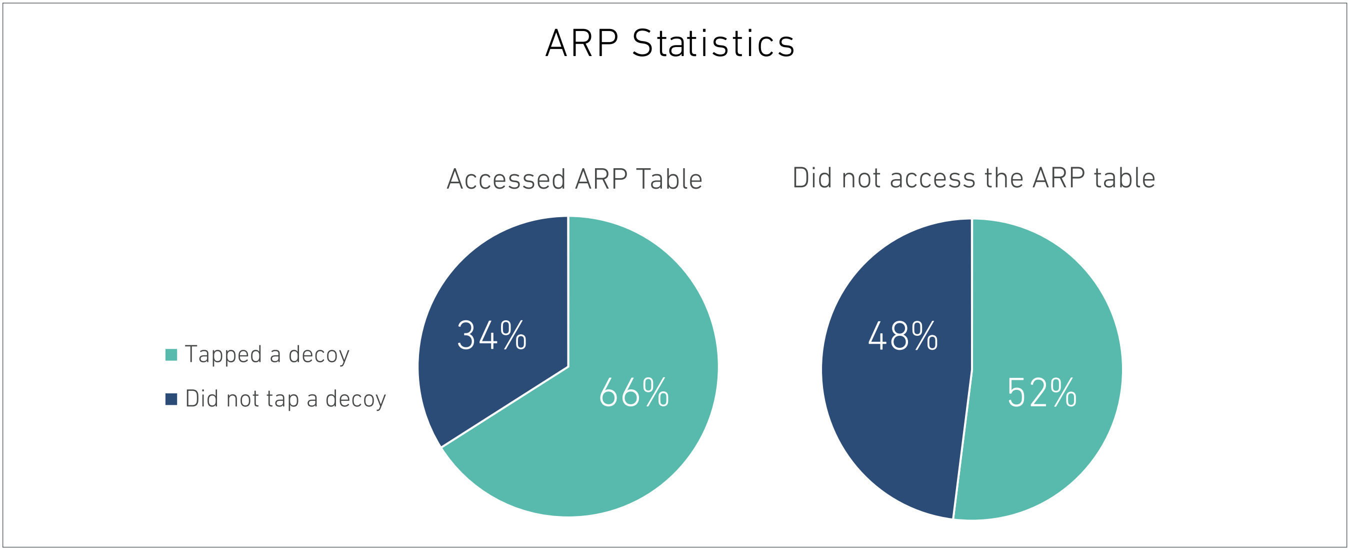


Figure 23: ARP poisoning increases the probability of tapping a decoy

Decoy Access

Probably the best way to measure the credibility and effectiveness of the decoy network is by measuring the level of interaction the attackers have with the system. Interaction with the decoy has two advantages for the defenders. One is the actual detection of the attacker, exposing his activity, tools, methods and even the attack vector. The second, and no less important, is the fact that occupying the attacker, causing him or her to spend time on a decoy system, serves to slow down the attack, giving the defenders time to start the remediation process while the attacker is still engaged with non-critical assets (i.e. decoys). The research revealed a very high interaction level, including:

1. On Average, each participant interacted with 9.7 different decoy services
2. 20 percent of participants accessed closer to 60 different decoy services

Trend Analysis — sophisticated vs unsophisticated attackers

There are some fundamental differences between "attacks" and "sophisticated attacks." This was made evident not only in pre-attack intelligence gathering as discussed in the sections above, but also in the level of interaction with decoy services.

The findings show that while participants tapped on average nearly 10 decoy services (see figure 25 below), there was a difference in the characteristics of the interaction. For example, participants who accessed a high number of services used mostly low interactive methods of access such as pings and SYNs. This points to the use of network scanners to find "live" hosts and services by observing their response to various requests. On a decoy, this type of behavior creates a large number of alerts, which quickly indicate the presence of an attacker.

Top Services Accessed[5] (total)

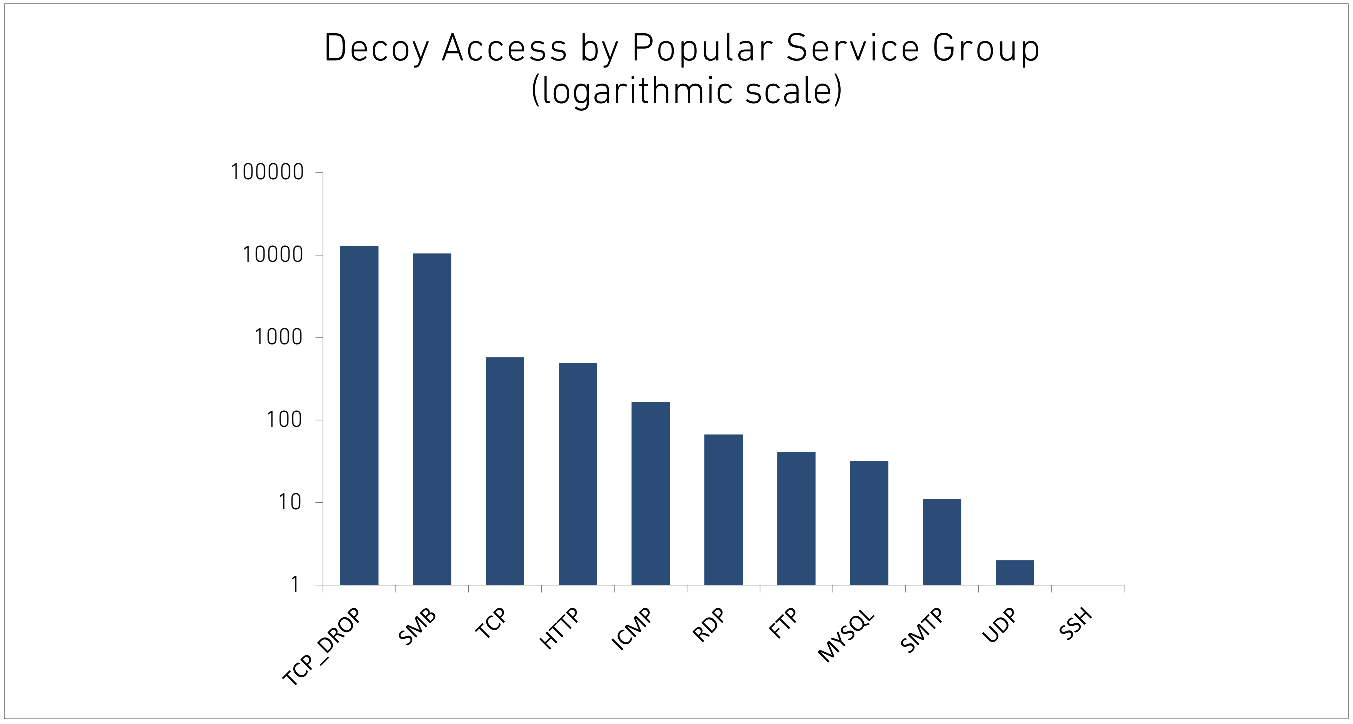


Figure 24: Decoy access by service group

[5] There was one application per decoy

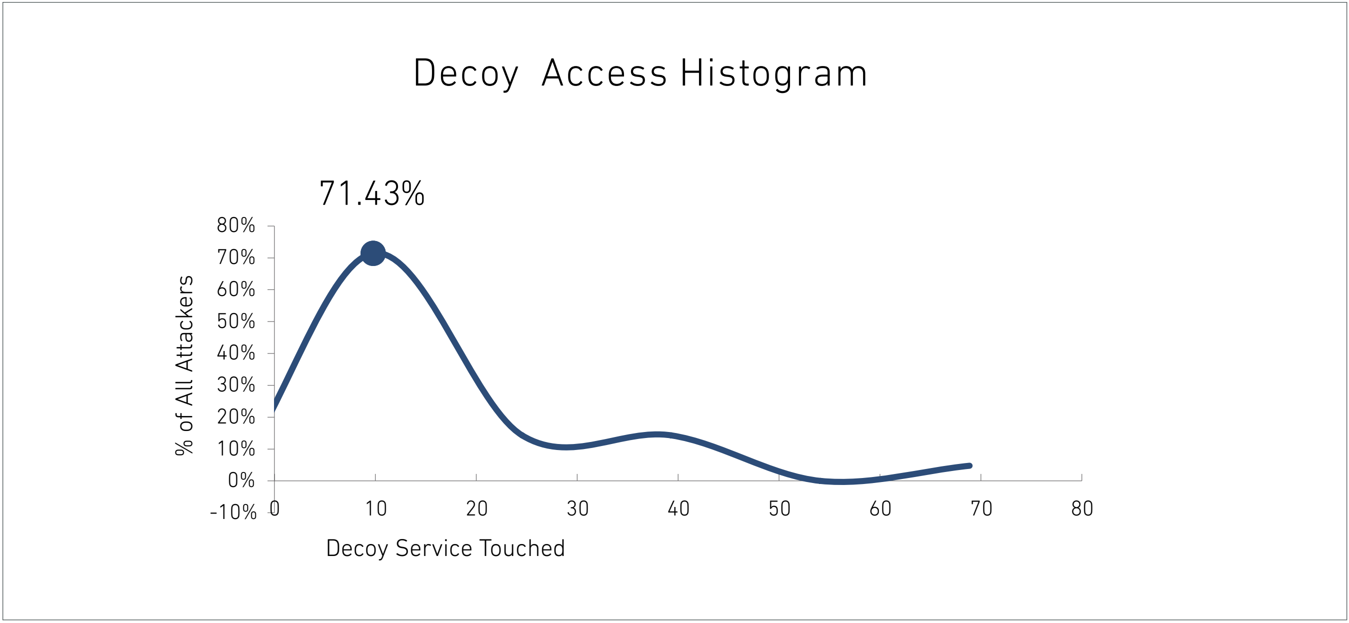


Figure 25: Decoy services access

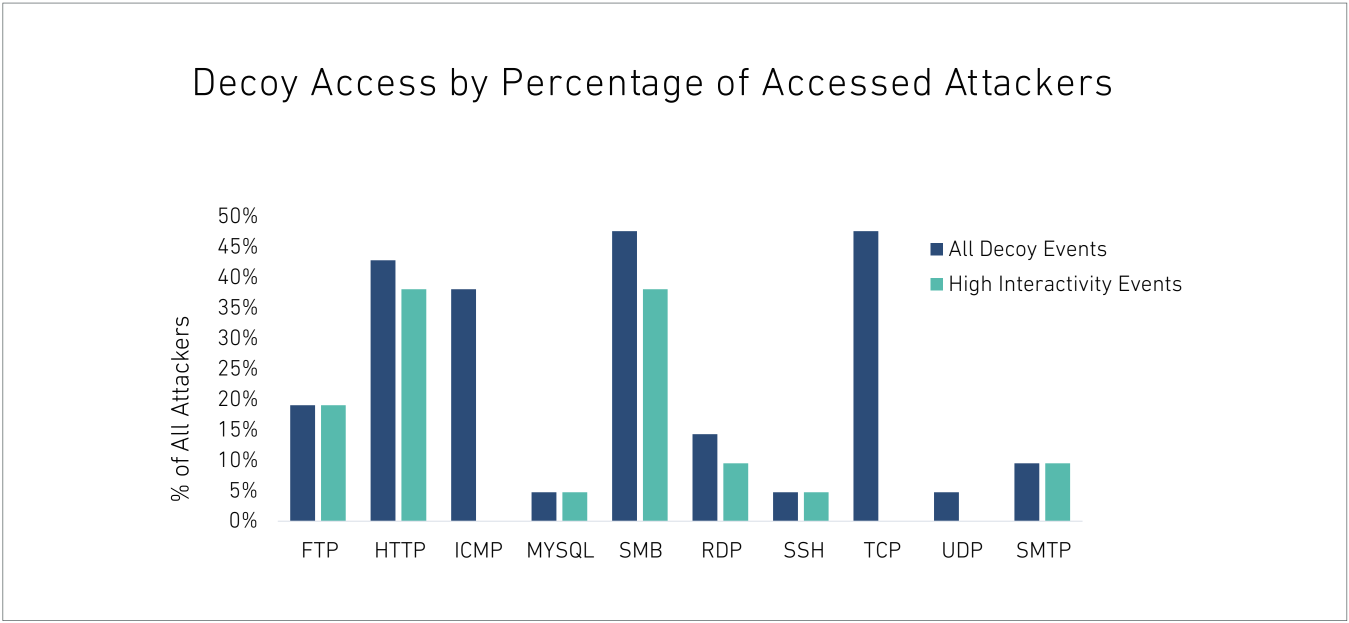


Figure 26: Decoy access and decoy interaction events

Decoy Authenticity Validation

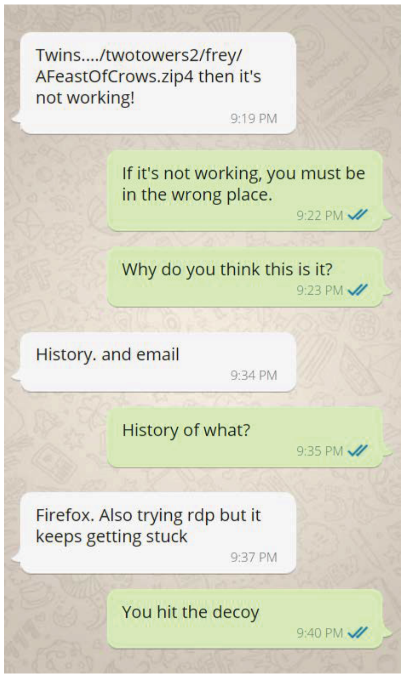
Another, even more interesting finding was the behavior of sophisticated attackers (i.e. those who used high interaction methods such as directory browsing and file fetching on FTP, running SSH commands, etc). These participants focused their efforts on a relatively small number of decoy services (four services per user on average) but spent a lot of time interacting with each of them. The conclusion here is simple: The participants, or attackers, had a hard time differentiating the decoys from the real assets.

One example of how credible decoys can be seen in figure 27 below. It is a screen shot from a chat our researcher had with one of the participants as the latter was attempting to extract information from what he thought was a real asset. After spending several hours on the task, our researcher finally revealed the fact the interaction was in fact with a decoy.

Decoy authenticity is most important. If an attacker can spot a decoy, he will avoid it. The fact that attackers used sophisticated methods and interacted with the decoys over a long period of time (in some cases several hours) proved that the deception layer was credible and therefore effective.

Decoy Access — Diversity Is a Must

As mentioned in the opening remarks, 66 percent of the attackers were initially detected by decoys. In most of these cases, once an attacker encountered a decoy, they interacted with it in a significant way, as depicted in figure 26. What is also evident is that no single decoy was tapped by more than 47 percent of attackers. This data point underlines the importance of decoy diversity, which serves to increase the overall decoy access, and hence significantly improves decoy-based detection rates.



Twins..../twotowers2/frey/AFeastOfCrows.zip4 then it's not working!
9:19 PM

If it's not working, you must be in the wrong place.
9:22 PM

Why do you think this is it?
9:23 PM

History. and email
9:34 PM

History of what?
9:35 PM

Firefox. Also trying rdp but it keeps getting stuck
9:37 PM

You hit the decoy
9:40 PM

Figure 27: A conversation between a researcher and one of the research participants

Summary of Findings

Deception Works

The primary goal of deception is to expose attacks as they happen, and in this respect the research clearly proved that deception works. All of the attackers triggered multiple traps and were detected by one or more detection mechanisms that had been put in place. Moreover, as has been shown in the previous section, while less sophisticated attackers were detected when attempting various scanning activities, more sophisticated attackers were tempted to interact with the decoys. By leading the attacker to spend (his or her) valuable time and efforts on fake assets, the deception layer was successful in both slowing the attack and diverting it from real assets.

Diversification Is Key

As shown in figure 28, deception has to be diverse in order to be effective. Breadcrumbs using different types of data spread around endpoints and across the network lead to decoys, data traps and beacon traps making deception deterministic and more effective. In addition, by augmenting the deception layer with network and traffic analysis, security analysts reach an even higher accuracy of detection.

Diversity is important for other reasons as well. Attackers will not always use the traps in the way the defenders intended. Some attackers take information they discover and use it in creative ways, for example, taking a password meant for one asset or service, and

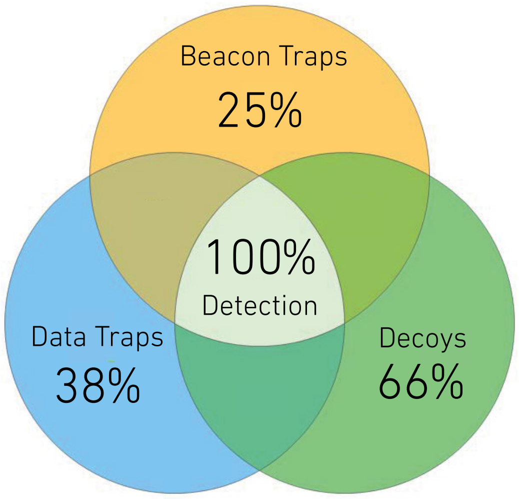


Figure 28: Attacker detection by deception mechanisms

attempting to use it in order to access another. When the credential information comes from a trap, it is of course false and so the attacker unknowingly creates his own traps. Hence, by providing a diversified deception layer, the attackers unknowingly participate in the process of broadening and thickening the organizational defense.

Expand the Attacker's Knowledge Gap

Sophisticated attackers use information that they gather either prior to the attack or during its initial stages in order to familiarize themselves with the network. This information helps them to focus the attack on lucrative assets, as well as to better disguise themselves and hide their activity until they successfully complete their mission.

So while minimizing the knowledge gap and becoming quieter and harder to detect are key goals of attackers, the defenders' goals are exactly the opposite. By planting breadcrumbs luring attackers to decoys, plus augmenting the deception layer in real-time, defenders can expand the knowledge gap exposing attackers without risk to real data, systems or operations.

Put Deception in the Right Context

The better the deception blends into the specific organization's environment, the less it arouses the attackers' suspicion. Hence, the more tailored traps and decoys are to the organization the more convincing they will be.

It has been shown that attackers may use low-interaction methods, i.e. scanning, to probe around the network. This type of activity is "noisy" — almost careless — and can be detected by even the simplest decoy or other network detection methods. However, sophisticated attackers use more subtle methods and will easily detect a trap if the decoy system does not behave like a real asset. The research showed that once the decoy is believable, even the most sophisticated attackers will spend much time interacting with it without realizing that they are in fact being watched, that their moves are being monitored, and their actions recorded.

Conclusions

The research presented a real-life scenario in which an organization finds itself subject to an advanced persistent threat or falling prey to a zero-day attack. While traditional outbound honeypots can be used to prevent attacks, the research focused on post-breach detection rather than attempting to prevent the infection (i.e. infiltration). In conjunction, deception technology slows the attackers — by deceiving them into believing they're interacting with real endpoints, servers, data repositories, etc. — and prevents further spread of the attack by deflecting it from real assets to decoy assets.

Deception-based security is being adopted by leading organizations as an effective and highly accurate method for detecting both human and malware attacks that have penetrated perimeter defenses. The research results proved in a controlled environment what is also seen in real field deployments — that even the most experienced and knowledgeable attackers are deceived by breadcrumbs and decoys and will be detected at an early stage of the attack. As has been shown, all attacks were detected long before any data could be exfiltrated. It has also been shown that deception serves to increase the attacker's knowledge gap, rendering much of the attacker's pre-attack intel irrelevant and providing defenders with a proactive, offensive tool with which to deflect the attack from real organizational assets.

The study showed that diversity is key for successful deception and that relying on a small number of decoy assets cannot provide sufficient coverage — even with a wide net of breadcrumbs. This was proven true in both human attacks and automated machine-based attacks.

Appendix A

Glossary

Assets	Resources in the network such as workstations, servers, laptops, routers, switches, mobile devices
Beacon traps	A mechanism built into a file (document or email) which sends a signal to a predefined server every time the file is opened
Breadcrumb	A fake resource or data resembling an item of information that points to or facilitates access to another location or an asset on the network or even outside of the network. Examples: email message containing URL and login information, registry entries containing pointers to data folders, or Active Directory data. Like the components they resemble, breadcrumbs point to assets, the difference being that they point to decoys which are fake assets. Also termed trap or mini-trap.
Canary	See beacon traps
CnC	Command and Control - Generally a human controlling malicious activity in the network from outside, for purpose of theft, espionage or other forms of harm to the target network or its owners
Decoy	A fake asset resembling a workstation, server, service, laptop, router, switch, mobile device or other computer component. Because decoys appear to be legitimate assets, attackers probe around the decoy looking for valuable information and this is activity that can be easily identified by the defender. Decoys are sometimes referred to as Next-Generation Honeypots.
Detection	The act of identifying an attacker or malware program that has infected an asset or resource.
Honeypot	A computer security mechanism set to detect, deflect, or, in some manner, counteract attempts at unauthorized use of information systems.
Infection	The act of an attacker gaining a foothold on a computer in the network. Once one asset is infected, it provides a platform for the attacker to infect another
Knowledge gap	The difference between the true picture of the network and the picture currently perceived by the attacker
Malware	Generally used to refer to a malicious program that works independently i.e. is not controlled by a human CnC
Noise	The amount of network activity being carried out by an attacker in his attempt to move to additional locations in the network
Perimeter security	Security mechanism deployed at the boundary between the private and locally managed-and-owned side of a network and the public and usually provider-managed side of a network. Perimeter security tools include firewalls, routers, web filters, etc.
Post-breach detection	Identifying an attacker who has penetrated the network
Prevention	Prevention is all about keeping attackers and malicious users out of the organizational network. Also see Perimeter Security.
Trap	Another term for breadcrumb (see above).

Fidelis™
Cybersecurity

Fidelis is the industry's only completely integrated, automated network and endpoint detection and response platform. Fidelis improves the efficiency and effectiveness of security operations teams by condensing alert data into actionable threat summaries and then automating response and investigation actions instead of piling more alert data on already fatigued security staff. With automatic validation, investigation and prevention of attacks, Fidelis is engineered for visibility, designed for response and trusted by the most important brands in the world. See what you've been missing.