# Deceiving the Attacker: Some Fake Treasures for Your Network

The need of attackers to gain access and exfiltrate data is critical to the sustainment of criminal enterprises so identifying when a criminal is looking around your network for valuable information can be an invaluable tool to effectively shut them down when correctly implemented. As attackers and environments have considerable variability the suggestions in this post should be seen as a place to start a discussion and not a comprehensive deception and alert solution. The four types of deceitful assets in this article are systems, service, accounts, and files. As each has its own value rating determined by the type of attacker specific scenarios which have been chosen are intended to render the greatest value with the shortest amount of reading.

The problem with honeypots is when an attacker accesses one that's it, there is no place to go. When an organization develops a deceptive system there is a front-end as well as a supporting database. The front-end can be publicly facing or internal but must not be "advertised". For public front-ends that means the interface is hidden using a robots.txt file [1] (which should never be used for legitimate organizational assets) or bu using HTML tags [2]. This front-end should be connected to a fake database. This database should be fairly large and contain pseudo-data that mimics real data. As an example, the database could have usernames, hashed passwords, addresses, and hashed payment data. As with all the solutions in this article, this system needs to be highly audited with all alerts being considered critical. The alerts can be through logging from within the systems themselves, based on network flow rules, or through IDS and Fire Wall based alerts on the SIEM. When building the system it is important to remember the system must be realistic to the organization. This means healthcare organizations should not have fake e-commerce systems and banks should not have fake HIPAA information in their database, but also includes patch compliance and services. The fake system's patch level should be within one patch cycle of the hosting organization and should not use services that are foreign to other systems used by the organization.

SSH and DNS are two services that seems to invite both automated and manual intrusion attempts. It is this reason that near-current recommendations are to change the default port numbers whenever possible. This can be as range from changing the default port number for RDP on windows [3] and OpenSSH on RHEL [4] to more complex network service based ports such as DNS, NTP, and DHCP. When this latest practice is used to prevent the success of automated attacks it provides organizations with an opportunity to create fake service ports. The idea is to mimic the fake system scenario listed above by using portproxy [5] for Windows or port forwarding for RHEL [6]. The idea is to have automated and manually attackers target a well-known port then have all of their traffic redirected to another system that has been planted with appetizing (fake) data. These ports and the system they redirect traffic towards should never be used by internal personnel and any alerts or activity associated with them should be investigated. One caveat to be aware of is the use of network scanners and

mappers, both can hit these ports if they are not configured to act in a different way; however, in the case of scanners, taking the number of hosts from the scanner results and performing a "sanity check" against the alerts in the fake system can help identify any rouge devices. There should also be fake accounts on the system as any real accounts could be pirated if an attacker does successfully access that system.

Fake accounts are a bit more specialized in the world of deceptive tactics. This is because the best fake accounts have real permissions which in the case of user accounts means they require orchestrated response activity. If one of these accounts is used there needs to be an orchestrated response that disables the account and terminates all sessions. The idea is to create non-privileged (non-admin for the purposes of this article) and privileged (admin) accounts that have appetizing names and permissions. For example, "VMadmin1" might be an appetizing account name for an attacker. These accounts should never be used by internal personnel and their very existence should be highly confidential so any would-be internal threats are not aware of their existence or purpose. Again, these accounts do need to be part of a privileged group in the domain so they are not transparently bogus. A different type of fake account that can also be used is the fake VIP account. In this scenario the organization creates email accounts for every member of corporate leadership that is not advertised or known. Every corporate leader that is named on the public website should have a parallel fake account. These accounts should be technically prohibited from sending email to any address other than other fake email accounts but should be permitted to receive email. They should also have a number of folders that have a number of "fake" emails from other fake email accounts. Any emails sent to these accounts can be inspected for phishing and any access of these accounts should create a high criticality alert that is investigated. Fake email accounts may require a higher level of security maturity than most of the other proposed solutions, such as fake files.

No organization can exist without creating valuable data and no valuable data can be reasonably leveraged without creating some form of sorting and structure mechanism. For most organizations that system is a file system; therefore, it is only reasonable to believe attackers will look for files that contain valuable information that can be used to further attack or which can be stolen. Some files are based on older security practices like robots.txt [1] or slightly newer files such as install files, or complete directories. Robots.txt can be filled with fake information as described in the "fake systems" section. Organizations that use Windows for server and end-use operating systems it is common to use the unattend.xml file to increase efficiency and permit windows to be installed and configured without an admin overseeing the process at time of install. Within this file is an administrator password which makes it a wonderful place for an attacker to pick-up valuable credentials when the file was not deleted [7]. This also provides an opportunity for an organization to place a fake unattend.xml file on various devices. These fake files should contain fake admin account credentials that are unique to that specific file. This will permit the quick identification of which device originated the compromised credentials. The actual file access logs should be audited [8] and sent to the central logging service. This will permit the identification of any file that has experienced interaction but where the offender did not take that admin credentials. For RHEL a kickstart file is used which also contains highly valuable (root)

privileges on the "rootpw" line [9]. This presents a very similar opportunity to create a fake file with a fake rootpw. This fake rootpw will not work because there is only one root account on a Linux system so auditing file access attempts is important. In Linux the auditctl command can be used to create auditing rules for this file [10], but again, these logs should be exported to a central log repository for alerting and response activities. For the "fake system" listed in the first section or for any system where feasible (meaning legitimate systems users will not erroneously interact) fake directories can be created and populated with fake files. The fake directories should have appetizing names that are relevant to the organization or host system (ex. initiatives, research, testing, roll-out, deployment, customer, finance, scan results, POAMs, incidents, investigations, pentest, etc.). These directories should also be highly monitored with any access or interaction triggering an alert that starts an investigation. In the case where "normal" users are causing too many "false positives" a directory may need to be moved to a location which is of greater strategic advantage. From fake systems to fake directories, there are many fake treasures that can be placed on a network and far more exist than have been discussed in this article.

The need of attackers to gain access and exfiltrate data is critical to the sustainment of criminal enterprises so identifying when a criminal is looking around your network for valuable information can be an invaluable tool to effectively shut them down when correctly implemented. As attackers and environments have considerable variability the suggestions in this post should be seen as a place to start a discussion and not a comprehensive deception and alert solution. The four types of deceitful assets in this article were systems, service, accounts, and files. As each has its own value rating determined by the type of attacker the specific scenarios which have been chosen were intended to render the greatest value with the shortest amount of reading.

References

[1] https://www.seobility.net/en/wiki/Robots.txt

[2] https://www.techrepublic.com/article/guide-to-using-noindex-nofollow-and-disallow/

[3] https://docs.microsoft.com/en-us/windows-server/remote/remote-desktop-services/clients/change-listening-port

[4] https://sharadchhetri.com/centos-7-rhel-7-change-openssh-port-number-selinux-enabled/

[5] https://docs.microsoft.com/en-us/windows-server/networking/technologies/netsh/netsh-interface-portproxy

[6] https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/security_guide/sec-port_forwarding

[7] https://docs.microsoft.com/en-us/windows-hardware/manufacture/desktop/update-windows-settings-and-scripts-create-your-own-answer-file-sxs

[8] https://docs.microsoft.com/en-us/windows/security/threat-protection/auditing/apply-a-basic-audit-policy-on-a-file-or-folder

[9] https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/installation_guide/sect-kickstart-syntax#sect-kickstart-commands

[10] https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/security_guide/sec-Defining_Audit_Rules_and_Controls#sec-Defining_Audit_Rules_with_auditctl

Thank you for reading

-Andrew Kosakowski

www.Anthko.com