I. J. Computer Network and Information Security, **2017, 7, 1-10**
Published Online July 2017 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijcnis.2017.07.01

Malware-Free Intrusions: Exploitation of Built-in Pre-Authentication Services for APT Attack Vectors

Aaron Zimba
University of Science and Technology Beijing/Department of Computer Science and Technology, Beijing, 100083, China
E-mail: azimba@xs.ustb.edu.cn

Zhaoshun Wang
University of Science and Technology Beijing/Computer Science and Technology, Beijing, 100083, China
E-mail: zhswang@sohu.com

***Abstract*—**Advanced Persistent Threat (APT) actors seek to maintain an undetected presence over a considerable duration and therefore use a myriad of techniques to achieve this requirement. This stealthy presence might be sought on the targeted victim or one of the victims used as pawns for further attacks. However, most of the techniques involve some malicious software leveraging the vulnerability induced by an exploit or leveraging the ignorance of the benign user. But then, malware generates a substantial amount of noise in form of suspicious network traffic or unusual system calls which usually do not go undetected by intrusion detection systems. Therefore, an attack vector that generates as little noise as possible or none at all is especially attractive to ATP threat actors as this perfectly suits the objective thereof. Malware-free intrusions present such attack vectors and indeed are difficult to detect because they mimic the behavior of normal applications and add no extra code for signature detection or anomaly behavior. This paper explores malware-free intrusions via backdoors created by leveraging the available at pre-authentication system tools availed to the common user. We explore two attack vectors used to implant the backdoor and demonstrate how such is accessible over the network via remote access while providing the highest level of system access. We further look at prevention, detection and mitigation measures which can be implemented in the case of compromise.

***Index Terms*—**Remote Access, Authentication, Backdoor, APT, Accessibility Tools, RDP.

I. Introduction

Recent advancements of Internet technologies have likewise seen the emergency of various forms of cyber-attacks [1]. These attacks span from sophisticated undertakings such as Advanced Persistent Threats (APT) [2] to those that simply leverage the ignorance of benign users. Each new form of attack introduces a new attack vector which in turn contribute to the expansion of the attack surface of the respective attack domain. Thus, today's cyber-attacks don't merely depict a single or isolated attack but rather a myriad of atomic attacks which actualize a given security breach as witnessed in APTs [3]. Attackers, therefore, seek to employ different attack vectors in order to reach their target and launch a successful attack. The aforementioned attack vectors are as a result of exploitation of vulnerabilities of a system or the flaw in design or implementation of security mechanisms.

Cyber-attacks can generally be classified as targeted or untargeted attacks. Regardless of whether a given attack is of the former or latter class, it can be deemed as either outsider or insider attack [4]. Therefore, depending onto which category an attack falls, it will utilize different facets of attack vectors to accomplish the overall objective. APTs, for example, not only seek to intrude a system to breach confidentiality or integrity but also to maintain a persistent presence for advancement of future attacks. APTs therefore seek to remain undetected for long periods of varying durations [5]. To achieve this stealthy intrusion and undetected presence, attackers use different techniques most of which include malware. But malware is very noisy even in the absence of an Intrusion Detection System (IDS) because unknown files issuing critical system calls, changing registry entries and initiating network connections always raise a red flag. Thus the ideal stealthy intrusion an attacker would want to implement is one which does not indulge malware but utilizes lapses in the system. This is ideal because normal system binaries even if used covertly pose a big challenge to detect the malicious activity. As later elaborated in this paper, poor configurations and oversight of such system files can lead to backdoors which an attacker can leverage persistently, a condition so desirable for APTs. One of the most famous backdoors resulting from oversight and poor configuration is the use of Windows accessibility tools for password recovery or access to a password-locked system. The typical work around involves replacement of accessibility suite binaries with an executable such as cmd.exe which when invoked with the appropriate keystrokes presents an interactive system level command prompt at pre-login thereby bypassing the Operating System's authentication requirement. If the replacement remains unriveted, the system remains vulnerable and accessible on wild internet to any threat actor so long remote access remains active. This is evidenced by probing for remote access configured in this manner using the SHODAN engine [6]. It should also be noted that a malicious user, such as a disgruntled system administrator, could also leave such a backdoor for future access when he's no longer with the organization of the moment. Other threat actors in this respect would include a rogue system administrator or technical personnel who intentionally sets up such backdoor access and routinely uses it to access the system offsite via remote access. There are different ways of implementing this backdoor and we use the least labor intensive which involves manipulation of the system's registry entries.

This paper tackles the exploitation of system files for creation of a backdoor which is accessible over the network leveraging the default Remote Desktop Protocol (RDP) that ships with the Operating System. We carry out tests on five different versions of the Windows Operating System and we prefer registry alterations over binary executable replacement for backdoor establishment due to reliability and obscurity of the former.

II. Prerequisites and Related Concepts

As mentioned in the preceding section, a successful attack usually has specific preconditions for its actualization. Likewise, the materialization of the attack in contention requires that some conditions be met. In this case, accessibility tools should be available (which they are by default in a Windows installation) and remote access which uses RDP running on port 3389 by default configuration settings [7].

A. The Windows Accessibility Tools

Windows Accessibility Tools are designed to aid computer users with an impairment or condition. Therefore these tools are available to the user before authentication of a session so that even the aforementioned users are provided with the resources and ability to login into the system. It is these tools which are leveraged for creation of the backdoor since they run at system level even before authentication. The user typically invokes a combination of keystrokes to access these functionalities. Table 1 below lists some common accessibility tools available at Windows pre-authentication along with their invocation keystrokes and the corresponding executable system binaries.

Considering the information in Table 1, there are two attack vectors which can be pursued to plant a backdoor locally; either by switching an executable binary with cmd.exe or setting cmd.exe as the debugger for a specified accessibility suite executable binary. Network access to this backdoor is achieved by activation of remote access via RDP.

The Windows Operating System, however, enforces security for system files as those depicted in Table 1. Thus, going with the first attack vector of replacing any of the accessibility suite binaries with another file induces a rigorous system check. Windows first checks whether the file replacing the binary is digitally signed by Microsoft. It further checks whether the file resides in the system directory %systemroot%\system32 and in so doing advertently implements Integrity Level validation for secure objects and administrative permissions as well.

Table 1. Windows Accessibility Tools

Accessibility Tool	Invocation Keystrokes	Corresponding Executable Path
Display Switcher	Windows Key + P	%systemroot%\System32\DisplaySwitch.exe
Contrast Manager	Alt + Shift + PrntScr	%systemroot%\System32\EaseOfAccessDialog.exe
Application Switcher	Alt + Tab	%systemroot%\System32\AtBroker.exe
Utility Manager	Windows Key + U	%systemroot%\System32\Utilman.exe
Sticky Keys	Shift Key five times	%systemroot%\System32\sethc.exe
Magnifier	Windows Key + " = "	%systemroot%\System32\Magnify.exe
Narrator	Windows Key + "Return"	%systemroot%\System32\Narrator.exe

Windows runs other security checks all which cmd.exe satisfies, in addition to the aforementioned. Nevertheless, Windows does not check the integrity of the running binary, e.g. whether the hash of the running executable collides with any of that in %systemroot%\system32, which in itself is an indicator of compromise. Furthermore, Windows does not check the context of execution, i.e. why an interactive command prompt is running with system level permission at pre-authentication. In this covert action, the Operating System thinks that it's running a legitimate executable from the accessibility suite when in the actual fact it's an application providing interactive system-wide access. It should be noted however that replacing these executable binaries requires administrative level access which could initially be achieved by social engineering or with malicious code whose sole purpose is privilege escalation, executable binary replacement and self-deletion to maintain the non-detection requirement.

The second attack vector, which avoids running into file ownership permissions issues, can be utilized which involves setting registry entries. In this attack vector, a registry entry is created which points cmd.exe as the debugger for a specified accessibility suite executable binary. If we desire to exploit creation a backdoor using accessibility tool number 2 in Table 1, then we add the following entry to the registry:

REG ADD "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options\EaseOfAccessDialog.exe" /t REG_SZ /v

Debugger /d "C:\windows\system32\cmd.exe" /f

With this entry in place, an invocation of the keystrokes Alt + Shift + PrntScr at the pre-authentication login interface will present an interactive command prompt running in system level. It's assumed in the above registry entry that the operating system is installed in the logical disk C. Otherwise the path should reflect the installation partition of the base system.

B. Remote Access via Default RDP

There are many ways of accessing a host remotely and Windows ships with remote desktop application which uses RDP. Accessibility tools are available over a remote desktop connection implying that the backdoor can be initiated by invoking the appropriate keystrokes in Table 1. Again, remote desktop access is turned off by default but we argue still that the initial malicious script could in addition turn on remote desktop access in the event that it's not switched on. RDP [8] is a Microsoft proprietary protocol which provides a graphical user interface for remote access connections over the network. It is one of the most scanned service on the Internet owing to its security importance [9]. The RDP daemon, by default, listens on port 3389 for both TCP and UDP requests. The sysdm.cpl commands enables the activation of remote access through which sessions can be established using the mstsc.exe process on the client's side. With the backdoor created and remote desktop activated, access to the victim host on the network is achieved by invoking the corresponding accessibility keystrokes.

III. Experiment Setup and Attack Model

The deployed infrastructural test-bed comprises two networks separated by a simulated Internet network. The first network has a Kali Linux host which the attacker uses to launch his series of attacks onto the second network which comprises five targeted Windows hosts as victims, all capable of remote access through RDP. We test the attack on Windows XP, Windows Vista, Windows 7, Windows 8 and Windows 10 which are connected to the same LAN via a switch and a stub router to the outer network. We do not consider Windows Server version Operating Systems as they lie beyond the scope of the paper. Our experiment setup is shown in Fig. 1 below.

We prefer Kali Linux for adversarial usage due to its robust security tools and the availability of all dependencies needed to run the script used in the actual attack.

As depicted in the Fig. 1 above, the attacker resides in a different subnet but could as well be located on the same LAN. Later versions of Windows Operating System, from Windows Vista onwards, introduce a new security feature, Network Level Authentication (NLA) [10] which requires authentication from the entity seeking remote desktop connection. We test the two attack scenarios where NLA is activated and one where it's switched off. It must be noted, however, that the NLA security feature is optional and not mandatory and is off by default. Windows XP SP3 NLA activation requires registry configuration.

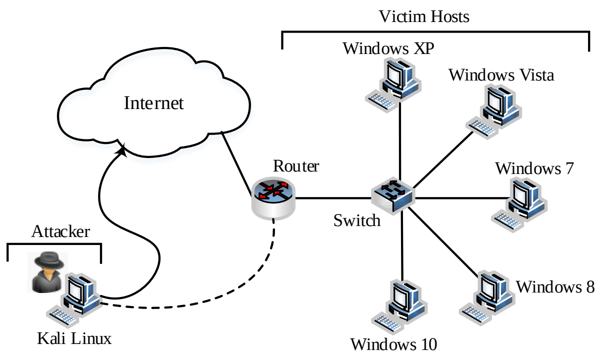


Fig.1. Experiment Setup of Remote Access Backdoor via RDP.

A. The Attack Model

We model the backdoor attack using conceptual units namely goals, assets, agents and actions as basic building blocks of the attack. It follows henceforth that the attack comprises an agent which can be a human actor or otherwise, who executes a series of required actions with the sole purpose of obtaining assets to reach the final goal.

1) Attacking Agents

Agents represent the threat actor carrying out attack. The actor can be software or human. The agent of our model is a human attacker who is usually classified based on his technical skills set. Since our bone of contention is APT, we distinguish the agent to be a highly skilled attacker with a considerable level of stealthiness and non-traceability.

2) Assets

Assets can represent anything, and the information thereof, that the attacker may need to acquire to use to achieve his final goal. These may include knowledge about the underlying Operating System, the host IP address, port number, banner versions etc. The attacker might use additional tools like network scanners to obtain such information. We give details of such assets as used in our experiment in the next section.

3) Actions

A successfully executed action returns or completes the corresponding asset thereby accomplishing the related goal. Our actions are twofold; one directed towards the network and one against a host. Actions against the network in our setup can be thought of as part of reconnaissance attacks in that they evince which victims are vulnerable to the attack. An action against a host could be the sending of accessibility keystrokes via the network to a host which has RDP port open. Execution of actions is prone to noise generation such as unusual network traffic. It is our assumption henceforth that since the attacking agent is an APT threat actor, the threshold of noise generated by these actions in the course of the attack is low enough not to raise intuitive suspicion.

4) Goals

A goal represents a request knowing all actions which may complete the associated assets. Goals differ depending on the scenario and at times may be nested as a subset of the other. There can be a goal of establishing RDP through TCP on port 3389 but the port scan action has to return a value that port 3389 is open. In the event of security through obscurity, where the default port has been be altered, a banner grabbing action may be required to complete to determine the service running on the obscured port.

B. Attack Tree Modeling

We build an attack tree [11] to show how malware-free intrusion fits amongst other well-known attack vectors as shown in Fig. 2 below. The root node denoted by G0 is the attacker’s ultimate goal of obtaining System Level Access of a given host. This can be achieved by pursing the appropriate attack vectors spawning from the leaf nodes. The subsequent sub-goals (representative of tree nodes) running from G1 through G10 denote the following: G1 - System Access via Network Attack, G2 - System Access via Offline Attack, G3 - Malware-Free Intrusion, G4 - Malware Delivery Attack, G5 - Burglarize Server Room, G6 - Bribe Admin, G7 - Sniff Credentials Online, G8 - Invoke Accessibility Backdoor, G9 - Payload Delivery via Social Engineering and G10 - System Software Vulnerability Exploit. Fig. 2 below shows a lean attack tree with the aforementioned parameters.

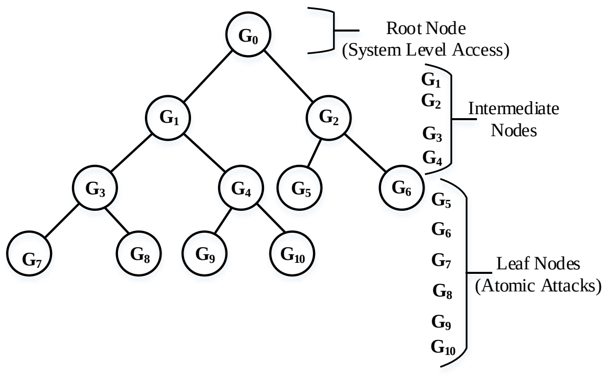


Fig.2. Attack Tree of System Level Access

Child nodes $\mathbf{G_1}$ and $\mathbf{G_2}$ spawn from the parent node $\mathbf{G_0}$ and share a disjunctive Boolean **OR** association. This implicitly entails that the attacker has an option of attaining System Level Access online via the network or offline locally. The intermediate node $\mathbf{G_2}$ spawns two children nodes $\mathbf{G_5}$ and $\mathbf{G_6}$ likewise sharing a disjunctive **OR** relationship implying that the attacker has the option of either bribing the Admin or physically breaking into the server room or wherever the victim host is located (on the assumption that it's not an insider attack). Due to the physical constraints and need for contact imposed by the pursuit of the attack path via node $\mathbf{G_2}$, a remote attacker will most likely pursue the attack path via node $\mathbf{G_1}$. Notwithstanding the aforementioned, we traverse the route via $\mathbf{G_1}$ because our paper's scope is constrained to remote access, thus via the network. This reduces the degree of the root node $\mathbf{G_0}$ from 2 to 1 where the edge $\{\mathbf{G_1},\mathbf{G_0}\}$ is the isthmus.

Seeing that the children nodes of node $\mathbf{G_1}$ share a disjunctive **OR** association, the attack path through this node avails the attacker two options of either using malware or intruding malware-less denoted by nodes $\mathbf{G_4}$ and $\mathbf{G_3}$ respectively. These two nodes contain respective atomic attacks which would otherwise act as the basic actions described earlier in our attack model. The corresponding adjacency matrix of the resulting attack tree of System Level Access via network attack is a square matrix A_G of order 8:

$$AG = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \quad (1)$$

From the adjacency matrix (1), we deduce four attack scenarios emanating from the leaf nodes with the following corresponding paths:

$$P_1 = \{\mathbf{G_7}, \mathbf{G_3}, \mathbf{G_1}, \mathbf{G_0}\}$$
$$P_2 = \{\mathbf{G_8}, \mathbf{G_3}, \mathbf{G_1}, \mathbf{G_0}\}$$
$$P_3 = \{\mathbf{G_9}, \mathbf{G_4}, \mathbf{G_1}, \mathbf{G_0}\}$$
$$P_4 = \{\mathbf{G_{10}}, \mathbf{G_4}, \mathbf{G_1}, \mathbf{G_0}\} \quad (2)$$

The commonality $\{\mathbf{G_1}, \mathbf{G_0}\}$ reflected in all the above paths is due to the fact that the chosen approach for the attack is via the network. As earlier mentioned, malware generates a substantial amount of noise due to unusual network traffic and system calls, therefore a malware-free intrusion is especially attractive to APT. This eliminates the last two paths in (2). Remote access uses the RDP protocol which uses encryption [12] and this means that the attacker most likely does not pursue the route of sniffing admin credential sets off the network thus eliminating the first path in (2). We are therefore left with the path $P_2 = \{\mathbf{G_8}, \mathbf{G_3}, \mathbf{G_1}, \mathbf{G_0}\}$. The attack simulations of the proceeding section are all based on this attack path.

IV. Simulations and Results Analysis

The test-bed of our simulation is elaborated in section 3. We have five victim host all residing in the same network with the attacker in a different subnet. The backdoor created on the victim hosts is via registry alteration where cmd.exe is mapped as the debugger for accessibility suite executable binaries as elaborated in section 2.1. We launch the attacks from Kali Linux. We run the attack in two phases; reconnaissance attack in phase one and the active attack in phase two where we employed an automated script.

A. Phase One: Victim Reconnaissance

In the first phase, we assume the case of an untargeted attack where the attacker does not have prior knowledge of his victims. For this we use the inbuilt Nmap tool in Kali Linux to scan the network for host discovery. We engage Nmap further to scan ports of the discovered hosts and determine whether RDP port 3389 is open. We also changed the RDP port number of one host to an arbitrary 5000 (via registry manipulation of the corresponding Hex value) in order test for security via obscurity against our attack. The results of the Nmap probe are shown in Table 2 below.

Table 2. Nmap Network Probe Results

Host ID	Open Ports/Protocol	Discovered Service
All Hosts	135/TCP	Msrpc
	139/TCP	Netbios-ssn
	445/TCP	Microsoft-ds
10.1.1.2 (Win XP SP2)	3389/TCP	Ms-wbt-server
	123/UDP	Ntp
10.1.1.4 (Win Vista)	5000/TCP	Upnp
	123/UDP	Ntp
	137/TCP	Netbios-ns
10.1.1.5 (Win 7)	3389/TCP	Ms-wbt-server
	137/TCP	Netbios-ns
	5357/TCP	Wsdapi
10.1.1.6 (Win 10)	3389/TCP	Ms-wbt-server
10.1.1.7 (Win 8)	554/TCP	Rtsp
	2869/TCP	Icslap
	3389/TCP	Ms-wbt-server
	5357/TCP	Wsdapi

OS fingerprinting in Nmap identified the operating system of the respective hosts as shown in the Host ID column of table 2.The default open ports common on all hosts are 135, 139 and 445 running the services Msrpc, Netbios-ssn and Microsoft-ds respectively all listening over TCP. The RDP on port 3389 is only available if switched-on on the local host. The network probe further revealed the availability of remote access via RDP listening on TCP port 3389 as shown in bold, running as Ms-wbt-server service. We identified an obscured TCP port 5000 on host 10.1.1.4 which upon further probing of the service's banner turned out to be RDP. This is the host whose default RDP port was altered via the registry.

B. Phase Two: Backdoor Attack

Now that the attacker has knowledge of which hosts are running RDP service for remote access, he needs to establish an RDP session and probe for the availability of the backdoor by initiating the corresponding keystrokes outlaid in table 1. This could be easily achieved by using the rdesktop client [13], native to UNIX and Unix-like hosts. However, rdesktop establishes only a single session and since the attacker's objective is to probe as many hosts as possible, it would be inefficient to pursue this route as it would be manually laborious. We instead engage the script whose algorithm is summarized as:

Algorithm

```
Input: IP Address List ($IP_n$)
Ouput: RDP Session
1.  Read Input list $IP_n$
2.  Check correct format of $IP_n$ items
3.  Initialize $IP_n$
4.  For $IP_i$ less than or equal to $IP_n$
5.  While true; do
6.   Check if host $IP_i$ is alive
7.   else exit
8.  Connect RDP socket $IP_i$:3389
9.   exit if timeout exceeded
10. Send backdoor keystrokes upon session establishment
11. Detect presence of cmd
12. Save result to local directory
13.  End For
```

Fig.3. Algorithm Summary for Backdoor Probing of IP Address List

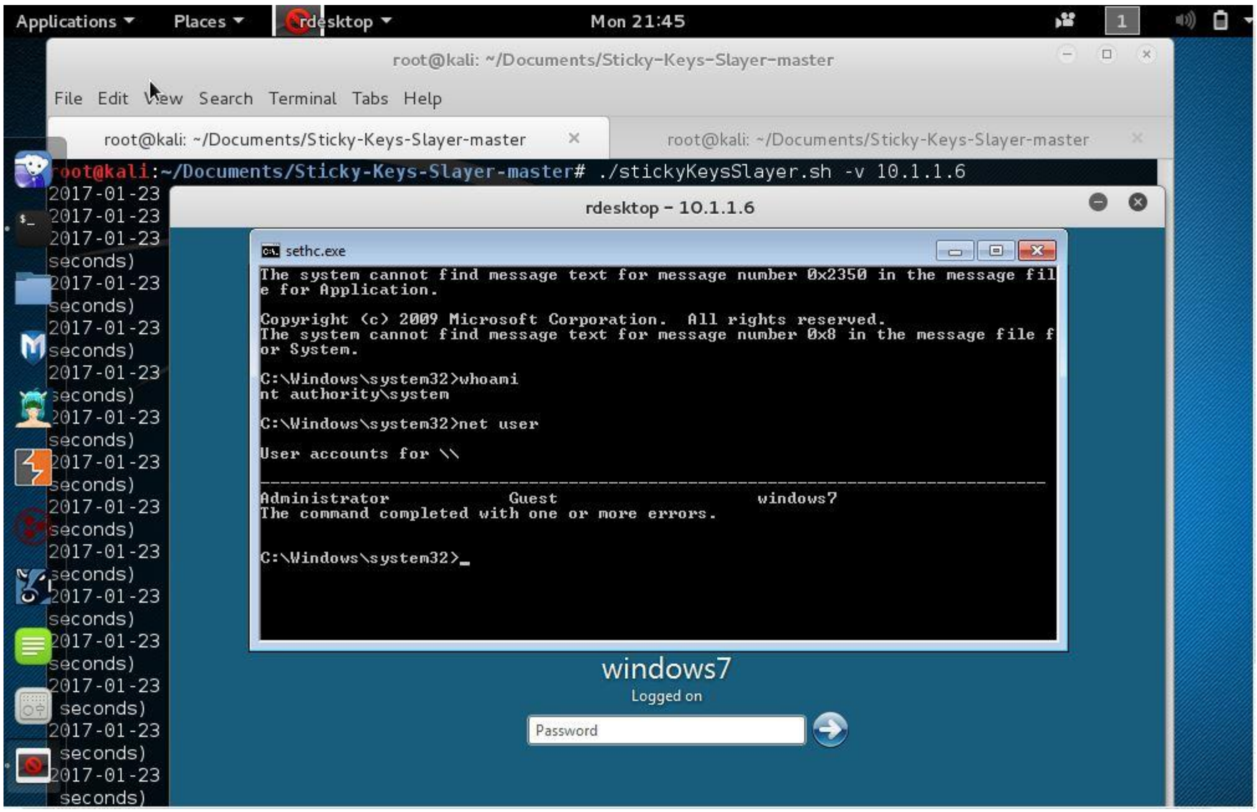
The automated script [14] we use probes for the keystrokes which invoke the backdoor.

We ran the script, on our network depicted in Fig. 1, with the Nmap output host-list in Table 2 as our input. The result was the establishment of multiple RDP sessions with the remote hosts on the IP address list. Fig.4 below shows a snapshot of acquired system level access using the backdoor via an RDP session initiated from the Linux machine of the attacker.

As can be seen in Fig. 4, the "*whoami*" command reveals that the command prompt is running as NT Authority, the highest system level access sought by the APT attacker. This is evident despite the fact that the "*net user*" command lists all the users configured on the remote host which includes the Administrator. At this juncture, the attacker can carry out almost any other task including reconfiguring the users and effect many more global system settings. Further observation shows that the accessibility tool exploited in this instance is the Sticky-Keys as evidenced by the title header of the command prompt. We also take note of the default directory upon remote invocation of the command prompt which is %systemroot%\system32 – the default directory for all pre-authentication accessibility tools' binary executables.

However, the host with the RDP daemon listening on the obscured arbitrary port did not establish an active connection. This is a form of security through obscurity

[15], where the responsible persons that be rely on the obscurity of the implementation to provide a layer of security. The port 5000 appeared to be offering the Upnp service when in the actual fact it was an RDP daemon listening on that port.


Applications
Places
rdesktop
Mon 21:45
root@kali: ~/Documents/Sticky-Keys-Slayer-master
File Edit View Search Terminal Tabs Help
root@kali: ~/Documents/Sticky-Keys-Slayer-master
root@kali: ~/Documents/Sticky-Keys-Slayer-master
rdesktop - 10.1.1.6
sethc.exe
windows7
Logged on
Password

```
root@kali:~/Documents/Sticky-Keys-Slayer-master# ./stickyKeysSlayer.sh -v 10.1.1.6
2017-01-23
2017-01-23
2017-01-23
seconds)
2017-01-23
seconds)
2017-01-23
seconds)
2017-01-23
seconds)
2017-01-23
seconds)
2017-01-23
seconds)
2017-01-23
seconds)
2017-01-23
seconds)
2017-01-23
seconds)
2017-01-23
seconds)
2017-01-23
seconds)
2017-01-23
seconds)
```

```
The system cannot find message text for message number 0x2350 in the message fil
e for Application.

Copyright (c) 2009 Microsoft Corporation.  All rights reserved.
The system cannot find message text for message number 0x8 in the message file f
or System.

C:\Windows\system32>whoami
nt authority\system

C:\Windows\system32>net user

User accounts for \\

-------------------------------------------------------------------------------
Administrator            Guest                    windows7
The command completed with one or more errors.


C:\Windows\system32>_
```

Fig.4. Established Remote Access Session With Backdoor via RDP

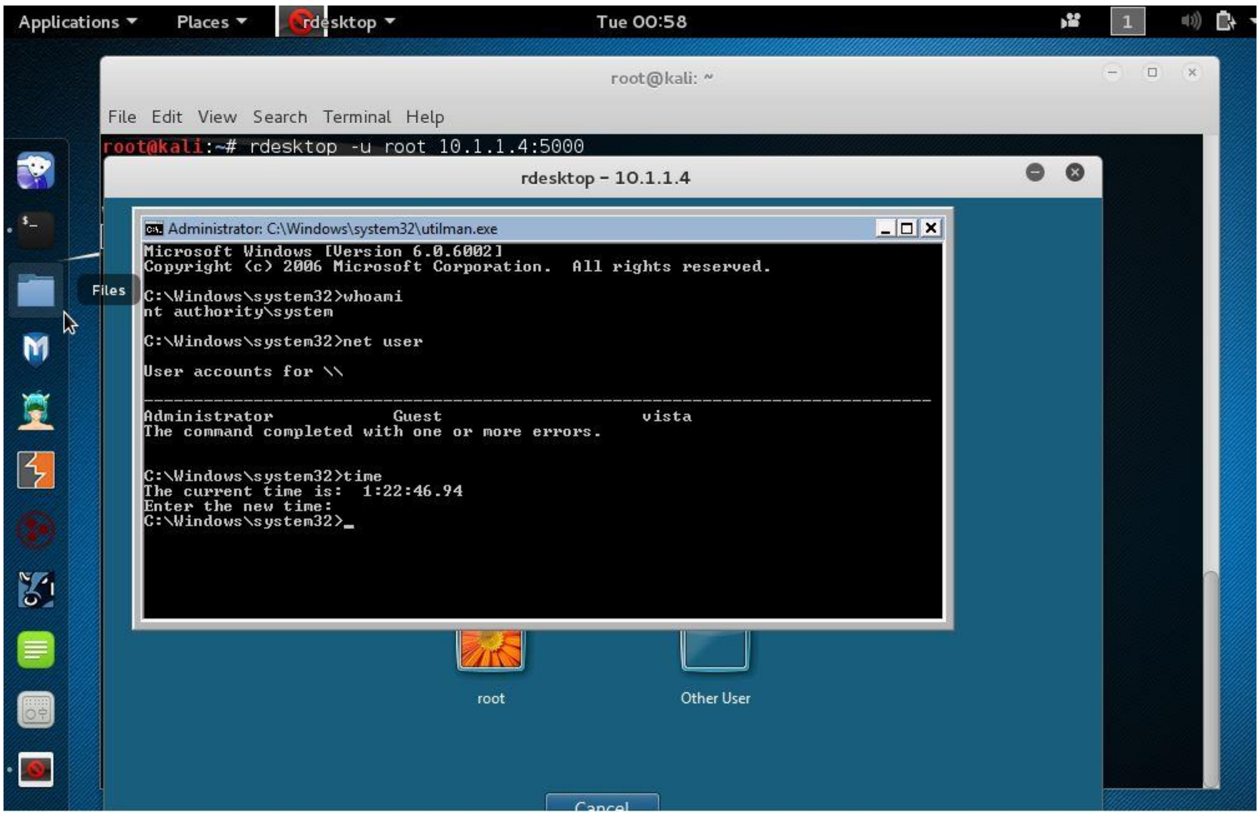
We did not yield any session in this regard because the used script uses the default RDP port to probe for session establishment.

That being said, we had to implement manual banner grabbing for the host in question. We used the rdesktop client available by default on the system to establish an RDP session while specifying to strictly connect on port 5000. The result was a successful normal establishment of a remote access connection and upon further invoking appropriate keystrokes for the backdoor launching, we got the desired command prompt with system level access as shown in Fig. 5 below.

The remote access session established as depicted in Fig. 5 shows the socket in use specifying the obscured port 5000. The "*whoami*" command shows that the default user is the desired system level access. Likewise the title header of the command prompt shows that the exploited accessibility tool for this backdoor is the Utility Manager. The attacker has access to all parts of the system including system files and further plant malware-free backdoors, manipulate system time to invoke other attacks or exploit the system's trust relations with other hosts on the network. This shows that obscuring the RDP port is not an active defense. In fact an attacker might not even need to banner-grab the obscured port if there main aim is to probe remote access availability. On the other hand, the attacker could simply launch a brute force attack on all ports requesting to establish a remote desktop connection. This consequently means that all obscured ports listening for incoming RDP connections would reply to such requests. For further analysis, we in the same manner as in Fig. 4 take note of the default directory upon remote successful invocation of the command prompt which is %systemroot%\system32 as shown above in Fig. 5– the default directory for all pre-authentication accessibility tools' binary executables.

C. Attacks with Network Level Authentication (NLA) Enabled

Originally, Network Level Authentication is meant to prohibit the server offering remote access from sending over a load of resources such as log-in screen before establishing a connection. With NLA disabled, the remote access server offers up a login screen thereby exposing the version of the operating system. NLA prohibits offering such resources before authentication. This, according to the original specification [10] apparently helps thwart denial of service attacks as the server would not use a lot of resources as it would require authentication before anything else. We tested the systems offering remote access with NLA enabled and found that it was not possible to establish any session, neither reach any login screen. The attack does not succeed if NLA is enabled. Notwithstanding the aforementioned, NLA requires the participation of a third entity – an authentication server, a proprietary domain controller in this case. Furthermore, the participating hosts all need to belong to the same domain. This implies that it will not be effective for standalone hosts or users or any other users not part of a specific domain. Even though all our hosts belonged to the same LAN, none of them was able to access the other via remote access with NLA enabled. In addition, NLA does not provide support for other credential set providers.



```
root@kali:~# rdesktop -u root 10.1.1.4:5000
```

```
Microsoft Windows [Version 6.0.6002]
Copyright (c) 2006 Microsoft Corporation.  All rights reserved.

C:\Windows\system32>whoami
nt authority\system

C:\Windows\system32>net user

User accounts for \\

-------------------------------------------------------------------------------
Administrator            Guest                    vista
The command completed with one or more errors.


C:\Windows\system32>time
The current time is:  1:22:46.94
Enter the new time:
C:\Windows\system32>_
```

Fig.5. RDP Session Established Over an Obscured Port

We monitored other parameters on hosts providing remote access so as to contrast how RDP sessions are handled on different operating system versions. We summarize the monitored properties of the all the victims' systems in Table 3 below.

We noticed that the process associated with the remote access service on all the hosts is identical – the SvcHost process. Though the observed PIDs are all only locally significant to the underlying system, the SvcHost handles all RDP services together with other multiple services in each and every implementation of the observed operating system. Therefore, the RDP sessions are handled as services running under an instance of the SvcHost process [16] and not an independent process with a unique PID. Since this process runs multiple instances in memory, the process instance associated with remote access listens on the default port 3389. The specific service for this port inside of SvcHost is Terminal Services. The attacker thus would tell whether a new victim, accessed through other means, has RDP services available by checking not only the default RDP port but also the corresponding Terminal Services running inside of the appropriate instance of SvcHost process. Likewise, the SvcHost process whose default port was obscured to 5000 indeed listened on this port with the underlying Terminal Services providing the remote access service.

The exploited binaries for the backdoor was different for each host as depicted in Table 3. This was easily obtained from the title header of the launched command prompt via the RDP connection. The default directory upon connection for all the backdoors was %systemroot%\system32 which is the core of the operating system. This is coherent with the fact that the exploited file runs from this directory as per Windows security requirements. When the backdoor keystrokes are invoked in the logged in session of a registered user, the access level of the resulting command prompt corresponds to the access level of that user with the default directory being %systemroot%\system32. This is in contrast to NT Authority/System access level obtained if the backdoor is launched over the RDP remote access connection which provides high level system-wide access, a permission level nearly corresponding to the root super-user on Unix and Unix-like systems. This difference in access levels testifies that accessibility tools run under privileged mode at pre-authentication of the system. It therefore follows henceforth that accessing these tools gives full access to the system as demonstrated in this paper.

V. Recommendations and Best Practices

The backdoor via accessibility tools does not generate malicious traffic or issue unusual system calls. This is due to the fact that this exploit uses system files covertly with any suspicious anomalies as earlier discussed. This implies that anti-virus systems, which are based either on signatures or anomaly behavior [17] [19] [20] might find it daunting to locate such an intrusion. Mitigating malware-free intrusions needs a different approach from that taken by the traditional intrusion detection and prevention systems. We now consider some techniques which can be used to mitigate the accessibility backdoor attack for both a host with and without network connectivity.

Table 3. Monitored Parameters on Victim Hosts

Host ID	RDP Daemon PID	RDP Port #	Accessed Backdoor File	Default Backdoor Dir.	Default Local User Level	Default Backdoor User Level
10.1.1.2 (Win XP SP2)	868	3389	Sethc.exe (Sticky Keys)	%systemroot%\system32	Administrator	NT Authority/System
10.1.1.4 (Win Vista)	1360	5000	Utilman.exe (Utility Manager)	%systemroot%\system32	Administrator	NT Authority/System
10.1.1.5 (Win 7)	1224	3389	Magnify.exe (Magnifier)	%systemroot%\system32	Administrator	NT Authority/System
10.1.1.6 (Win 10)	604	3389	EaseOfAccessDialog.exe (Contrast Manager)	%systemroot%\system32	Administrator	NT Authority/System
10.1.1.7 (Win 8)	964	3389	Narrator.exe (Narrator)	%systemroot%\system32	Administrator	NT Authority/System

A. *Detection and Prevention*

Since there are two main act vectors of planting the backdoor, i.e. through file replacement and through registry modification as elaborated in earlier sections, efforts to detect and prevent the backdoor should put these two approaches into perspective. The most practically efficient way of detecting the backdoor is probing for the availability of remote access of a given host. The availability thereof should be probed both on the default port 3389 and all other obscured ports because as demonstrated in previous sections, it is possible to obscure the listening port for remote access via RDP and a port scan can yield a false positive. Invoking the accessibility keystrokes over the established remote access connection should reveal whether the backdoor has been planted. One approach is to detect the presence of the backdoor offline by computing hashes of the %systemroot%\system32 binaries and comparing any matches. Typically, the backdoor exists if any of the accessibility suite executables' hash matches the hash of any other executable (e.g. command prompt executable) with the capability of providing system level access. Such a match should be treated as an indicator of compromise.

Since the second act vector utilizes the registry for backdoor implantation, the second approach to detecting the backdoor would be a thorough registry evaluation and review to check whether any other system executable capable of providing the desired system level access has been set as a debugger for a given accessibility tool. For the two aforementioned approaches of detecting the backdoor, we suggest the use of a scanner [18] for efficiency. In environments with hundreds or thousands of hosts, it wouldn't be practical to manually probe each and every host. Therefore we suggest the use of an automated tool, like the script used in our simulations to detect the presence of the backdoor. It should be noted, however, that such tools might be limited to checking only specified files and not necessarily all those which are associated with the backdoor.

B. *Remediation of the Attack*

The presence of the backdoor is a clear indication of compromise. Mitigation in an instance of occurrence should seek to restore the authenticity of all the %systemroot%\system32 binaries and not only those associated with accessibility tools since it's apparent that any service and file available during pre-authentication can be exploited for the backdoor. We recommend absolute deletion of the affected files and replacing them with trustworthy files from a clean source.

In as far as registry alterations are concerned, mitigating this attack vector calls for a registry integrity check. This should without fail be accompanied by removal of all registry entries associated with the backdoor. Keeping a registry snapshot of a healthy system and comparing any modifications thereof should help identify any such malware-free intrusion.

VI. Conclusion

Malware-free intrusion do not present a new set of attacks but are rather part of attack cycles. However, they are proving attractive to APT attacks due to their stealthiness. We presented in this paper a malware-free intrusion that leverages the exploitation of built-in system tools. The intrusion, in form of a backdoor accessible over a remote access connection affects all versions of the Windows operating system from Windows XP to the current version of Windows 10. The attack is possible because despite all the security measures in place concerning the Systemroot, the operating system does not check the context of command prompt invocation at pre-authentication even over a remote desktop connection. Obscuring the port for remote access does not thwart the attack as specifying a connection with the obscured port yields a connection. The SvcHost process runs the Terminal Services that listen for RDP connections. Attackers can back trace this process for determination of the corresponding PID and obscured port. The user privilege level availed when the backdoor is run locally and over an RDP connection differs: the user over RDP connection, hence attacker, has the highest level of access higher than the administrative user of the system. This verifies that the accessibility tools accessed at pre-authentication run at system level permission.

The presence of the backdoor is a clear indication of compromise. The two attack vectors utilized by this backdoor are via system file replacement and registry modification. Detecting the backdoor calls for hashing and detecting any matching collisions of the hash. No two system files in the systemroot should share the same hash as this is an indicator of compromise. Review of registry entries to detect whether any of the system files capable of providing system level access is set as the debugger for any of the system binaries activated at pre-logon should be able to detect the backdoor. NLA thwarts this attack but the constraints imposed by the security framework supporting NLA makes it impractical to activate it on every system using remote access.

ACKNOWLEDGMENT

This work is supported by the University of Science and Technology Beijing.

R EFERENCES

[1] Mohammad Rasmi, Ahmad Al-Qerem,"PNFEA: A Proposal Approach for Proactive Network Forensics Evidence Analysis to Resolve Cyber Crimes", IJCNIS, vol.7, no.2, pp.25 -32, 2015.DOI: 10.5815/ijcnis.2015.02.03.

[2] Tankard Colin. "Advanced persistent threats and how to monitor and deter them." Network security 2011.8 (2011), pp. 16-19.

[3] Virvilis Nikos and Dimitris Gritzalis. "The big four-what we did wrong in advanced persistent threat detection?." Availability, Reliability and Security (ARES), 2013 Eighth International Conference on. IEEE, 2013.

[4] Ashutosh Gupta, Bhoopesh Singh Bhati, Vishal Jain,"Artificial Intrusion Detection Techniques: A Survey", IJCNIS, vol.6, no.9, pp.51-57, 2014. DOI: 10.5815/ijcnis.2014.09.07.

[5] Daniel Gonzales, Jeremy Kaplan, Evan Saltzman, Zev Winkelman and Dulani Woods "Cloud-trust-a security assessment model for infrastructure as a service (IaaS) clouds." IEEE Transactions on Cloud Computing. (2015).

[6] Zach Grace. "Hunting Sticky Keys Backdoors" (March 2015) [Online] Available: https://zachgrace.com/2015/03/23/hunting-sticky-keys-backdoors.html

[7] Longzheng Cai, Yu Shengsheng, and Zhou Jing-li. "Research and implementation of remote desktop protocol service over SSL VPN." In Services Computing, 2004. (SCC 2004). Proceedings. 2004 IEEE International Conference. (2004), pp. 502-505.

[8] Remote Desktop Protocol. (2016) [Online] Available: https://msdn.microsoft.com/en-us/library/aa383015(v=vs.85).aspx [Accessed 1st November 2016].

[9] Durumeric Zakir, Michael Bailey, and J. Alex Halderman. "An internet-wide view of internet-wide scanning." In 23rd USENIX Security Symposium (USENIX Security 14). (2014), pp. 65-78.

[10] "Network Level Authentication for Remote Desktop Services Connections." [Online] Available: https://technet.microsoft.com/en-us/library/cc732713.aspx [Accessed 20th November 2016].

[11] Mauw Sjouke and Martijn Oostdijk. "Foundations of attack trees." International Conference on Information Security and Cryptology. (CISC 2005). Springer Berlin Heidelberg (2005), pp. 186-198

[12] "Remote Desktop Protocol". Microsoft. (March 30, 2009). [Online] Available: https://msdn.microsoft.com/en-us/library/aa383015(VS.85).aspx [Accessed 10th December 2016]

[13] “Rdesktop” (2017). [Online] Available: http://www.rdesktop.org/ [Accessed 4th January 2017]

[14] “Sticky-Keys-Slayer” (2017). [Online] Available: https://github.com/linuz/Sticky-Keys-Slayer [Accessed 2nd January 2017]

[15] Duraiswamy, K., and R. Uma Rani MCA. "Security through obscurity." KSR College Of Technology, Tiruchengode (2005).

[16] Russinovich Mark, Solomon David, Ionescu Alex. "Windows® Internals". (5th ed.), Microsoft Press, ISBN 0-7356-2530-1. (2009).

[17] Sourabh Saxena. "Demystifying Malware Traffic." SANS Institute InfoSec. (August 2016)

[18] Sticky-keys-scanner. [Online] Available: https://github.com/TrullJ/sticky-keys-scanner [Accessed 4th January 2017].

[19] Rieck Konrad, Philipp Trinius, Carsten Willems, and Thorsten Holz. "Automatic analysis of malware behavior using machine learning." Journal of Computer Security 19, no. 4 (2011), pp. 639-668.

[20] Alazab Mamoun, Sitalakshmi Venkataraman and Paul Watters. "Towards understanding malware behaviour by the extraction of API calls." In Cybercrime and Trustworthy Computing Workshop (CTC), 2010 Second, pp. 52-59. IEEE, (2010).

Authors’ Profiles

Aaron Zimba is currently a PhD student at the University of Science and Technology Beijing in the Department of Computer Science and Technology. He received his Master and Bachelor of Science degree from the St Petersburg Electrotechnical University in St Petersburg in 2009 and 2007 respectively. He is also a member of the IEEE. His main research interests include Network and Information Security, Cloud Computing Security and Network Security Models.

Zhaoshun Wang is a Professor and the Associate Head of the Department of Computer Science and Technology at the University of Science and Technology Beijing. He graduated from Department of Mathematics at Beijing Normal University in 1993. He received his PhD from Beijing University of Science and Technology in 2002. He completed postdoctoral research work at the Graduate School of the Chinese Academy of Sciences in 2006. He holds patents and has many awards to his name. His main research areas include Information Security, Computer Architecture and Software Engineering.

How to cite this paper: Aaron Zimba, Zhaoshun Wang,"Malware-Free Intrusions: Exploitation of Built-in Pre-Authentication Services for APT Attack Vectors", International Journal of Computer Network and Information Security(IJCNIS), Vol.9, No.7, pp.1-10, 2017.DOI: 10.5815/ijcnis.2017.07.01