Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 7943586, 13 pages
https://doi.org/10.1155/2018/7943586

Research Article

Automatically Traceback RDP-Based Targeted Ransomware Attacks

ZiHan Wang,[1] ChaoGe Liu,[1] Jing Qiu,[2] ZhiHong Tian,[2] Xiang Cui,[2] and Shen Su[2]

[1]*Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China*
[2]*Cyberspace Institute of Advanced Technology Guangzhou University, Guangzhou, China*

Correspondence should be addressed to ChaoGe Liu; liuchaoge@iie.ac.cn, Jing Qiu; qiujing@gzhu.edu.cn, and ZhiHong Tian; tianzhihong@gzhu.edu.cn

Received 13 July 2018; Revised 24 October 2018; Accepted 22 November 2018; Published 6 December 2018

Guest Editor: Vishal Sharma

While various ransomware defense systems have been proposed to deal with traditional randomly-spread ransomware attacks (based on their unique high-noisy behaviors at hosts and on networks), none of them considered ransomware attacks precisely aiming at specific hosts, e.g., using the common Remote Desktop Protocol (RDP). To address this problem, we propose a systematic method to fight such specifically targeted ransomware by trapping attackers via a network deception environment and then using traceback techniques to identify attack sources. In particular, we developed various monitors in the proposed deception environment to gather traceable clues about attackers, and we further design an analysis system that automatically extracts and analyze the collected clues. Our evaluations show that the proposed method can trap the adversary in the deception environment and significantly improve the efficiency of clue analysis. Furthermore, it also helps us trace back RDP-based ransomware attackers and ransomware makers in the practical applications.

1. Introduction

Ransomware was first emerged in late 1980s [1, 2] and has resurfaced since 2013 [3]. Recently, several wide-spread ransomware attacks have caused significant damages on a large number of user systems and businesses on the Internet. *Symantec* reported a 250% increase in new crypto ransomware families between 2013 and 2014 [2]. In May 2017, *WannaCry* spread across more than 150 countries and 200,000 computers in just a few days, and severely disrupted many businesses and personal systems [4, 5]. In addition, *specifically targeted ransomware* like *Crysis* disrupted many small and large enterprises across the globe; e.g., *Trend Micro* observed that the *Crysis* family specifically targeted businesses in Australia and New Zealand in September 2016. The number of such targeted ransomware attacks was doubled in January 2017, compared with in late 2016 [6]. What is more, the lack of focus on security has left IoT (Internet of Things) devices vulnerable, which has been the target of 10% of all ransomware attacks. Researcher predicts IoT ransomware attacks being likely to increase to around 25% to 30% of all ransomware cases [7].

Because traditional ransomware was typically spread randomly without specific targets via network scanning or host probing, they can be easily detected by monitoring of the abnormal behaviors in host activities such as file system operations and network traffic [1, 3, 8]. Recently, more and more ransomware attacks aimed at specific targets. *Kaspersky Security Bulletin* indicated that targeted attacks have become one of the main propagation methods for several widespread ransomware families in 2017 [9, 10]. For instance, an attacker using *Crysis* ransomware first logs in a victim's host and spreads itself via a brute force attack on the common Remote Desktop Protocol (RDP). Such a targeted ransomware attack usually has a clear command-and-control structure and aimed at resource exploitation and resource theft on these targets, while generating fairly limited noisy on hosts and networks which is hard to detect.

Existing ransomware defense methods (designed for dealing with randomly-spread attacks) usually protect a host by blocking the spreading of ransomware attacks (in nearly real-time) based on the signatures generated by ransomware detection solutions. However, because of the different characteristic of targeted ransomware attacks with less notable patterns, these traditional blocking-based defense systems become much less effective for these targeted attacks.

To address this issue, we propose to utilize advanced defense schemes to protect important hosts under targeted ransomware attacks. In this paper, we utilize the *cyber deception technology* to help us protect critical systems through attack guidance, by drawing attackers away from these protected systems. While the cyber deception technology helps us protect important targets (such as in dealing with the Advanced Persistent Threat (APT) [11, 12]), it cannot help us traceback attack sources. To address this issue, we further design specific techniques to traceback RDP-based ransomware attacks and identify the original attack sources as the main deterrence of ransomware attackers.

Our deception environment simulates an actual user system in three layers with multiple monitors to observe various key system operations, related to login, network communication, clipboard, process, shared folder, and file system. It collects traceable clues and helps us detect the RDP ransomware attack. Because traditional traceback methods usually require security experts to manually analyze a large amount of collected clues, it is difficult for make them to achieve fast responses. Therefore, we develop an *automatic analysis system* to work on traceable clues by taking advantage of natural language processing and machine learning techniques.

To evaluate our system, we invite 122 volunteers in a simulated RDP-based ransomware attack. The proposed system was able to capture traceable clues through the proposed deception environment. It can also automatically analyze the clues effectively. The convergence rate of the analysis system reaches about 98%. Moreover, we demonstrated that it helps us traceback RDP-based attack sources in practical applications.

In summary, this paper makes the following contributions:

(i) We propose a systematic method to deter RDP-based ransomware by identifying attackers, which traps ransomware attackers via a cyberdeception environment and uses an automatic analysis system to obtain traceable clues and identify attack sources.

(ii) We build a deception environment to trap RDP-based ransomware attacker, by simulating an user environment in three layers: a network layer, a host layer and a file system layer. The environment helps us discover attacker behaviors and collects attacker-related information.

(iii) We develop an automatic analysis system with natural language processing and machine learning techniques to automatically recognize effective clues for tracing back ransomware attack sources.

(iv) We designed two practical experiments to test RDP-based ransomware attacks and ransomware makers, and demonstrated the feasibility of the proposed system.

The remainder of the paper is structured as follows. In Section 2, we briefly present background and related work. In Section 3, we describe the methodology of our systematic method. In Section 4, we present the implementation of our deception environment prototype. In Section 5, we describe the details of the clue analysis system. In Section 6, we discuss the evaluation setup and results. We conclude this paper in Section 7.

2. Background and Related Work

2.1. Related Work on Ransomware Defense. Ransomware is a type of malware which manipulates an user system to extort money. It operates in many different ways, e.g., simply locking a user's desktop or encrypting an entire file system. Recent rampant ransomware attacks have called for effective ransomware defense solutions. In the studies that tackle ransomware counteraction, several solutions are proposed to confront this attack [14, 15].

Some of these solutions are proposed to deal with all types of ransomware [1, 16–20]. For example, *Kharraz* presented a dynamic analysis system called *UNVEIL*. The system analyzes and detects ransomware attacks by modeling ransomware behaviors. It focuses on the observation of three elements, namely, I/O data buffer entropy, access patterns and file system activities [1]. Moreover, some others are type-specific solutions that deal with only one type, such as crypto-ransomware [21–25]. For example, *Scaife* presented an early-warning detection system that alerts users during suspicious file activities [21]. Utilizing a set of behavior indicators, the detection system can halt a process that appears to tamper with a large amount of user data. Furthermore, it is claimed that the system can stop a ransomware execution with a median loss of only 10 files. Similarly, some studies tackle the detection of specific ransomware families only [26–28]. For example, *Maltester* is a family-specific technique proposed by *Cabaj* to detect *Cryptowall* infections [27]. It employs dynamic analysis along with honeypot technology to analyze the network behavior and detect the infection chain.

These solutions can be categorized into prevention and detection. However, these two kinds of countermeasures have the following disadvantages. First, many prevention measures require many services to be disabled, which is likely to affect service functionality. For example, *Prakash* suggested several prevention measures including disabled macros in office documents, and restricted access permissions on "Temp" and "Appdata" folders [29]. Secondly, the detecting system is often difficult to conceal itself and perform its functions when against ransomware attacks that precisely aiming at specific hosts, e.g., using the common Remote Desktop Protocol (RDP). Finally, while these countermeasures can be used to detect or block specific ransomware attacks, they cannot fundamentally inhibit the spread of ransomware. But traceability technology can fundamentally inhibit the ransomware spread by traceback to attack sources.

2.2. RDP-Based Ransomware Attacks. Traditional ransomware randomly spreads across the Internet, in executable files, development kits, macro files, and other malicious programs on a large scale with various dissemination methods, including phishing emails, puddle attacks, vulnerability attacks, server intrusion, and supply chain pollution. They

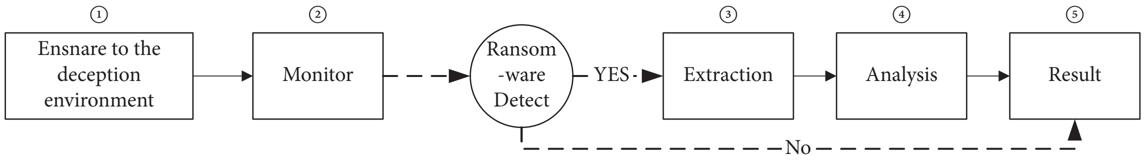


Figure 1: Data collection and analysis process of the whole prototype.

use different ways to trick a victim to launch such programs. Among these dissemination method, phishing emails is the most widely used. However, according to Kaspersky's 2017 ransomware report, the number of targeted ransomware attacks based on RDP is growing rapidly.

Recently, more and more ransomware criminals have spread ransomware using RDP services and then installed ransomware manually. These attackers use a brute-force method to acquire usernames and passwords on a target machine with an active RDP service [9]. For instance, one of the typical families, *Crysis*, a copycat of *Locky*, not only aims at common businesses but also targets healthcare service providers [9]. *Crysis* gains access to admin level privileges by stealing passwords and credentials. In addition, during an RDP session, the attacker uses both clipboard and shared folders to upload files to a remote host. And then, attackers can installed ransomware manually.

2.3. Cyber Deception Technology. Because many critical systems are known and always on, it is difficult to protect them from potential network attacks.

(i) Attackers can use zero-day vulnerabilities, highly antagonistic malicious code, or other resources to break the defense system.

(ii) Because humans are always the weakest link in defense systems, attackers can use social engineering to identify system weaknesses and penetrate the defense.

(iii) Attackers can repeatedly explore the potential vulnerabilities on a target system to identify its weaknesses.

However, when an attacker aims at a specific target, e.g., exploiting its RDP service, traditional passive defense methods cannot be usually less effective. Therefore, we need to use advanced active solutions to deal with such attacks with less observable features, such as cyber deception.

The earlier use of cyber deception technology is honeypot. Honeypot detects attacks by deploying a series of systems or resources in the service network that do not have real business. When a trap is accessed, it represents an attack. Honeypot system generally waiting attacks passively, and does not have the role of misleading and confusing attackers. What's more, the honeypot system does not have real business, and does not have high interactive characteristics, which may easy to be identified by attackers. Compared with the traditional honeypot system, the cyber deception system can be deployed more conveniently, the cyber deception environment is more real, and can be linked with existing defense products. It can provide more effective solutions for APT attacks, ransomware attacks, intranet attacks and other threats defense. A Gartner report in 2015 [30] pointed out the market prospect of deception-based security defense technology and predicted that 10% of organizations will use deception tools (or tactics) to counter cyber-attacks in 2018. Compared with the traditional passive defense approach, cyber deception technology is an active defense approach and can be applied to all stages of network attacks. We can use this technology to trap the RDP-based attacker, detect targeted attacks, and deter ransomware attackers by precisely identifying them.

Trap RDP-Based Ransomware Attackers. A targeted ransomware attack generally has three steps: detection, infiltration, and execution [31]. However, traditional security solutions are unable to cope with the internal translation phase. In addition, traditional honeypot technology (often used to fight network attacks) generally does not focus on tracing back to attackers. However, cyber deception technology can deceive the attacker into a surveillance environment and consume his time and energy with bait information.

Detect RDP-Based Ransomware Attacks. Once the attacker obtains the correct username and password combination, he usually returns multiple times within a short period to try and infect the compromised host [6]. In one particular case, *Crysis* was deployed six times on an endpoint within a span of 10 minutes. As a result, by monitoring in the cyber deception environment, we can detect RDP-based ransomware attacks in time and determine the attacker's behavior through the environment monitor.

Deter the Ransomware Attacker. Deterring ransomware attackers can be approached in two ways. First, if an attacker realizes he is entrapped, it becomes a deterrent. Second, if the attacker is exposed to the deception environment and remains within the perspective of the defense surveillance, the monitor can collect the attacker's traceable clues that are accidentally released by the attacker (e.g., IP address, path, nickname, strings). The exposure of these clues, hidden from attackers, can be a powerful deterrent to other attackers.

3. Methodology

In this section, we describe our method of tracing back RDP-based ransomware attackers. Figure 1 summarizes the data collection and analysis process of the entire prototype. First, we implement a deception environment to trap attackers. Second, we monitor RDP-based ransomware attacks and collect information when they occur. Third, we extract effective clues from the monitor information. Fourth, we use automatic analysis to screen a large number of clues for tracing back the attacker. Finally, we will generate a report to traceback the RDP-based ransomware attacker. We refer readers to Sections 4 and 5 for the detailed implementation of this prototype.

3.1. Deception Environment. Generally, the ransomware attack execution stage has two steps: login and spread [31]. To build a deception environment is nontrivial in practice because it must make the ransomware attacker believe that it belongs to a real user and the user data is worthy to attack. Because advanced attackers always exploit static features based on certain analysis systems before they launch attacks [32], an intuitive approach to address such reconnaissance attacks is to build the user environment in such a way that the user data is valid, real, and nondeterministic. In addition, the environment serves as an "enticing target" to encourage ransomware attackers. We elaborate on how to generates an artificial, realistic, and enticing user environment for the RDP-based ransomware in Section 4.

The RDP-based attackers commonly upload malicious programs in the following ways before spread ransomware: (1) The attacker downloads malicious programs on the Internet; (2) programs are transferred through FTP, SCP, or other transport protocols; (3) programs are uploaded through the clipboard; (4) programs are uploaded through a shared folder. The clipboard and shared folders are most commonly used to transfer programs by RDP-based ransomware attacks because they are simple and convenient. However, both are easy to monitor by our proposed system.

3.2. Environment Monitor. In order to avoid the attacker's observation and collect more attacker's information, a shared folder and clipboard on the remote PC are always used to transfer ransomware programs from the attacker machine after the attacker logs in to the environment. This paper proposes three monitor layers: the network layer, the host layer and the file layer. We elaborate on how to configure the monitor system for the deception environment in Section 4.

The Network Layer Monitor. The network layer monitor detects a remote connection and collects information including the remote IP addresses, remote ports, status codes of ports, keyboard layout and so on. When the RDP-based attacker logs in to the host, the monitor can obtain information and detect the attack without the attacker's knowledge.

The Host Layer Monitor. We propose to detect changes such as processes and clipboards, by monitoring the host layer. The host layer monitor can gather information about the attacker's behavior and their use of these system applications in the deception environment. For instance, as the clipboard is in the system-level heap space, any application in the system has access to it. The RDP-based ransomware always takes advantage of the clipboard to interact between applications. Moreover, it might get the clues left by the attackers using the clipboard locally, as the Windows system shares the clipboard by default during the RDP session.

The File Layer Monitor. By monitoring the file layer, we can identify ransomware attacks by file changes. Furthermore, it can gather local traceable information by monitoring files in the shared folder. For instance, as a shared folder on the remote PC is always used to transfer ransomware from the attacker machine during the RDP session. In addition, for a more convenient and quick attack, the attacker often mounts the entire local disk to the remote computer. As a result, through the monitor of the shared folder, we can detect the newly-added shared folders in real time and capture a large amount of path information automatically.

3.3. Clue Extraction. Through environmental monitors, it can gather a lot of information left by attackers, such as login information, communication information, clipboard content, folder path, and portable execution (PE) file Many traceable clues can be extracted here, including but not restricted to IP address, keyboard layout, compile path, and file path. In order to analyze these clues quickly, we divided them into two categories: string clues and path clues. These clues are then submitted to the automatic analysis system. We will elaborate the types of clues that the proposed system can extract in Section 5.1.

3.4. Automatic Analysis. According to our investigation, current traceback tools mostly analyze clues manually. However, we usually have to deal with a large amount clues with no semantic correlation. Because such manual traceback analysis usually takes a lot time and efforts, we propose an automatic analysis system and we will elaborate on how to analyze clues automatically in Section 5.2.

4. Implementation of the Deception Environment

As the Windows platform is the main target of ransomware, we choose Windows as the proof of concept implementation. In this section, we describe the implementation details of a Windows-based deception environment prototype. It elaborates on how the deception environment traps ransomware attackers, how the monitor detects the RDP-based attack and collects traceable clues. The entire system implementation process is shown in Figure 2.

4.1. At the Network Layer

4.1.1. The Login Monitor. The login monitor is used to detect attacks in real time and collect the attacker's login information. On the Windows platform, Win32 is an environment subsystem that provides an API for operating system services and functions to control all user inputs and outputs. The login monitor relies on Windows APIs to gain access to the system and run with privileges to access their own areas of memory. It uses Winsock 2.0 to get access to networks and uses protocols other than the TCP/IP suite. The login monitor takes network requests and sends those requests to the Winsock 2.0 SPI (Service Provider Interface) by calling the main Winsock 2.0 file Ws2_32.dll. It provides access to transport service providers and namespace providers. The

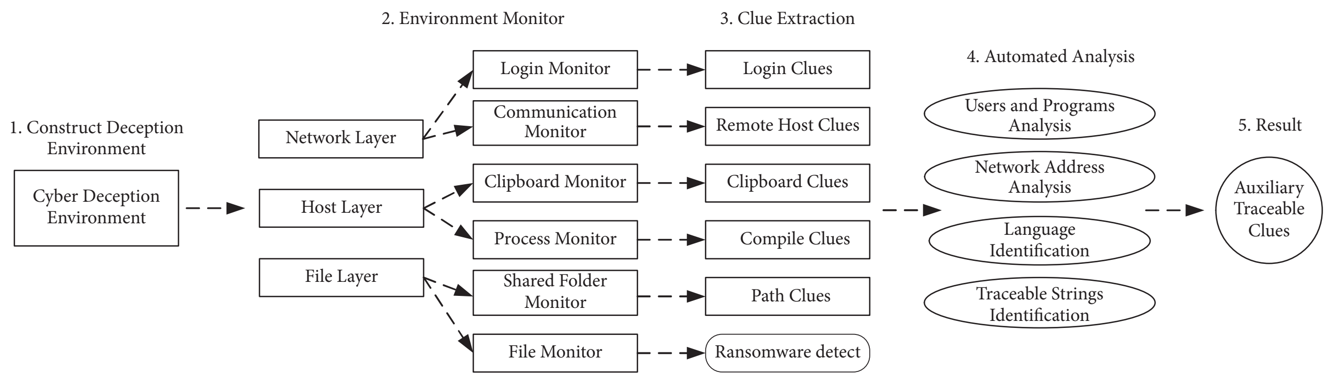


Figure 2: RDP-based ransomware attack traceback system process.

IP Helper API makes it possible to get and modify network configuration settings for the localhost. It consists of the DLL file iphlpapi.dll and includes functions that can retrieve information about the protocols, such as TCP and UDP [33]. As a result, the login monitor can directly access the data buffers involved in transmission control protocols, routing tables, network interfaces, and network protocol statistics.

4.1.2. The Communication Monitor. The communication monitor is responsible for captures network traffic. By locating the TCP packets in the network traffic which contain the interactive configuration information of a RDP login, the RDP connection information can be obtained as the attacker's personal information. The main packet characteristics: (1) The packet's name is often ClientData. (2) The packet location is usually after the TCP three-way handshake. (3) The data packet position in the front. (4) The amount of data is significantly larger.

4.2. At Host Layer

4.2.1. The Deception Host. According to our observation, the main methods used by attackers to login a remote host are: (1) weak password direct login; (2) add access account login through vulnerability. As a result, we deliberately set the administrator privileges of the deception environment as weak passwords and leave common vulnerabilities in the environment, such as *EternalBlue*, to attract attackers. To guide attackers to upload ransomware using only clipboard and shared folders, we block external traffic transfers from outside the environment and close common transfer ports (e.g., port 20, 21, 80, and 443).

4.2.2. The Clipboard Monitor. The clipboard monitor can obtain clues in real time by monitoring the clipboard's changes. It uses Clipboard Viewer to listen to message changes in the clipboard without affecting its contents. The Clipboard Viewer is a mechanism that can get and display the contents of the clipboard. As Windows applications are message-driven, the key to the monitor is responding to and processing clipboard change messages. When the content changes, the monitor triggers the *WM_DRAWCLIPBOARD* message and sends the changed message to the first window of the Clipboard Viewer Chain. After each Clipboard Viewer window responds to and processes the message, it must send the message to the next window according to the handle of the next window in its saved linked list. The clipboard monitor can obtain the clipboard's new contents by using the "*GetClipboardData*" Windows API through the window. When a subsequent copy or cut operation are executed, the data in the clipboard are rewritten. As a result, the clipboard monitor guarantees real-time listening and writes real-time information to the log file. The log file is updated whenever the clipboard monitor receives a clipboard change notice. When the log file is updated, to prevent it from being detected by the attacker or being encrypted by ransomware, the monitor sends it to a secure host and completely erases it from the environment.

4.2.3. The Process Monitor. To run a program on a Windows system, a new process must be created. The monitor gets the PE file run by the attacker by monitoring the environment process. It first records the state of processes commonly used in the deception environments before a RDP-based ransomware attack. After the login monitor detects such an attack, it monitors the system for newly created processes in the environment through the Windows API. "*CreateToolhelp32Snapshot*" takes status snapshots of all processes in real time, which includes the process identifier (PID). When a suspicious process has started, the monitor recognizes it by the PID and looks up the process's running path with the help of "*GetModuleFileNameEx*". Finally, the monitor finds the suspicious program's PE file through the path and copies it to the secure host.

4.3. At the File Layer

4.3.1. Deception Files. Deception files are constructed with two goals. First, we need to make the attacker believe that the deception environment is a real user's host. Second, we need to make the attacker believe that there are resources in the environment that are worthy of attacking.

To simulate a realistic environment, we deploy large numbers of different types of files on it, for example, images, audio files, database files, and documents that can be accessed in a Windows session. Based on *Amin Kharraz's* research [1], we created four file categories that ransomware always tries to find and encrypt: documents (∗.txt, ∗.doc(x), ∗.ppt(x), ∗.xls(x), ∗.pdf and ∗.py), keys and licenses (∗.key, ∗.pem, ∗.crt, and ∗.cer), file archives (∗.zip, ∗.rar), and media (∗.jp(e)g, ∗.mp3, and ∗.avi). We obtain these files in three ways. First, we create files with valid headers and content using standard libraries (e.g., python-docx, python-pptx, pdfkit, and OpenSSL). Second, using Google search syntax and crawler technology, we download a large number of files on the Internet. Third, we collect a number of non-confidential documents from the hosts of 20 volunteers to emulate actual user environments. When we assign user files for the deception environment, the path length is generated randomly. Each folder may have a set of subfolders randomly. For each folder, a subset of extensions is randomly selected. Furthermore, each directory name is generated based on meaningful words. Consequently, we generate paths and extensions for user files, giving them variable file depth and meaningful content.

To make the simulated environment more valuable, we deploy bait information on it, such as database, false code comments, digital certificates, administrator password, SSH keys, VPN keys, browser history passwords, ARP records, and DNS records. When the bait information is obtained by an attacker, it may trick it to attack the deception environment.

4.3.2. The File Monitor. The file monitor can detect the ransomware by monitoring file type changes and file entropy changes, which method is proposed in 2016 [21]. The type of data stored in a file can describe the order and position of specific byte values unique to a file type. Since files generally retain their file types and formatting in the deception environment, the bulk modification of such files should be considered suspicious. When the monitor sees this type of changes, we can infer that a ransomware attack has occurred.

Entropy can express the randomness of each character in a string. The higher the entropy value, the stronger the randomness. The Shannon entropy of an array of bytes can be computed as the sum:

$$e = \sum_{i=0}^{255} P_{B_i} \log_2 \frac{1}{P_{B_i}} \qquad (1)$$

for $P_{B_i} = F_i / totalbytes$ and F_i, the number of instances of byte value i in the array. As the entropy value is represented by a number from 0 to 8, the entropy value of 8 represents the byte array composition of its completely uniform distribution. Since the probability of each byte occurring in the encrypted ciphertext is basically the same, the entropy value will be close to the upper limit. Because the ransomware always encrypts a large number of files, when we detect that a file change to a high entropy value file in a short period of time and also change the file type, we assume that the file is subject to a ransomware attack.

4.3.3. The Shared Folder Monitor. By traversing the disk storage in real time, the shared folder monitor discovers the updates of the shared folders in real time. It obtains the contents of the attacker's files locally, which are often not noticed by the attacker, thus revealing some unexpected traceable clues. The monitor can access a list of paths to the attacker shared folders.

As we originally observed, shared folders using Remote Desktop often have a path in the remote host with the prefix "\\tsclient\". When the monitor traverses the storage to this prefix, it uses "*FindFirstFile*" to find the first file. It then uses "*FindNextFile*" to find the next file with the returned handle. When the resulting handle is in a folder format, it continues to traverse all files under that folder. Initially, the monitor tries to get the full file names and file contents by traversing the new shared folder. However, during the actual experimentation, it is found that as the number of files in storage grows, the monitor takes far more time and resources to get all the file contents than just the file paths. All traverses are more likely to alert the attacker. Therefore, the monitor only obtains the file paths that the attacker shares on its host with the help of "*GetFileName*". Moreover, in order to prevent encryption by the ransomware, the mounted disk monitor will directly transfer the acquired shared file path list to another secure host.

5. Clue Extraction and Analysis

Through the deception environment, we trap the ransomware attacker and collect a lot of information that may contain many traceable clues. However, such traceable clues are often not visually observable and are complex in nature. In addition, many of the above clues contain information that is not helpful in tracing back ransomware attackers. Therefore, in order to assist in the analysis of the monitor information and extract the effective main traceable clues, in this section we propose how to extract clues and how to analyze traceable clues using an automatic approach after extraction.

5.1. Clue Extraction. We mainly obtain kinds of clues from the extraction, including remote login information (IP addresses), network traffic, clipboard contents (pictures and texts), shared folder information (path strings), and ransomware samples (compile time and compile paths). The shared folder path clues can be obtained directly from the monitor. However, clipboard clues, compile clues, and remote host clues are often not visually observable and are complex in nature. As a result, the extraction module mainly focuses on the extraction of remote host clues, clipboard string clues, and compilation clues.

Remote Host Clue Extraction. The IP address, port numbers, and folder path clues can be directly obtained from the login information and folder paths. TCP packets that interact with the configuration information in a network communication PCAP package are usually named "*ClientData*". We extract the client name field from the "*ClientData*" packet to obtain the attacker's hostname. In addition, the *KeyboardLayout* field indicates the default keyboard layout for an attacking host; e.g., the Chinese Simplified layout number is 0x0004, and the American English keyboard layout number is 0x0409. The remote user's idiomatic language (the mother tongue) can be found by the keyboard layout to infer the attacker's nationality.

Clipboard Clue Extraction. The main file formats available to the clipboard monitor are Windows Bitmap, GDI file, ANSI characters, Unicode characters, and WAV audio data. We mainly aim at extracting the traceable clues of character types. It extracts character clues from the clipboard in various formats by judging the *GetClipboardData* API's "*DataType*" value.

Compilation Clue Extraction. For all Windows RDP-based ransomware samples that we examined, we empirically observed that the most commonly used formats for these samples are the PE file, especially ∗.exe and ∗.dll; some PE files have compilation information in the file, and this information does not change with the migration of the programs. As a result, it is a good way to obtain the creator's information. A PE file mainly consists of five major components: DOS MZ header, DOS stub, PE header, section headers, and section content. Each component contains a great deal of information. There is very little information that we can use to identify the creator, and some identification information needs to be extracted from the content of each section. In this paper, there are many clues in the PE files that can be extracted to trace back the attacker: file name, PE file type, compiler version, compilation path, compile time, last modified time, last open time, IP address, URL, domain name, language, string, wide character, and so on. We extract most of the clues with *PEView* [34]. However, since it cannot directly obtain the compilation path, we use the *pefile* [35] tool to extract paths by locating in the PE structure.

Since the extraction clues include different encoding formats, to facilitate observation and the unified mode of subsequent analysis, the extraction system completely converts the data encoding obtained into the UTF-8 format and saved in the SQLite database. Before submitted to the analysis system, the extracted clued are divided into two categories: string clues and path clues.

5.2. Automatic Analysis. At this point, the clues extracted from this system mainly include string clues and path clues. The number of string clues is large, mixed with a large number of unidentifiable strings. Because the number of path clues is also very large with no semantic correlation, it is difficult to identify traceable clues manually. So we focus on how to automate the identification of traceable clues for path clues.

5.2.1. Users and Programs Analysis. In the analysis of the path clues, we first propose to obtain the attacker clues by identifying the features of the context-related segmentation on the same path. For instance, each user has a separate user folder, and it is located in the "Users" folder under Drive C. As a result, the system can obtain the user name at the attacking host by obtaining the folder name under the "Users" folder (e.g., *C:\Users\Dell\...*). The "Program files" folder usually contains the name of the software program installed on the machine (e.g., *C:\Program Files\Microsoft Visual Studio 11.0\...*). What is more, the QQ account number is always located in the "*\QQ\QQfile\*" folder (e.g., *D:\QQ\QQfile\86∗ ∗ ∗ ∗ ∗∗086\FileRecv\...*).

In this way, the analysis system can quickly and accurately get host names, email accounts, program names, social software numbers, and other traceable clues that are carried in the attacker's file paths. However, such user information for less of the overall clues is acquired by chance. Therefore, we will conduct further analysis on this basis.

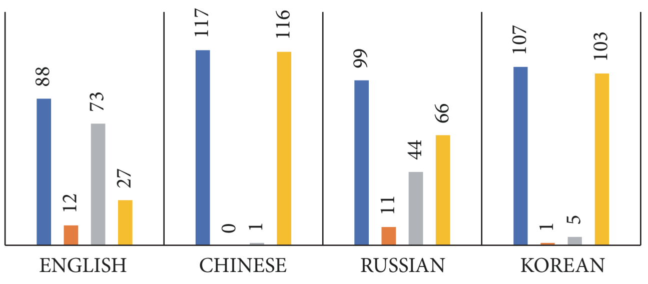
5.2.2. Account Analysis. Through the analysis of *APT1* and some other attribution reports, we find that the mapping between the attacker and the physical world identity can be better obtained by analyzing the account number left by the attacker. This information includes but is not limited to the location of the IP address, the spelling and registration of the domain name, the URL corresponding IP address, and the domain name of the mailbox account. Because it is difficult to identify this information effectively in a large number of strings and path clues, the analysis system automatically identifies the IP address, domain name, URL, and mailbox account by regular matching. Then, with the help of threat intelligence and big data technology, more relevant clues are obtained.

5.2.3. Language Identification. User languages often help to determine an attacker's idiomatic language, but because of a large number of languages in different countries and the high similarity of some languages, we use automatic analysis systems to identify the language of clues. We test the accuracy of two language identification toolkits using entire path information in four different languages. We have found "*langid.py*" toolkit to be overall more accurate than "*langdetect*" toolkit. The comparison results are shown in Figure 3. The *langid.py* is a language identification toolkit developed by *Lui* and *Baldwin* at the University of Melbourne [36]. It combines a naive Bayes classifier with cross-domain feature selection to provide domain-independent language identification.

5.2.4. Traceable Strings Identification. When traceable strings are needed, traditional string analysis methods usually use Named Entity Recognition (NER). However, the clues to be analyzed mainly include strings and path. Strings before and after the path separator have few semantic correlations. What is more, the string between the path separators and the remaining strings to be analyzed are mostly semantically unrelated due to their limited length. So this paper proposes an algorithm, which can quickly and automatically analyze the traceable strings in strings and path.

In order to filter out meaningful traceable clues related to the attacker's identity, the path clues and string clues are split into strings and identified by common words and gibberish in the following steps. Figure 4 shows the automatic traceable clues identification system process.

Make Stop Words. The system splits the path by the path delimiter, as these separated path strings that are common to multiple computers have no identifying effect. So, we take out the file string names that are common to 20 normal user



■ langid toolkit correct recognition
■ langdetect toolkit correct recognition
■ langid toolkit incorrect recognition
■ langdetect toolkit incorrect recognition

FIGURE 3: Accuracy of Two Language Identification Tools: In tests using the same known language path clues, the accuracy of the *langid* toolkit for English, Chinese, Russian and Korean was 88%, 100%, 99%, 99.0% respectively, while the accuracy of the langdet toolkit was 73%, 0.85%, 40%, and 4.60%, respectively.

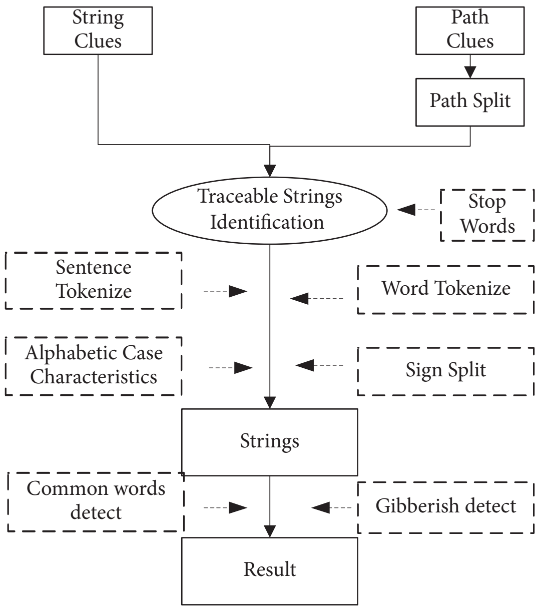


FIGURE 4: Process of automatic identifying of traceable clues.

computers as stop words. Then, it removes the stop words from the strings after each split.

Segmentation. After removing the stop words, we separate sentences and words for each segment of the path with tokenize (e.g., NLTK, NER) [37]. However, as the segments of the path are not semantically related, the segmentation is less effective. Based on our extensive research on path information, users often prefer to use symbols such as underscores to split file names. In addition, they also like to use capitalization in multiword strings to facilitate reading. As a result, the system uses the common identifier in the document and alphabetic case characteristic to segment the strings again.

Common Words Removal. After segmentation, the system obtains a lot of repeatable strings. But most of them are commonly used words and does not help in identifying ransomware attackers. As a result, we filter out the common words by comparing each string with a dictionary. If the string can be matched in a dictionary, the system marks the common word property of the word as false, otherwise, as true.

Gibberish Detect. Based on our analysis and observation, we found that there is a large amount of garbled information in path clues. Most of these strings are generated randomly, and people can recognize this randomness manually, but it is difficult for a program to recognize it automatically. As a result, we propose detecting the gibberish by training the Markov chain model with English texts. For each string X, there is a probability that the i character in X is

$$\begin{aligned} & \mathrm{P}\left\{X_{i+1}=x \mid X_1=x_1, X_2=x_2, \ldots, X_i=x_i\right\} \\ & \quad = P\left\{X_{i+1}=x \mid X_i=x_i\right\} \end{aligned} \tag{2}$$

Each letter is related to the upper and lower two letters. This relationship can be expressed by 2-gram. For example, with regard to the string "*Ransomware*": *Ra*, *an*, *ns*,. . .*e[space]*. The analysis system can record how often the characters appear next to each other, through the collection of gibberish and some commonly used normal phrases or vocabulary. It normalizes the counts, after reading through the training data and then measures the probability of generating the string based on the digest by multiplying the probability of the pair of adjacent characters in the string [38]. The training data can help statistics corresponding to gibberish and normal strings, respectively, the average transfer probability. This probability then measures the amount of possibilities assigned to this string according to the data as observed by the model. When we test the string "Ransom," it computes

$$\begin{aligned} prob\left['Ransom'\right] &= prob\left['R'\right]\left['a'\right] \times prob\left['a'\right]\left['n'\right] \\ & \quad \times \cdots \times prob\left['m'\right]\left[''\right] \end{aligned} \tag{3}$$

If the input string is gibberish, it will pass through some pairs with very low counts in the training phase and hence have low probability. The system then looks at the amount of probability per character for a few known normal strings and a few examples of known gibberish and then picks a threshold between the gibberish's most possibilities and the normal string's least possibilities:

$$\begin{aligned} & threshold \\ & \quad = \frac{\left(min\left(nomalprobs\right) + max\left(gibberishprobs\right)\right)}{2} \end{aligned} \tag{4}$$

When we analyze the string 'X', if $prob['X'] > threshold$, the analysis system will view the string as normal string. If $prob['X'] \leq threshold$, the analysis system will view the string as 'False.' Then, the system removes the strings with the 'False' type as gibberish.

Table 1: Traceable strings recognized result [13].

String	Common Words Recognized	Gibberish Recognized	Final Recognized
Wh∗ ∗ ∗∗ter	True	True	True
gandcrab	True	True	True
kate_z	True	True	True
vwrtjty	True	False	False
program	False	True	False

Identification Result. After the above analysis, the automated analysis system outputs the strings for which both the common word tag and the gibberish tag are True, which is the list of traceable clues [13]. Table 1 is reproduced from Z. H. Wang et al. (2018) [under the Creative Commons Attribution License/public domain].

When the string is a normal word, such as "program", the common words recognition will make it "False." Gibberish recognition can recognize the word only with non-randomly generated identifier strings. Moreover, it cannot identify the words that do not conform to the spelling patterns of common words (e.g., vwrtjty). The string is considered to have auxiliary traceability only if both of the analysis values are true. We rely extensively on string inversion to verify the accuracy of the system. It is found that most of the traceable strings on the volunteer storage can be recognized. Table 1 shows the recognition results of several typical strings.

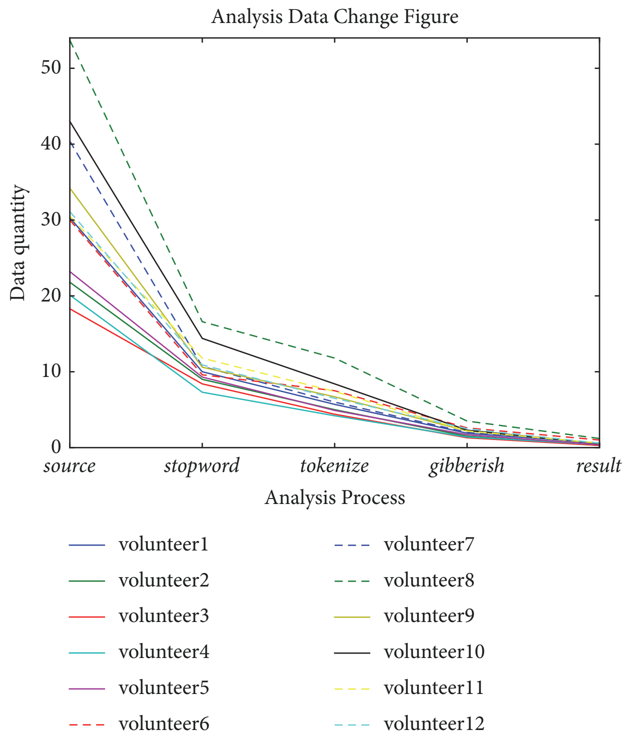


Figure 5: The data volume changes in 12 volunteers' traceable clues through the analysis system. The screening rate can reach 98.34% [13].

6. Evaluation

In this paper, we evaluate the proposed method with two experiments. The goal of the first experiment is to demonstrate that the proposed method can help trace back to the RDP-based ransomware attackers and capture their private information. The goal of the second experiment is to demonstrate that the method can also automatically recognize clues in ransomware samples and help trace back to ransomware makers.

6.1. RDP-Based Ransomware Attacker

6.1.1. Clue Capture and Analysis. We used Windows 7 system virtual machines to deploy the deception environment and assess the effectiveness of it. While Windows 7 is not required, it was chosen because of the wide range of applications and because it is one of the main targets of ransomware. We invited 122 professional volunteers to help with the experiment and provided them with an experimental fleet of 12 virtual hosts. Most of the volunteers' login information, clipboard content, shared folder path, and uploaded PE file were able to be successfully captured by the monitor system.

We choose 12 computers' path the information collected from volunteers and record the rate of convergence after each step of the analysis. Figure 5 shows how the number of traceable clues remains as each step of the data is processed by the analysis system [13]. Figure 5 is reproduced from Z. H. Wang et al. (2018) [under the Creative Commons Attribution License/public domain].

6.2. Traceback Attackers. Next, we carried out a detailed analysis of the information left over from one of the volunteers' attacks. By understanding the data from the automatic analysis of an attacker, we can infer that the attacker's real identity may have the following characteristics: (1) The attacker is likely to come from China. (2) The attacker may be a security and related personnel. (3) The attacker may work in the Chinese Academy of Sciences and have relations with a large security manufacturer in China. The conclusion is shown in Figure 6. It shows the results from the analysis system in five sections.

Username. Through automatic analysis, a total of three user names from the attacker's host were extracted from a large number of clues, in which, by using the "Dell" user, it can be assumed that the attacker was using the Dell host.

Programs. The automatic analysis system identified several typical software programs installed in the attacker's host. For instance, "QQ" is a widely used real-time social tool in China; "AliPay" is an online payment tool developed by Alibaba and widely used in China; "Sogou input" is an input method software for Chinese developed by China Sogou Company; MeiTu is a photo beautification software developed by a Chinese company and widely used in China; In addition, "Sinfor" and "360" are both well-known security manufacturers in China and have a large number of safety-related products; "Visual Studio" is a common development software. It is found that much Chinese-made software widely

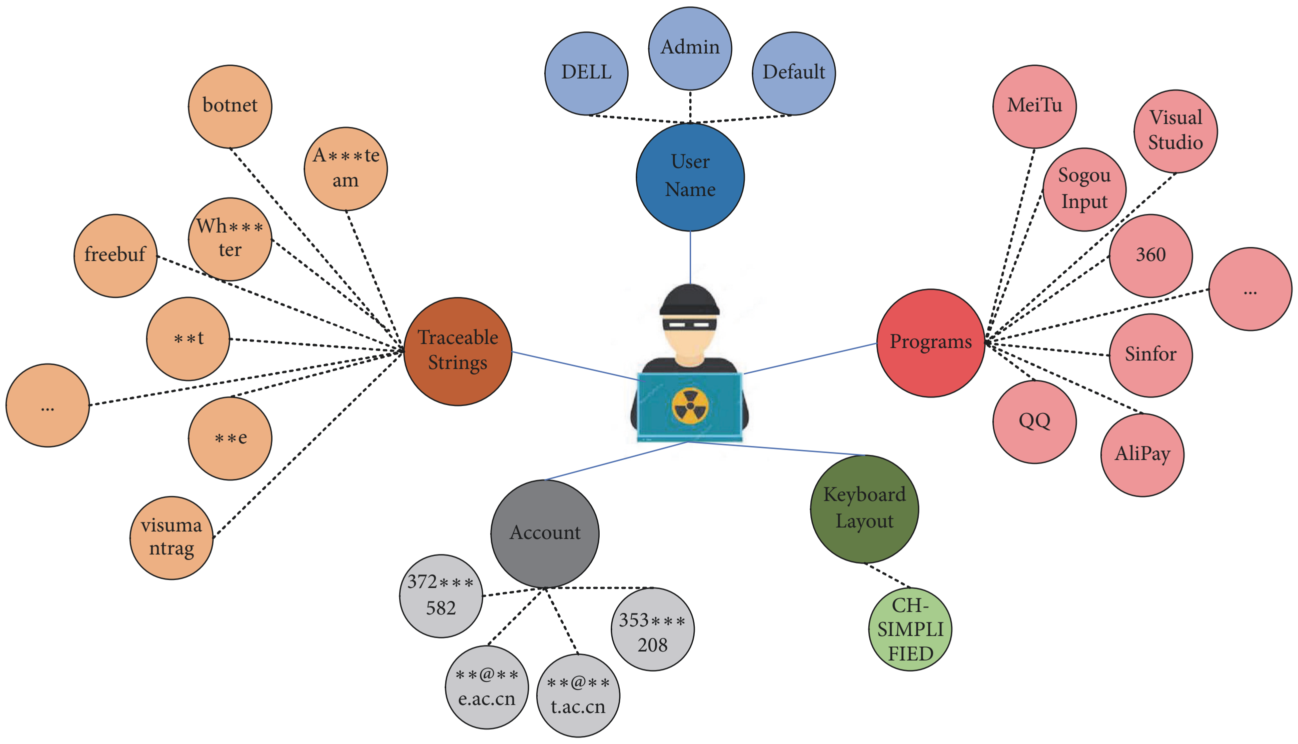


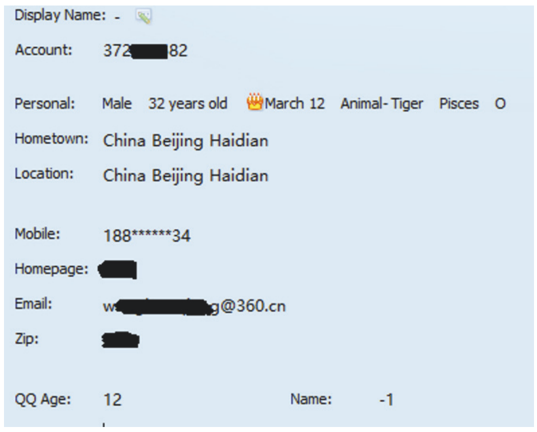
Figure 6: Analyze an attacker using automatic analysis system to display traceable information about the attacker in five aspects: host user name, program software, keyboard layout, account information, and traceable string.

used in China was installed on the attackers' hosts, as well as some development and security products that were installed less often by ordinary users.

Keyboard Layout. Through network communication analysis, we obtain the attacker's keyboard layout is Simplified Chinese, from which it can be inferred that the attacker's native language is probably Chinese. This is consistent with the host installed software features.

Account Number. According to the system automatically identified account number, it found two QQ accounts and two e-mail accounts. The e-mail accounts for the Chinese Academy of Sciences business accounts can confirm the above assumption that the attackers from China, and it is likely to work in the Chinese Academy of Sciences. Through the registration information of the QQ account (as shown in Figure 7), we can find the following information: (1) the attacker could be a male, aged 32; (2) Internet-related work, likely with *360 Security*; (3) located in *Haidian District, Beijing, China*. In the "353∗ ∗ ∗208" account registration information, mail, work, and other information are empty, nicknamed a special English string: "*Wh*∗ ∗ ∗*ter*" it can be presumed that the account is likely to be a private use.

Traceable Strings. The automatic analysis system sifts strings from the monitor that may be traceable. "*Freebuf*" is a security information exchange website commonly used by Chinese security personnel. "*Bootnet*" is a secondary attack method that is often used by attackers, and is often used by



Display Name: -
Account: 372[redacted]82
Personal: Male 32 years old March 12 Animal- Tiger Pisces O
Hometown: China Beijing Haidian
Location: China Beijing Haidian
Mobile: 188******34
Homepage: [redacted]
Email: w[redacted]g@360.cn
Zip: [redacted]
QQ Age: 12 Name: -1

Figure 7: The registration infographic of QQ account 372∗ ∗ ∗82.

security researchers as the main research direction. "∗∗*e*" and "∗∗*t*" are acronyms for departments under the Chinese Academy of Sciences, while "*A*∗ ∗ ∗*team*" is a security research team in the "∗∗*e*" department. "*Visumantrag*" is often used when processing visas from various countries. The "*Wh*∗ ∗ ∗*ter*" matches the nickname of the QQ number "*353*∗∗∗*208*" account and is likely to be its regular nickname.

6.3. Ransomware Maker Automatic Analysis. In order to verify that the analysis system is available in tracing back the ransomware maker, we obtained more than 8000 ransomware

Table 2: Same identifier for different samples [13].

Sample Hash	Same Identity Strings
68dd613973f8a∗ ∗ ∗7befe0f4b461d258ce569b9020079∗ ∗ ∗9ae0f090266a60cc	jqxxppho/aonk/je7z2
fa0c084aef41969∗ ∗ ∗1f427a1b65e1b1464fa82d297e712∗ ∗ ∗e1fdf312c04481	
4a36da0286d42cd∗ ∗ ∗306fba99239c8238821beaf40744be∗ ∗ ∗ba862dbc6d90f	
70cbc839bcb88∗ ∗ ∗7422bbd2fa812dd3de02391b6a9f1∗ ∗ ∗0762a6f49cf32501	

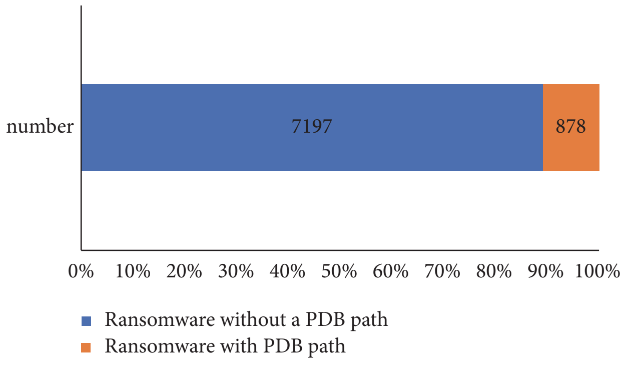


Figure 8: Of the 8,075 real ransomware samples, 11% were able to extract the Program Data Base (PDB) path, while the remaining 89% of authors may have chosen to hide the PDB path during compilation.

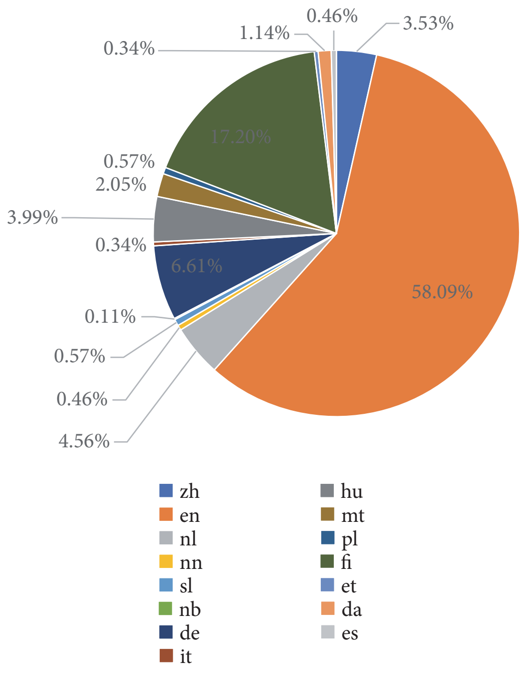


Figure 9: Of the 878 ransomware sample PDB paths, English accounted for 510, and the remaining monitored languages were Chinese: 31(3.53%), Dutch: 40(4.56%), Norwegian Nynorsk: 4(0.46%), Slovenian: 5(0.57%), Norwegian Bokmål: 1(0.11%), German: 58(6.61%), Italian: 3(0.34%) Hungarian: 35(3.99%), Maltese: 18(2.05%), Polish: 5(0.57%), Finnish: 151(17.20%) Luxembourgish: 3(0.34%), Danish: 10(1.14%), and Spanish: 4(0.46%).

samples from *VirusTotal* [39] and extracted the Program Data Base (PDB) path from the samples (as shown in Figure 8). It is shows that, despite the large number of attackers' deliberate erasure of the compiled information, legacy PDB path information can still be found from a large number of ransomware samples.

What is more, the analysis system is used to automatically analyze more than 800 different ransomware samples' PDB path. We found multiple identity strings in the analysis results of different samples. One of the typical results is shown in Table 2 [13]. Table 2 is reproduced from Z. H. Wang et al. (2018), [under the Creative Commons Attribution License/public domain]. We used *VirusTotal* to validate the samples and found that they were all Cobra family ransomware. As a result, it could be assumed that these samples are made by the same ransomware maker. When one sample is traced back, other samples can also arrive at the conclusion of the ransomware maker. When different traceable information is analyzed from different samples, the identity information of the ransomware maker can be described by integrating the information.

Language Identification. When we carry on the language recognition to the PDB path of 878 samples, the result is as Figure 9. The system automatically identifies the path which mainly contains 11 languages, of which English accounts for the largest proportion.

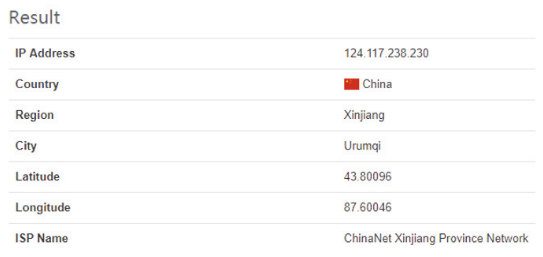
Actual Case Analysis. The automatic identification system helped us identify the traceable strings in the PDB path. Take the example of a sample PDB path (the Chinese has been translated into English) with an MD5 value of "60c6a92afb∗ ∗ ∗6d0c16f7".

"*E:\\LIAO\\STUDIO\\UAC\\simulated an improved version of the 360 anti-virus program 1117 Bale 1∗4.1∗∗.2∗∗.∗∗0(∗∗1)_9818 program*\\WriteSystem32\\Release\\VirtualDesktop.pdb"

We can infer from the automatic analysis result that the ransomware maker is likely from China. We can prove this point by the following:

(1) Language Identification: This is done by identifying the path language, which can be used to speculate that the attacker may have come from China. By understanding the Chinese strings in the path, we find that the meaning of this text is to improve to bypass the detection of 360, security software that has a large market in China.

(2) IP address: The system identified an IP address in the PDB path. With the help of threat intelligence database, it is found that the IP came from *Xinjiang, China* (Figure 10).


Result

IP Address	124.117.238.230
Country	🟥 China
Region	Xinjiang
City	Urumqi
Latitude	43.80096
Longitude	87.60046
ISP Name	ChinaNet Xinjiang Province Network

Figure 10: Partial infographic of IP: 1∗4.1∗∗.2∗∗.∗∗0 in the threat intelligence library.

(3) Traceable Strings: The extracted identifier string "*LIAO*" resembles a Chinese pinyin and is most likely a Chinese surname. To sum up, we speculate that the ransomware maker is most likely from China.

7. Conclusion

In this paper, we focus on tracing back RDP-based ransomware. Recently, more and more ransomware attackers are using RDP attacks to spread ransomware with impunity due to the use of strong cryptography. We propose tracing back the attack sources to deter RDP-based ransomware with the proposed deception environment. It collects traceable clues and performs automatic analysis by using natural language processing and machine learning techniques. The evaluation shows that it is able to trap attackers and collect traceable clues left by attackers in a deception environment. With automatic clue identification, it can converge the amount of traceable clues to about 2%. We use this method to analyze two practical cases and show its effectiveness. By tracing back to ransomware attackers, we can provide a strong deterrent to stifle the development of ransomware.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Additional Points

This manuscript provides additional deception environment monitors and analytical methods by using more volunteers and samples. It proves the validity of the methods through specific traceback cases (i.e., traceback the ransomware attacker and traceback the ransomware maker).

Disclosure

An earlier version of this paper was presented at the International Conference: IEEE Third International Conference on Data Science in Cyberspace, Guangzhou, China, 18-21 June 2018. The authors' initial conference paper focused mainly on the effectiveness of deception environment monitors and the screening rate of the analysis system.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The authors would like to thank VirusTotal for providing ransomware samples and the volunteers from the University of Chinese Academy of Sciences. This work is supported in part by the National Key research and Development Plan (Grant no. 2018YFB0803504) and the National Natural Science Foundation of China (61871140, 61572153, U1636215, 61572492, and 61672020).

References

[1] A. Kharraz, "UNVEIL: A Large-Scale, Automated Approach to Detecting Ransomware," in *Proceedings of the 25th USENIX Security Symposium (USENIX Security 16)*, pp. 757–772, USENIX Association, 2016.

[2] K. Savage, P. Coogan, and H. Lau, "The evolution of ransomware," Tech. Rep., 2015.

[3] A. Kharraz, W. Robertson, D. Balzarotti, L. Bilge, and E. Kirda, "Cutting the Gordian Knot: A Look Under the Hood of Ransomware Attacks," in *Detection of Intrusions and Malware, and Vulnerability Assessment*, vol. 9148 of *Lecture Notes in Computer Science*, pp. 3–24, Springer International Publishing, Cham, 2015.

[4] S. Mohurle and M. Patil, "A brief study of Wannacry Threat: Ransomware Attack 2017," *International Journal of Advanced Research in Computer Science*, vol. 8, no. 5, 2017.

[5] X. Yu, Z. Tian, J. Qiu, and F. Jiang, "A Data Leakage Prevention Method Based on the Reduction of Confidential and Context Terms for Smart Mobile Devices," *Wireless Communications and Mobile Computing*, vol. 2018, Article ID 5823439, 11 pages, 2018.

[6] J. Yaneza, "Brute Force RDP Attacks Plant CRYSIS Ransomware," https://blog.trendmicro.com/trendlabs-security-intelligence/brute-force-rdp-attacks-plant-crysis-ransomware/.

[7] "10% of Ransomware Attacks on SMBs Targeted IoT Devices," https://www.darkreading.com/application-security/10–of-ransomware-attacks-on-smbs-targeted-iot-devices-/d/d-id/1329817.

[8] Q. Tan, Y. Gao, J. Shi, X. Wang, B. Fang, and Z. H. Tian, "Towards a Comprehensive Insight into the Eclipse Attacks of Tor Hidden Services," *IEEE Internet of Things Journal*, 2018.

[9] "Kaspersky Security Bulletin: STORY OF THE YEAR 2017," 2017, https://media.kasperskycontenthub.com/wp-content/uploads/sites/43/2018/03/07164824/KSB_Story_of_the_Year_Ransomware_FINAL_eng.pdf.

[10] Z. Tian, Y. Cui, L. An et al., "A Real-Time Correlation of Host-Level Events in Cyber Range Service for Smart Campus," *IEEE Access*, vol. 6, pp. 35355–35364, 2018.

[11] R. Ross et al., *Managing Information Security Risk: Organisation, Mission, and Information System View*, National Institute of Standards and Technology, 2011, http://csrc.nist.gov/publications/nistpubs/800-39/SP800-39-final.pdf.

[12] Y. Wang, Z. Tian, H. Zhang, S. Su, and W. Shi, "A Privacy Preserving Scheme for Nearest Neighbor Query," *Sensors*, vol. 18, no. 8, p. 2440, 2018.

[13] Z. H. Wang, X. Wu, C. G. Liu, Q. X. Liu, and J. L. Zhang, "RansomTracer: Exploiting Cyber Deception for Ransomware Tracing," in *Proceedings of the IEEE Third International Conference on Data Science in Cyberspace*, pp. 227–234, 2018.

[14] B. A. S. Al-rimy, M. A. Maarof, and S. Z. M. Shaid, "Ransomware threat success factors, taxonomy, and countermeasures: A

survey and research directions," *Computers & Security*, vol. 74, pp. 144–166, 2018.

[15] F. Jiang, Y. Fu, B. B. Gupta et al., "Deep Learning based Multi-channel intelligent attack detection for Data Security," *IEEE Transactions on Sustainable Computing*, 2018.

[16] N. Andronio, S. Zanero, and F. Maggi, "HelDroid: dissecting and detecting mobile ransomware," in *Research in Attacks, Intrusions, and Defenses*, vol. 9404 of *Lecture Notes in Computer Science*, pp. 382–404, Springer, 2015.

[17] F. Mercaldo, V. Nardone, A. Santone, and C. A. Visaggio, "Ransomware Steals Your Phone. Formal Methods Rescue It," in *Formal Techniques For Distributed Objects, Components, And Systems: 36th IFIPWG 6.1 International Conference, FORTE 2016, held as part of the 11th International Federated Conference On Distributed Computing Techniques, DisCoTec 2016*, E. Albert and I. Lanese, Eds., pp. 212–221, Springer International Publishing, 2016.

[18] D. Sgandurra, L. Mu±oz-Gonzßlez, R. Mohsen, and E. C. Lupu, "Automated Dynamic Analysis of Ransomware: Benefits, Limitations and use for Detection," https://arxiv.org/abs/1609.03020.

[19] S. Song, B. Kim, and S. Lee, "The Effective Ransomware Prevention Technique Using Process Monitoring on Android Platform," *Mobile Information Systems*, vol. 2016, Article ID 2946735, 9 pages, 2016.

[20] T. Yang, Y. Yang, K. Qian, D. C.-T. Lo, Y. Qian, and L. Tao, "Automated detection and analysis for android ransomware," in *Proceedings of the 17th IEEE International Conference on High Performance Computing and Communications, IEEE 7th International Symposium on Cyberspace Safety and Security and IEEE 12th International Conference on Embedded Software and Systems, HPCC-ICESS-CSS 2015*, pp. 1338–1343, USA, August 2015.

[21] N. Scaife, H. Carter, P. Traynor, and K. R. B. Butler, "CryptoLock (and Drop It): Stopping Ransomware Attacks on User Data," in *Proceedings of the 36th IEEE International Conference on Distributed Computing Systems, ICDCS 2016*, pp. 303–312, Japan, June 2016.

[22] M. M. Ahmadian and H. R. Shahriari, "2entFOX: A framework for high survivable ransomwares detection," in *Proceedings of the 13th International ISC Conference on Information Security and Cryptology, ISCISC 2016*, pp. 79–84, Iran, September 2016.

[23] M. M. Ahmadian, H. R. Shahriari, and S. M. Ghaffarian, "Connection-monitor & connection-breaker: A novel approach for prevention and detection of high survivable ransomwares," in *Proceedings of the 12th International ISC Conference on Information Security and Cryptology, ISCISC 2015*, pp. 79–84, Iran, September 2015.

[24] D. Kim, W. Soh, and S. Kim, "Design of Quantification Model for Prevent of Cryptolocker," *Indian Journal of Science and Technology*, vol. 8, no. 19, 2015.

[25] C. Moore, "Detecting Ransomware with Honeypot Techniques," in *Proceedings of the 2016 Cybersecurity and Cyberforensics Conference (CCC)*, pp. 77–81, Amman, Jordan, August 2016.

[26] F. Mbol, J. Robert, and A. Sadighian, "An Efficient Approach to Detect TorrentLocker Ransomware in Computer Systems," in *Cryptology and Network Security*, S. Foresti and G. Persiano, Eds., vol. 10052 of *Lecture Notes in Computer Science*, pp. 532–541, Springer International Publishing, Cham, 2016.

[27] K. Cabaj, P. Gawkowski, K. Grochowski, and D. Osojca, "Network activity analysis of CryptoWall ransomware," *Przegląd Elektrotechniczny*, vol. 91, no. 11, pp. 201–204, 2015.

[28] J. Chen, Z. Tian, X. Cui, L. Yin, and X. Wang, "Trust Architecture and Reputation Evaluation for Internet of Things," *Journal of Ambient Intelligence & Humanized Computing*, vol. 2, pp. 1–9, 2018.

[29] C. Le Guernic and A. Legay, "Ransomware and the Legacy Crypto API. Paper presented at the Risks and," in *Risks and Security of Internet and Systems: 11th International Conference, CRiSIS 2016*, Roscoff, France, 2017.

[30] L. Pingree, *Emerging Technology Analysis: Deception Techniques and Technologies Create Security Technology Business Opportunities*, Gartner, 2015.

[31] "Ransomware and Businesses," https://www.symantec.com/content/dam/symantec/docs/security-center/white-papers/ransomware-and-businesses-16-en.pdf, 2016.

[32] A. Kharraz and E. Kirda, "Redemption: Real-Time Protection Against Ransomware at End-Hosts," in *Proceedings of the International Symposium on Research in Attacks, Intrusions, and Defenses*, pp. 98–119, 2017.

[33] F. Bergstrand, J. Bergstrand, and H. Gunnarsson, Localization of Spyware in Windows Environments,.

[34] "PEView," https://www.aldeid.com/wiki/PEView.

[35] "python-pefile," https://pypi.python.org/pypi/pefile.

[36] M. Lui and T. Baldwin, "langid., py, An off-the-shelf language identification tool," in *Proceedings of the ACL 2012 system demonstrations. Association for Computational Linguistics*, pp. 25–30, 2012.

[37] S. Bird and E. Loper, "NLTK: the natural language toolkit," in *Proceedings of the ACL 2004 on Interactive poster and demonstration sessions. Association for Computational Linguistics*, Barcelona, Spain, July 2004.

[38] "Rrenaud. Gibberish-Detector," https://github.com/rrenaud/Gibberish-Detector/blob/master/README.rst.

[39] "VirusTotal," http://virustotal.com.