Stealthy Video Capturer: A New Video-based Spyware in 3G Smartphones

Nan Xu[1], Fan Zhang[2 1], Yisha Luo[1], Weijia Jia[1], Dong Xuan[3] and Jin Teng[1]
Dept. of Computer Science, City University of Hong Kong, Hong Kong SAR, China[1]
Dept. of Electronics & Information Engineering, Huazhong University of Sci. & Tech., Wuhan, China[2]
Dept. of Computer Science and Engineering, The Ohio State University, USA[3]
nanxu2@student.cityu.edu.hk, {fanzhang, yishaluo, wei.jia, jinteng}@cityu.edu.hk
xuan@cse.ohio-state.edu

ABSTRACT

In this paper, we investigate video-based vulnerabilities in 3G Smartphones. Particularly, we design a new video-based spyware, called Stealthy Video Capturer (SVC). SVC can secretly record video information for the third party, greatly compromising Smartphone users' privacy. We implement the spyware and conduct extensive experiments on real world 3G Smartphones. Our experimental results show that the spyware can capture private video information with unremarkable power consumption, CPU and memory occupancy, hence being stealthy to Smartphone users. Moreover, SVC can naturally be resistant to almost all commercial anti-virus tools, like McAfee, Kaspersky and F-Secure mobile version. To the best of our knowledge, our work is the first one to address video-based vulnerabilities in 3G Smartphones. We expect our work will prompt serious attentions on this issue.

Categories and Subject Descriptors

C.2.0 [**Information Systems Applications**]: General—*Security and Protection*

General Terms

Design, Security

Keywords

Security, 3G Smartphones, Spyware, Privacy

1. INTRODUCTION

Currently Smartphones are widely used and the number of Smartphones in use globally has expanded dramatically in recent years. A report from Canalys "Worldwide smart mobile device market, Canalys Q4 2007" [1] shows that Smartphone shipments hit 118 million in 2007, up 53% in 2006. Industry analysts predict that more than 200 million Smartphones will be sold in 2009 [2]. As mobile telephony networks are gradually shifting to 3G (or beyond), 3G Smartphones are getting more and more popular. Almost all manufacturers have developed models that are equipped with 3G connections, such as Apple iPhone, HTC Diamond and Nokia N96. So it's a general trend that 3G Smartphones will enjoy a fairly large market share in the near future.

1.1 Motivations

As 3G Smartphones have made their ways into our everyday lives, they virtually bear witness to some most private parts of our lives. A malicious hacker can tap this kind of information by taking control over our intimate friends, 3G Smartphones. Just imagine that anybody-from a stranger in the bedroom to a competitor in the boardroom-can 'peep' through a 3G Smartphone on a person's life at any time or take over one's private information, no matter whether it be a classified document, a phone call or even a real-time video clip!

Nowadays compromising one's privacy through a 3G Smartphone is becoming a reality. Since 3G Smartphones run identifiable operating systems, like Windows Mobile, Symbian and RIM etc., they are actually like diminutive computers with restricted resources and functions. To date too many attacks against traditional computers have been recorded, so it's not unreasonable to assume that 3G Smartphones will somehow yield and confide to a malicious invading hacker. Thanks to many latent security vulnerabilities existing in 3G Smartphones, entryways are ready there, inviting hackers to install malwares or other undesirable scripts. For example, Dubbed Cabir [26] is a classic proof-of-concept malware in 3G Smartphones. Though Dubbed Cabir doesn't cause any damage to an infected device, yet it exhausts the phone's battery as the malware will copy itself to another one through an opened Bluetooth connection. Evidently our protections of 3G Smartphones are far from enough.

To make matters worse, more and more private information is entrusted to our friend, the 3G Smartphone, which is getting more and more powerful in performance and diversified in functionality. Besides phone numbers, contact lists, documents in varied formats etc., we even turn it into an online video-recorder, seeing that almost every 3G Smartphone is equipped with a camera, and the wireless options offered by 3G Smartphones, such as 3G networks, Bluetooth (BT), WiFi or IrDA, are good enough to handle certain types of video transmission. So once 3G Smartphones defect, some horrific video clips may be easily captured and hence disclosed, resulting in great privacy breach and realization of the aforementioned apprehension of 'being peeped'.

1.2 Our Contributions

In this paper, we investigate video-based vulnerabilities in 3G Smartphones through the design of a new video-based spyware, called Stealthy Video Capturer (SVC)[1]. It is a form of spyware that allows hackers to automatically activate the built-in camera on 3G Smartphones. SVC performs just like a stealthy ghost hiding in ones's 3G Smartphone which will capture its owner's behaviors at any time. As a result, the trusted phone can become a potential "traitor" by turning you in, for example, to be a leading actor on the Candid Web video. We hereby claim the following contributions:

- To the best of our knowledge, we are the first team that designs and implements such a video-based spyware in the real world 3G Smartphones.
- A series of algorithms and mechanisms are designed to promote SVC's stealthiness. For instance, we carefully evaluate main types of infection methods to find a most covert and failsafe one; a set of triggering schemes for recording and sending are also designed to avoid significant impact on normal operations of 3G Smartphones, as well as improve SVC's efficiency in information collection.
- Extensive evaluations are carried out on the real 3G Smartphone test beds. The stealthiness and performance of SVC are comprehensively studied here.

It's also worth mentioning that there's a double-fold nuance in the successful development of the so-called SVC. Apart from encroachment upon one's privacy, SVC can also help the police to monitor the criminal suspects. Thus in this study, we will take the viewpoint from both hackers and investigators. Therefore, in the rest of the paper, the party that initiates SVC is neutrally called the SVC intender and the normal user who uses the Smartphone is called the Smartphone owner or the device owner interchangeably.

The remainder of this paper is organized as follows. In Section 2 we give an overview of SVC. Section 3, 4 and 5 present video capture and sending, SVC triggering algorithm and infection method in detail. Section 6 shows extensive experiment results based on the performance of SVC, and evaluates the performance of SVC. We discuss implications and extensions of our work in Section 7. We present related work in Section 8, and conclude in Section 9.

2. SVC OVERVIEW

2.1 Challenges

To realize SVC as a video-based spyware running in 3G Smartphones, three phases must be given due consideration respectively: 1. Stealthily install SVC into 3G Smartphones; 2. Collect the video information from 3G Smartphones; 3. Send the video file to the SVC intender. Since these phases are organically and temporally linked, a comprehensive design should be given to enable the development of SVC.

[1]We can provide the basic version for legitimate usage requests through Email.

There are grand challenges for the SVC realizations. In the first phase of infection and installation, we'll address the problem of disguise and cheating, in order to convince the device owner that our SVC, which is in fact a Trojan horse, is not harmful at all. In the second phase, we need to find alternative means to control Smartphone cameras, since Windows Mobile's own APIs are not well tailored to our needs, for example, native APIs will generate large video files which can be easily discovered and such settings as format selection, data compression and video storage are not tractable through native APIs. Meanwhile, a set of triggering mechanism for camera operation is also a must to enhance SVC's stealthiness and efficiency. Timing and triggering are also vital challenges for the third phase of file sending, since bad sending timing may cause observable side-effects.

Moreover, the issue of stealthiness remains the prime concern throughout our design. 3G Smartphones generally have limited resources such as CPU, memory, and battery power. In particular, dramatic power consumption is the most critical feature that could cause user's attention. In order to achieve desirable stealthiness, many factors should be taken into consideration, such as CPU load, memory usage and power consumption, which are interlocking and cannot be solved separately. Thus, how to do everything behind the stage without users' notice stands in need of systematic study. In this light, a series of mechanisms and algorithms are proposed and implemented. The detailed design will be given in Section 3, 4, 5.

Only stealthy running is not enough for SVC to fulfill its functions efficiently. Another challenge in the strategy is when to grab the intended video information. To solve this problem, SVC should recognize the useful information and decide when to execute video capturing. For example, a 3G Smartphone in the device owner's pocket or briefcase or during sleeping time may not have useful information to be captured. However, because of variance of environmental changes, the issue may have many open questions which are further described in Section 4.

2.2 SVC Architecture and Working Flow

SVC requires a tight control over the Smartphone's camera and should run at the right time according to the inside and outside environments. The system architecture of SVC is depicted in Figure 1. Based on the Windows Mobile 5.0/6.0 platform, SVC employs the relevant APIs to control the hardware, such as the camera, the system monitor and wireless connections. In the application layer, SVC is composed of three parts, namely *Video Capture*, *Triggering Algorithm* and *File sending*. We elaborate on their roles and implementation details below.

The three main parts of SVC realize different functions. *Video Capture module* takes charge of the camera. *File sending module* sends the video data to remote SVC intenders by practical modes, such as Email and MMS. *Triggering Algorithm* is the control module which determines the right time to turn on the video capture process or send the video information. The *Triggering Algorithm* is described in Section 4.

To give a big picture of SVC, we describe the working flows of *Video Capture* and *File sending process*.

When *Triggering Algorithm* provides the message to exe-

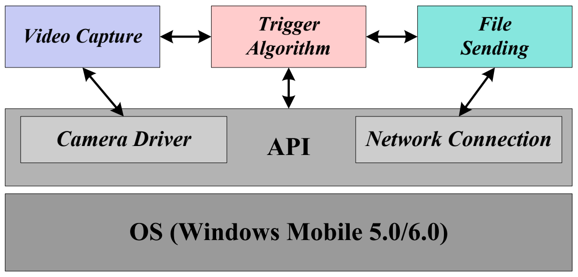


Figure 1: System Architecture

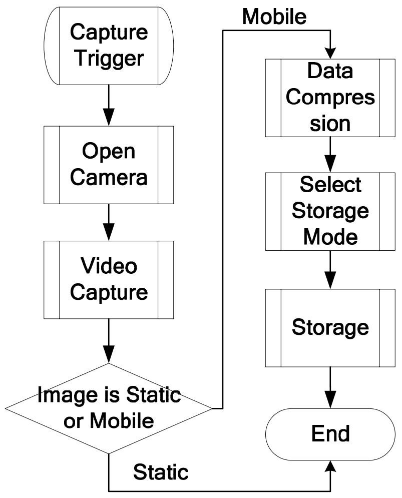


Figure 2: Working flow of *Video Capture Process*

cute the video capture module, the working flow of *Video Capture module* is triggered which is shown in Figure 2. First, the camera is opened using DirectShow filter to get the images. If the continuous images are static, the *Video Capture process* will terminate. Otherwise, the video frames are compressed using specific coding method, e.g. H.263. Using this strategy to determine whether to continue the video capture, we can acquire information stealthily and efficiently.

It is important to make economical use of memory in a mobile device. To reduce the size of the storage, data compression is necessary. SVC can select storage mode, such as storing the data in one file or several files. The video data files are hidden in the disk, waiting to be sent to the SVC intender.

Sending Process is described in Figure 3. When the sending trigger wakens the *Sending Process*, the hidden data files are assembled and sent by one communication mode.

3. VIDEO CAPTURE AND SENDING

The main function of SVC is to capture the device owner's privacy and send the useful information to a remote SVC intender.

Our key idea in implementation is to make device owners unaware of the whole process. SVC should be smart enough to hide itself and finish the full process stealthily. Then there are three factors which should be considered for its stealthiness:

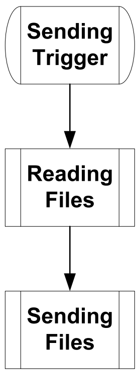


Figure 3: Working flow of *File Sending Process*

- No trace of execution on the Smartphone screen;
- Exert some control over the video recording process, e.g. when we get nonvisible images or static images, the program will intelligently turn itself off, avoiding much consumption of CPU and power to reduce the possibility of being detected by the Smartphone owners.
- Control over the storage of video files. First, the secretly recorded video cannot be in the standard 3G Smartphone format, otherwise some device owners may easily check the video and realize themselves to be peeped. Second, the real time data compression should be employed so as to reduce the size of the recorded file from being suspected. Third, we should carefully place the video files in some folders, i.e., the hidden file folders, normally not frequented by the device owners. Video files can be a whole chunk or distributed in several places. Many small hidden files are less likely to be detected. The implementation will take the three issues into consideration as shown below.

3.1 Windows Hiding

It is a surprise when the device owner found that the camera of the 3G Smartphone is in operation and recording something without the device owner's intention when the windows of SVC are shown. Thus windows hiding is a primary function to make users unaware that the SVC is running.

To hide the windows, we use the hidden attributes of Windows Mobile Function *ShowWindow()* , for example, *ShowWindow(hWnd, SW_HIDE)*. This method is easy to implement and rarely discovered by normal device owners. However, *ShowWindow()* can influence the display states of the screen, e.g. if device owner has chosen the backlight off mode, *ShowWindow()* can force the backlight to be on. But SVC is not supposed to change the display states of the screen. To maintain the system state as the previous, we employ the WinCE API function *GetSystemPowerState()* to know the system states. When SVC begins running, it first detects the status of the Smartphone's screen. If the screenlight is on, SVC can directly hide the windows. Otherwise, SVC can turn off the screen and backlight at the same time for the stealthiness of SVC.

3.2 DirectShow Controls Design

A number of Windows Mobile 5.0 APIs (for example, *SHCameraCapture*) make it facile for a mobile application developer to access a camera, but their ease of use comes at a price of "flexibility". Most of the time, using the Camera APIs directly can offer a solution, but SVC needs more controllability and flexibility. Therefore, we exploit DirectShow [28] in another way to access a camera and to build a filter manually for handling the graph events in the application message handler. As the camera is the hardware component, for an application to control the camera, it needs to collaborate with the camera's drivers. Consequently, the video capture filter can enable an application to capture the videos.

If we employ the native API directly, video would be encoded using *WMV9EncMediaObject* by default, a DirectX Media Object (DMO). In this case, we can only use a DMO inside a filter graph with the help of a DMO Wrapper filter. Then, the encoded video data needs to be multiplexed and a Windows Media ASF writer filter is used for this task. The ASF writer multiplexes the video data and writes it into an .asf file. The whole process is shown in the following diagram in black. The affiliated API is limited in several respects:

- Uninterruptable and uncontrollable. Only the recording time can be set while the number of frames to record can not be controlled.

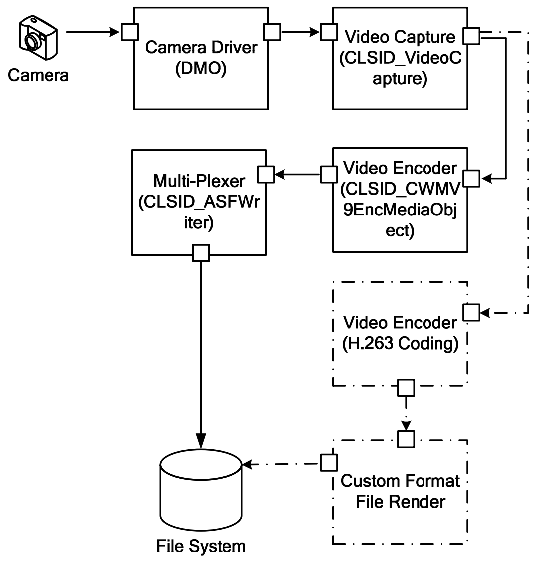
- Recordings can only be saved in one format, making it more easily detected.

- Incapable of compression and hence space-consuming.

As a result, we have to develop several COM controllers to enable modification of encoding and storage format. These controllers must be registered before execution. They must implement all functions required by a standard COM controller. In this light we insert two modules, namely Video Encoder and Custom Format File Render. The former module enables video compression through H.263, while the latter enables the device intender to save the file in a specified format which cannot be read by the device owner. The advantages of our approach, which is shown in Figure 4 in broken line are more flexible encoding methods, more covert data storage formats.

3.3 Data Compressing

Video capture process records video frames which must be compressed. Otherwise, the large video files are easy to be detected and difficult to be sent. Generally, the data compressing is based on the hardware driver of the built-in camera in 3G Smartphones. However, the hardware drivers of many 3G Smartphones are not open for programmer to use. Then we can not achieve the data compressing by hardware and only use software coding to compress the data. This method will take more CPU occupancy for computing. We need to select a coding scheme which can balance the CPU occupancy and data compression ratio. In our implementation, we use H.263 to do this. H.263 is a widely used data compression method for 3G video codec. It has a simple coding scheme and good real-time features. Compared with H.264, it has relatively lower data compression ratio but consumes less CPU cycles at the same time.



Camera
Camera Driver (DMO)
Video Capture (CLSID_VideoCapture)
Multi-Plexer (CLSID_ASFWriter)
Video Encoder (CLSID_CWMV9EncMediaObject)
Video Encoder (H.263 Coding)
Custom Format File Render
File System

Figure 4: The components in the filter graph used to capture video

3.4 Sending

File sending is the last step to be completed in its whole stealthy process. We can approach this issue from the specific features of SVC. For SVC, several features in file sending are most desirable. First, we wish that all wireless connections, like WiFi, Bluetooth, 3G networks or Internet, can be used to send the video file, and we are not just limited to a subset of those wireless options. Second, the real-time feature is not mandatory, since certain amount of delay is acceptable, say several hours. Third, the transmission protocol should be as simple as possible to facilitate our sending. The fewer restrictions, the better. Fourth, we favor transmissions free of charge, since it's cheaper and better in hiding our trace. Though MMS is designed to send multimedia data, yet it's normally not free of charge and too inflexible in size. FTP performs well for large file transmission. However, our video data is generally not large enough to be comfortable with FTP. Streaming excels in real-time transmission, but it requires nearly uninterrupted connection with streaming server, which is not quite possible for a mobile device, and the real-time feature is not so important under many cases. In sum, we prefer Email to other possible methods, such as MMS, FTP or Streaming, since Email is ubiquitously supported, free of charge and flexible in syntax and size.

Though it's just a routine procedure, yet some factors should be carefully handled. First, since the memory of 3G Smartphones is very limited, we cannot allocate too much memory, say a matrix of 10 thousand bytes, at a time. So the video file, which is normally very large, should be fragmented first and then sent separately. Second, a series of function calls should be made in the right order to take care of the handling of recipient list, the establishment of message stream and the attachment of the video file. Here the first factor should be taken into consideration to avoid operation failure.

To begin with, SVC will have the mail system initialized. After login onto the message store, SVC finds the current store entry and gets the Drafts entry, which is necessary for composing outgoing mail. Then a new message is created with all relative properties correctly set, especially the attachment of the video file. Finally, SVC sends out the mail and finishes all the cleaning up and resource recycling.

4. TRIGGERING ALGORITHMS

Triggering Algorithm is designed to determine when to turn on the video capture process and send the captured video. It includes *Capture triggering algorithm* which controls the video capture process and *Sending triggering algorithm* which controls the video sending process. To make the SVC as smart as possible, many practical scenarios should be considered. The purpose of the *Triggering Algorithm* is to make SVC stealthier and get more useful information. From these two sides, we give the strategy of how to design the smart *Triggering Algorithm.*

Practical applications dictate which is more important, stealthiness or information acquisition. These two concerns lead to different strategies in the timing and duration of turning on the cameras. Here we assume that SVC can be used in the following three scenarios:

- The first scenario is tracking, i.e. monitoring a certain device owner. This application exacts high stealthiness, which means that the device owner should have no knowledge of being monitored. In this scenario, the camera may be required to be turned on frequently. For stealthy sake, SVC should dynamically set the active time every time the camera is turned on. Seeing that some real-time feature is much desirable for our surveillance, the recorded data should be sent to the hacker as soon as possible.
- The second scenario is related with political or business espionage. Here information acquisition is of prime importance, such as what happens on evidence collection for a crime scene, or attendance and contracts signed at an important conference. It's these crucial information that the hacker focuses on. SVC is assigned the mission to acquire and send these information, so all job can be done once and for all. Under this scenario, data acquisition is much more important than stealthiness.
- The third scenario is a hybrid one, where SVC is used for much diversified everyday purposes. In this scenario, both factors of stealthiness and information acquisition should be balanced. So for different applications, we need to design different triggering mechanism for SVC.

To be exact, for stealthiness, the main challenge is to select a most opportune time to capture and send the video, as well as intelligent scheduling of camera functioning/hibernating. When SVC is working, it may consume the power and CPU cycles. To prevent device owner from suspecting, SVC must find a suitable opportunity to turn on those functions. For example, when a device owner is playing 3G Smartphone games, the CPU is busy. If the video capture is triggered at that moment, the owner may feel that the game runs too slow and suspect another program is running. When the device owner feels something wrong with the 3G Smartphone, he or she may carefully check if any abnormal program is running. Therefore, if *Triggering Algorithm* have the cognitive capability to detect device owners' contexts, it can be smarter and stealthier. The diagram of *Triggering algorithm* is shown in Figure 5.

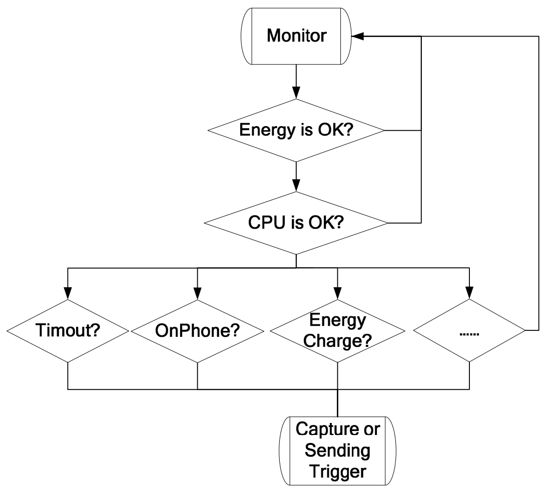


Figure 5: Working flow of Triggering Algorithm

Capture triggering algorithm employs Windows Mobile APIs to gather more context-aware information. It can get the power and CPU status and detect device owners' behaviors such as whether the owner is dialing numbers. Therefore, to make SVC stealthier, we designate some experiential observations for the "smart" *Triggering algorithm.* Generally speaking, the triggered opportunity should be selected in a period that a device owner should have low probability of suspecting his/her 3G Smartphone is in abnormal shape due to extra power consumption and CPU cycles etc.

On the other side, *triggering algorithm* should acquire more useful information at each execution. If SVC knows the living habits of the monitored device owner or owner's profile, it can be smarter to compromise users' privacy. When the device owner is doing something important, this period can be a good chance to trigger capture of video. If a CEO is known to be in the habit of holding a marketing briefing at certain time in a week, where some important decisions are likely to be made, then we can have our SVC-controlled Smartphone turned on in time to capture all the essential information disclosed at the meeting.

Thus there exists a tradeoff between stealthiness and information acquisition. Hence we need to balance these two factors when designing triggering strategies. Without any knowledge on the identity of our monitored victim, we can deduce some adaptive triggering algorithm based on features specific to 3G Smartphones. The following cases may be the good indication for Capture triggering algorithm to operate:

- The power of 3G Smartphones is in the interval of $[20\%, 80\%]$.
- The CPU is occupied no more than 50%.
- When the device owner is on the phone, he or she can not discover SVC is running.
- When 3G Smartphone's power is charged or when the WiFi, Bluetooth is on, it also can be a good chance to trigger capturing.

Sending triggering algorithm is similar to *Capture triggering algorithm*. *Sending triggering algorithm* controls the sending process and decides when to send the video data.

The basic function of *Sending triggering algorithm* is to judge whether 3G Smartphones are on line, i.e., 3G Smartphones are connected to the Internet through WiFi, GPRS or 3G. Thus *Sending triggering algorithm* can select a path to access the Internet. *Sending triggering algorithm* also needs to employ relevant Windows Mobile APIs to acquire more context-aware information. To make SVC stealthier, there also exist some schemes which are similar to *Capture triggering algorithm*, such as power consumption and CPU occupancy etc. We extend our discussions on smart triggering in Section 7.

We handle triggering algorithm by using system *CeSetUserNotificationEx()*. This function is particular in Windows CE because only Smartphones have the idling status. When Smartphones are in the idling status, normal application programs can not be called by the ordinary timer. But maybe the idling status is a good chance to capture video. We implement the cases in triggering algorithm by *CeSetUserNotificationEx()*. Through this special method, the triggering can work in the idling status of 3G Smartphones to launch SVC.

5. INFECTION METHOD

To embed SVC in a 3G Smartphone is called a infection process [22]. There are three ways for an SVC to infect a Smartphone: via a Trojan horse, a worm or a virus. We have employed a Trojan horse as the means for infection in our proof-of-concept implementation. 3G Smartphones owners tend to download softwares from Internet, such as games and tools. Trojan horse will pretend to be a useful tool or game for downloading. That's why Trojan horse is used as the major infection way for the Smartphones malware.

Our experimental SVC is hidden in the game of "tic-tac-toe " shown in Figure 6 that we develop in Windows Mobile environment. We first compile SVC to a binary executable file. Then this binary file is inserted as a resource file into the source project of "tic-tac-toe". After compilation, the "tic-tac-toe" game contains the executable codes of SVC. Our Trojan horse SVC is thus disguised under the appearance of "tic-tac-toe". When the "tic-tac-toe" game is executed, SVC obtains the initial address and length of the internal binary resources. It will also create a blank exe file in a covert place under system folder and copy all binary resource data automatically. System APIs are called to run SVC. This file will be executed repeatedly since then and independently from the original "tic-tac-toe" game.

The whole process of infection is hard to be detected by device owners. Owners will only take notice of the original normal software, but is unaware of the fact that the SVC has loaded itself in the memory. The SVC program will call system API functions to get itself loaded once it is embedded in 3G Smartphones.

6. EXPERIMENTAL RESULTS AND EVALUATION

6.1 Development Environment

This section evaluates the performance of SVC. The program runs on the O2 XDA Flame based on Windows Mobile

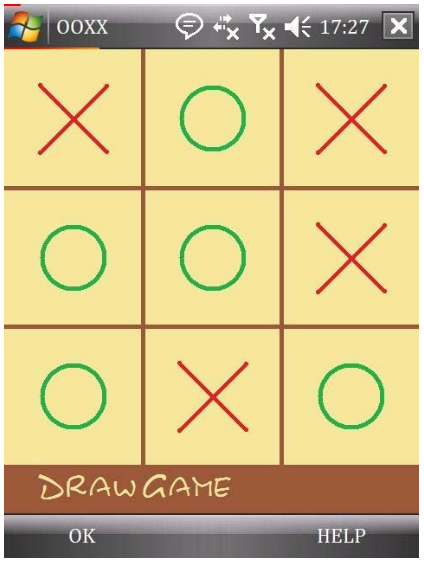


Figure 6: The scenario of tic-tac-toe

5 OS. The detail development platform is shown in Table 1. The experiment is carried on O2 XDA Flame and its specifications will affect our experiments. We list them in Table 2.

Device	O2 XDA Flame
Software Platform	Microsoft Visual Studio 2005; Microsoft ActiveSync 4.5 Developer; Windows Mobile 5.0 SDK for Pocket PC or for Smartphone; Windows Mobile 5.0 Emulator Images;
Tools	Visual C++

Table 1: Development Platform

Platform/OS	Windows Mobile 5.0
Network Technology	UMTS/GSM 900/1800/1900
Hardware/Processor	Intel XScale PXA270, 520MHz
Memory	2 GB + 64 MB Flash ROM, 128MB RAM
Battery	Li - Ion 1620 mAh, 3.7V
Messaging	SMS, MMS, Email
Wireless Connectivity	Bluetooth, 3G, GPRS, IrDA, WiFi(802.11b/g)
Camera Lens Type	CMOS, 2.0 Megapixel
Max. Resolution	1600*1200 (Secondary VGA camera)

Table 2: O2 XDA Flame Specifications

6.2 Evaluations

The purpose of evaluation is to do an analysis of CPU occupancy, memory usage and power consumption. Through experimental data statistics, we are able to completely reveal the stealthiness and running features of SVC.

We evaluate the performance of SVC with respect to different states. We divide the whole SVC execution procedure into four states, namely J1 (Backlight Off and System

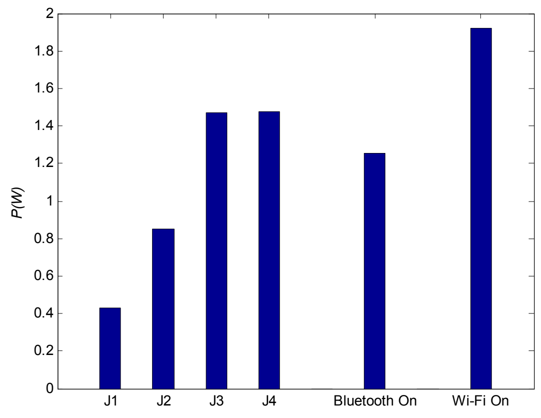


Figure 7: Power consumptions of the four phases

On), J2 (Camera Open and Shown), J3 (Video Capture and Encoding) and J4 (Video Capture, Encoding and Storage). Each state is explained below.

Power curve: In order to get power consumption of SVC, we employ *GetSystemPowerStatusEx2()* in our power testing program, which is in the Windows CE .Net library, to retrieve the value of system current and voltage. From Figure 7, we can see that the power consumption of the idle system with no backlight is only about 0.43W. And camera capturing and encoding are the main contributors to power consumption, which is about 2 and 3 times more than J1, while video file storage demands little power. The power consumption in J4, full execution of SVC, lies around 1.47W, which is between WiFi and Bluetooth power consumption. Through this analysis, we can see that our SVC's power consumption is roughly equal to that of wireless connections. Since WiFi and Bluetooth are among the most frequently used features of 3G Smartphones, the operation of our SVC is hard to be detected by device owners simply through observation of power consumption.

Figure 8 describes the power consumption in different video frame numbers. Frame rate is around 10 frames per second. We can see that power consumption is almost proportional to the number of frames. Therefore we can approximately compute the persisting time SVC can last. If the system is always idle, SVC can work about 4 hours. (Remark: 1.620*3.7/1.47=4.078) 4 hours is enough to secure some most valuable information, not to mention that the time may be substantially prolonged, if we turn the camera on for several minutes intermittently. So within a battery recharge period typically around 2-3 days for 3G Smartphones, we may span this whole period using our SVC without much impact on battery life-time. Thus a continuous and long-lasting surveillance of the victim is possible during these 2-3 days, avoiding the occasion that our peeking is disabled because of power failure.

CPU and Memory: We employ Supertasks V1.5 [3] to probe CPU occupancy. As Figure 9 shows, video capture of SVC alone consumes about 22% of CPU occupancy which is less than Windows Media Player (WMP) at 25%. However, with real-time encoding added by H.263, SVC occupies up to 88% of CPU load. Thus CPU is largely involved in data compression. And CPU occupancy of data storage process is insignificant. This 88% is quite a high burden on CPU. It attests to our concern in designing triggering algorithm that

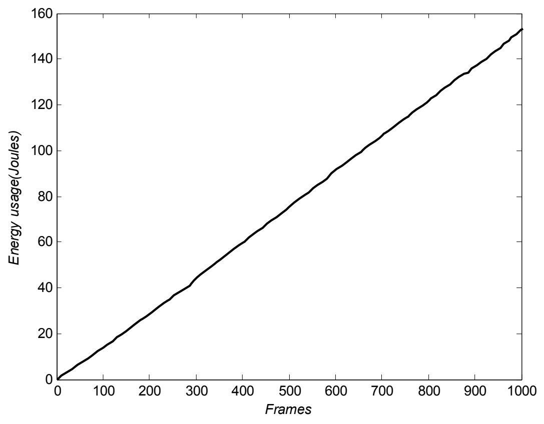


Figure 8: Power consumptions vs. Video Frames

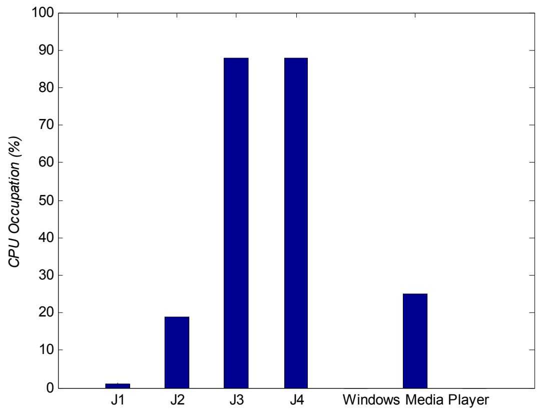


Figure 9: CPU usages of the four phases

our SVC should not be started when the CPU is heavily loaded. In the final analysis, SVC will not pose a significant impact on the routine functions, such as phonecalls and menu operations.

We also investigate the average memory usage of SVC and WMP by MemMaid 1.72 [4]. The memory usage is almost a constant. And memory usage of SVC is less than WMP, which is shown in Table 3. Compared with the total 128M Smartphone RAM, this memory usage is negligible.

Windows Media Player	1.61M
SVC	1.28M

Table 3: Average Memory Usage of SVC and WMP

File Size: The size of our video file is shown in Table 4. The image size is proportional to the number of frames. We develop H.263 data compression module as SVC encoding scheme to reduce the size of captured video file, and compared with native APIs, the size is notably cut down. The size of static image is much smaller than the mobile image because the mobile image sees less data compression ratio. Moreover, the stored video file is much smaller than system ROM (1/32). If compared with expansion memory (1/1024), it is negligible. It is hard for the device owner to detect where we've attached such a small file, which is also ready for wireless transmission.

Frames	Static Images(K)	Mobile Images(K)	Usage of ROM(%)
10	1.70	20.3	0.031
50	9.04	103	0.157
100	18.2	193	0.294
300	54.9	653	0.996
500	91.6	1030	1.572
1000	183	2048	3.125

Table 4: The Size of Video Files

6.3 Performance Analysis

We have evaluated three metrics (Power, CPU and Memory) for the performance of SVC. In triggering algorithm, we have mentioned that SVC can select a good opportunity to load itself. According to performance evaluation, we can see that recorded video less than 1000 frames each time is a good choice, because the size of the video file is suitable to send in Email mode and the duration of runtime is less than 100 seconds (1000/10). Therefore the probability of SVC being detected is very low.

Compared with J2, the values of two metrics (Power consumption, CPU occupancy) in J3 have an enormous increase. However, the values of J3 and J4 are almost the same. It's quite evident that video encoding costs much more power and CPU than file storage and it is almost half of the entire process. The underlying reason is that file storage is just simple hardware operations, which are much less complicated than computation-intensive video encoding. That is to say, how to greatly improve the performance of SVC is mainly dependent on a good real-time encoding algorithm, which should be fast and power-efficient.

For integral testing of SVC, since it is a malicious spyware, which is not suitable for online distribution, we've conducted a small-scale test amid ten persons of other groups in our lab. We gave them Smartphones carrying our SVC spyware. All of them were not aware of the abnormal behaviors of Smartphones when they played the "tic-tac-toe" game, and their captured videos in that day were sent to our test email.

7. DISCUSSIONS

7.1 Smarter Triggering Algorithms

SVC Triggering algorithms discussed in Section 4 intend to make SVC smart to become stealthy and efficient. The question is how to make SVC smarter to trigger video capture and sending. Context-aware approach may serve as a good candidate for SVC to grab the triggering opportunity. However, how to obtain the context information is also very challenging. For example, adaptive learning approach can be exploited to establish context aware info database according to the device owner's profile or pattern such as living habits.

Even without apriori knowledge, we can improve our triggering mechanism by exploiting Image Recognition algorithm to identify the captured video. It can help to recognize some features of the captured video. These features include the object's brightness, color and properties etc. For example, if the captured videos are always dark, then it is not necessary to store and send them. This can reduce the frequency of storing and sending files. On the other hand, it can improve the quality of the captured videos.

7.2 Resistant to Current Anti-virus Tools

Currently, commercial anti-virus tools are only good at pinpointing known features of malwares, such as signature or specific malicious maneuvers. Since SVC is a brand new spyware that exploits video-based vulnerability in 3G Smartphones, it is naturally resistant to the majority of anti-virus tools, like McAfee, Kaspersky and F-Secure mobile version, etc. The following several reasons all contribute to its robustness against anti-virus tools:

First of all, there are no relevant signatures of SVC in existing signature library of anti-virus tools.

Second, there are no malicious features in SVC which can be easily sensed by anti-virus tools, like code injection, file deletion and file self-copy etc.

Third, the intelligent triggering mechanism can easily defeat some signature based approach employed by some anti-virus tools. Generally no apparent anomalies of memory, power or CPU will be observable, so it's necessary for anti-virus tools to incorporate more advanced detection techniques to outwit our SVC, like that presented in [13].

7.3 Essential Factor of the Success of SVC

Nowadays Windows Mobile becomes more and more widely used as popular Smartphone OS. However, in spite of elaborate design of its security policy, Windows Mobile still falls easy prey to experienced hackers. The reason is many-fold.

First, the default setting for Smartphone often leaves Windows Mobile in the lowest two security levels out of the total four levels [27]. It's likely that device owners will not change them after acquisition of the Smartphone. Under these two security levels, malicious intenders can be easily authorized to call the system APIs, where all attacks can be launched without a hitch.

Second, the largest vulnerability in a security system is human. An amateur user can easily step into the trap by trying some 'funny' games, which is in fact a Trojan horse, and thereby legitimately allowing the malware enough clearance to complete all kinds of mischiefs.

In summary, the key factor leading to the success of SVC is that there is no efficient management policy of system APIs security for Windows Mobile. On one hand, for convenience and functionality, device owners can freely employ system APIs to realize various applications to achieve their different requirements. On the other hand, even set to high security levels, malicious usage of core APIs is still possible in the real world. It is a significant vulnerability in Windows Mobile, as device owners can use system APIs at will. Therefore, from the general viewpoint, the success of SVC is attribute to the loose management of system APIs.

7.4 Lessons From This Work

Based on our work, it can be seen that video based vulnerability is really existent in 3G Smartphones. The video-based spyware will invoke public attentions. The built-in camera will bring you entertainments but at the same time it may also turn to an evil eye. The video-based security problem in 3G Smartphones will be a potential threat to people's normal life. As we have shown that it is possible for malicious guy to exploit such vulnerability in a 3G Smartphone for wicked purposes.

Next what can we do to protect our privacy? 3G Smartphones are not as secure as expected. Good habits of using 3G Smartphones are necessary for Smartphone's security. For example, we should not arbitrarily download the executable file and should check Smartphone file directory regularly to examine whether there is any suspicious files etc [5].

8. RELATED WORK

Malware, e.g. like viruses, worms, Trojan horses and spyware have been threats to computer systems for many years, and security researchers constantly work on new countermeasures of malware, e.g. new malware analysis [16, 17, 18] and detection [19] technologies. However, because of the big market of Smartphones, malicious guys are bound to find the weaknesses and exploit them for mischief or, worse, for illegal gains. Researchers then gradually turn to focus on malware on Smartphones. We'll expand the correlative work from following aspects.

8.1 Audio-based and Video-based Spyware

Spyware is a kind of malware that is installed surreptitiously on a personal device to collect various types of personal information, without the user's consent. Researchers have recently found a new form of spyware [9] which targets the operating system for mobile phones. These types of spyware are becoming increasingly common. As a result, security issues become important as more sensitive data are carried in their mobile devices. This new form of spyware is a form of Trojan that tracks SMS text messages and allows them to be ready for others. FlexiSPY [10] is the latest in mobile device spyware that sells itself as a solution to "Protect your children" and/or "Catch cheating spouses" which can secretly record the phone's activities such as SMS messages sent or received, the call history, etc.

To date, there indeed exist some audio-based spyware [5] such as audio transcription in the real world 3G Smartphones. Using the mobile voice-recording APIs, an audio-based spyware can indeed change a mobile phone into a tape recorder. According to Windows Mobile SDK, Microsoft applies the Waveform Audio Functions to record and play wav files. Many audio recording APIs and codecs used by Windows can be easily applied to Windows Mobile because of the compatibility between Windows and Windows Mobile. As a result, these APIs can conveniently serve as a reference for the malicious guys.

However, realization of video-based spyware will introduce prohibitive difficulty compared with the audio-based spyware counterpart. The most important difference is that a 3G Smartphone usually has a built-in camera whereas a PC may not be equipped. Therefore, Windows Mobile SDK have video-based APIs for camera development but Windows SDK does not have such APIs. Second, the native direct APIs for camera development supported by Windows Mobile SDK are not flexible. The native direct APIs do not support the functions such as choosing data storing type and data compression, therefore, we must develop our own COM controller (DirectShow) to replace the native direct APIs with such function extensions. Third, there is not only one video codec mode for video capture in 3G Smartphones, so we should implement several video codec modes like RGB and YUV and apply them adaptively. Therefore, the realization of video-based spyware is much more difficult than that of audio-based not only in SDK aspect, but also in the design and implementation aspects.

The video-based spyware has already existed in PC. It is like a Surveillance Tool that monitors and captures data from computers through Web cam [11]. However, the Video-based spyware in PC is less harmful than in 3G Smartphones because of 3G Smartphone's mobility. The portable character of 3G Smartphones makes the video-based spyware in it capture more useful information than the spyware in PC. And a together-viewing video application in two proximate mobile device is presented in [23]. So far there is no relevant spyware in 3G Smartphones.

8.2 Other Mobile Malware

Mobile Malware has a short history. The first mobile phone virus, Cabir, was reported in 2004. However, mobile viruses and similar threats have turned out to be a minor concern. Is it finally time to worry about Mobile Malware? [6] The answer is absolutely "yes". Why? As Smartphones get cheaper, more people are using the devices, making Smartphones much more attractive targets. One of the early papers [7] on Smartphone malware describes the taxonomy of mobile malware. They point out that three goals are compromised: confidentiality, integrity and availability. Another early paper on mobile malware in [8] presents the first worm sighted for Symbian OS. If installed, the Smartphone will always search for nearby devices with activated Bluetooth interface to propagate. Mikko Hypponen [2] provides an overview on Smartphone. It gives an example to illustrate the procedure of a malware attack.

Recently, there are many research papers focusing on mobile malware detection, propagation and vulnerability in Smartphones [21]. For mobile malware detection, "SmartSiren" [12] represents an approach which collects communication data and sends it to a remote server in order to offload the processing burden from resource-constrained Smartphones. The data is jointly analyzed to detect both single-device and system-wide abnormal behaviors. In [13], a novel behavioral detection framework is proposed to detect mobile worms, viruses and Trojans, instead of the signature-based methods available for use in mobile device. It indicates that behavioral solution can identify current mobile malware with more than 96% accuracy and is acceptably low in cost for deployment. [14] addresses the problem that the limited battery-lifetime of mobile devices can be exacerbated by mobile malware targeting depletion of battery energy. It proposes a power-aware malware detection framework that monitors, detects, and analyzes previously unknown energy-depletion threats.

For mobile malware propagation, [15] models realistic topologies and provisioned capabilities of the network infrastructure, as well as the contact graphs determined by cell phone address books. Then they evaluate the speed and severity of random contact worms in mobile phone networks.

For Smartphone vulnerability, researchers are always working on finding new security flaws. Mulliner et al. [20] present a way to perform vulnerability analysis on MMS user agents of Windows Mobile phones. They detect vulnerabilities which can be used to create remote fault code injection attacks through MMS messages. Racic et al. [24] present an operating system-independent attack utilizing vulnerabilities in the MMS protocol. Also mitigating attacks on open functionality in SMS-Capable cellular networks are discussed in [25].

9. CONCLUSION AND FUTURE WORK

In this paper, we present SVC, a video-based spyware in real world 3G Smartphones. We study the video-based vulnerability in 3G Smartphones which can seriously compromise user's privacy. We identify the key challenges of SVC and present innovative design and implementations. A set of schemes are employed to enhance its stealthiness, including covert infection and smart triggering. We adopt the principle of Trojan horse for infection method and exploit 3G Smartphone's particular inner processes to put SVC as the background service program. Meanwhile, through smart triggering, we manage to enhance SVC's stealthiness, as well as improve its efficiency in video data acquisition. We have conducted extensive evaluations on our test bed and the experimental results have demonstrated the stealthiness and efficiency can work together in SVC. It's also proven that SVC is robust against almost all existing anti-virus tools. Furthermore, we expose a fundamental tradeoff between user convenience and system security for Windows Mobile.

We believe that the initial study exploited from SVC will draw widely attentions on 3G Smartphone's privacy protection and open a new horizon on 3G Smartphones security research and applications. Meanwhile, the applications of SVC in such cases as suspect tracking and kids care are also promising. We are currently investigating the modeling of smart spyware from the perspectives of stealthiness, video-based vulnerability and profile adaptive learning as well as the counter measurements against the new spyware, i.e. the study of "spear and shield".

Acknowledgement

The work is partially supported by Hong Kong SAR Competitive Earmarked Research Grant –GRF Grant No. 9041350 (CityU 114908) and CityU Applied R & D Centre (ARD(Ctr)) grant No. 9681001, and this work is also supported in part by the US National Science Foundation (NSF) CAREER Award CCF 0546668, the Army Research Office (ARO) under grant No. AMSRD-ACC-R50521-CI.

10. REFERENCES

[1] Canalys.com. http://www.canalys.com/pr/2008/r2008021.htm

[2] M. Hypponen. Malware Goes Mobile. *Scientific American*, pp. 70-76, 2006

[3] SuperTasks. http://www.softwareandson.com/SuperTasks/

[4] Memmaid. http://www.dinarsoft.com/memmaid/

[5] Z. Cheng. Mobile Malware: Threats and Prevention. *McAfee Avert®Labs Technical White Papers*, Sept., 2007.

[6] G. Lawton. Is It Finally Time to Worry about Mobile Malware? *Computer*, 41(5), 12-14, 2008.

[7] D. Dagon, T. Martin and T. Starner. Mobile Phones as Computing Devices: The Viruses are Coming! *IEEE Pervasive Computing*, 3(4), 11-15, 2004.

[8] M. Piercy. Embedded devices next on the virus target list. *Electronics Systems and Software*, 2(6), 42-43, 2005.

[9] Newest Electronic Threat: Mobile Spyware. http://www.nospysoftware.com/spyware-news/spyware-adware-mobile.php

[10] FlexiSPY Mobile Spyware: Monitoring solution or Security Nightmare. http://www.informit.com/articles/article.aspx?p=1185592

[11] Hidden Camera Threat Display. http://research.sunbelt-software.com/

[12] J. Cheng, S.H.Y. Wong, H. Yang and S. Lu. SmartSiren: virus detection and alert for Smartphones. In *Proc. of MOBISYS'07*, 258-271, 2007.

[13] A. Bose, X. Hu, K.G. Shin and T. Park. Behavioral detection of malware on mobile handsets. In *Proc. of MOBISYS'08*, 225-238, 2008.

[14] H. Kim, J. Smith and K.G. Shin. Detecting energy-greedy anomalies and mobile malware variants. In *Proc. of MOBISYS'08*, 239-252, 2008.

[15] C. Fleizach and M. Liljenstam, et al. Can you infect me now?: malware propagation in mobile phone networks. In *Proc. of WORM'07*, 61-68, 2007.

[16] R. Perdisci, A. Lanzi and W. Lee. McBoost: Boosting Scalability in Malware Collection and Analysis Using Statistical Classification of Executables. In *Proc. of ACSAC'08*, 301-310, 2008.

[17] A. Dinaburg, P. Royal, M. Sharif and W. Lee. Ether: Malware Analysis via Hardware Virtualization Extensions. In *Proc. of CCS'08*, 51-62, 2008.

[18] M. Sharif, V. Yegneswaran and H. Saidi, et al. Eureka: A Framework for Enabling Static Malware Analysis. In *Proc. of ESORICS'08*, 481-500, 2008.

[19] G. Gu, P. Porras, V. Yegneswaran, M. Fong and W. Lee. BotHunter: Detecting Malware Infection Through IDS-Driven Dialog Correlation. In *Proc. of USENIX Security Symposium*, 167-182, 2007.

[20] C. Mulliner and G. Vigna. Vulnerability Analysis of MMS User Agents. In *Proc. of ACSAC'06*, 77-88, 2006.

[21] C. Mulliner, G. Vigna, D. Dagon and W. Lee. Using Labeling to Prevent Cross-Service Attacks Against Smart Phones. In *Proc. of DIMVA'06*, 91-108, 2006.

[22] P. Royal, M. Halpin, D. Dagon, R. Edmonds and W. Lee. PolyUnpack: Automating the Hidden-Code Extraction of Unpack-Executing Malware. In *Proc. of ACSAC'06*, 289-300, 2006.

[23] G. Shen, Y. Li and Y. Zhang. MobiUS: Enable Together-Viewing Video Experience across Two Mobile Devices. In *Proc. of MOBISYS'07*, 30-42, 2007.

[24] R. Racic, D. Ma and H. Chen. Exploiting MMS Vulnerabilities to Stealthily Exhaust Mobile Phone's Battery. In *Proc. of Securecomm and Workshops*, 1-10, 2006.

[25] P. Traynor, W. Enck, P. McDaniel and T.L. Porta. Mitigating attacks on open functionality in SMS-capable cellular networks. In *Proc. of MOBICOM'06*, 182-193, 2006.

[26] New Cabir Worms Target Mobile Phones: http://www.viruslist.com/en/analysis?pubid=200119916

[27] Security Model for Windows Mobile 5.0 and Windows Mobile 6. http://www.microsoft.com/technet/solutionaccelerators/mobile/maintain/SecModel/aff7cf7f-0e11-4ef4-8626-f33bd969b35a.mspx?mfr=true

[28] A. Ranjan. Using a Camera with Windows Mobile 5. http://www.developer.com/ws/pc/article.php/10947_3621211_1,July 21,2006