Mobile Evil Twin Malnets – The Worst of Both Worlds

Christian Szongott, Benjamin Henne, and Matthew Smith

Distributed Computing & Security Group
Gottfried Wilhelm Leibniz University of Hannover
{szongott,henne,smith}@dcsec.uni-hannover.de
http://www.dcsec.uni-hannover.de

Abstract. The mobile computing world is undergoing major changes both in the capability as well as in the proliferation of mobile devices. While, up to now, mobile malware has played a relatively small role compared to the behemoth of desktop malware, the changing environment is steadily increasing the attractiveness of mobile devices as exploitable resources. The increased usage and connectivity of mobile devices opens up a much larger set of attack vectors to compromise them. In this paper, we adapt the evil twin rogue access point attack to the mobile domain and show how it can be used to create a mobile malnet, which is capable of spreading epidemically. We implemented the key components of the concept for the iPhone to study its properties in a laboratory environment. To demonstrate the dangers which come along with this kind of attack we simulate a metropolitan area and show how fast a malware can spread in a mobile environment.

1 Introduction

Researchers have predicted the epidemic spread of mobile malware many times in the last decade. However, a true outbreak of mobile malware has not occurred in the wild yet, leading to a certain level of complacency towards the threat of mobile malware.

That mobile devices have not been targeted in the past is partly due to the fact that the proliferation and capabilities of mobile devices were so limited that they were consequently also only of limited use to attackers. Compared to the effectiveness of Wi-Fi desktop worms with epidemic qualities such as the wildfire worm presented by Akritidis et al. [1] the capabilities of worms for mobile phones have been relatively small.

However, several key factors in mobile computing have changed in recent years, significantly altering the playing field and giving rise to a new threat of mobile malware [2]. The proliferation and capabilities of mobile, networked devices has increased markedly and the mobile phone has become the central digital hub of our lives, storing personal information, multimedia content as well as offering access to social networks, online banking, work networks and a host of

J. Pieprzyk, A.-R. Sadeghi, and M. Manulis (Eds.): CANS 2012, LNCS 7712, pp. 126–141, 2012.

other functionality. This increases both the value as a target as well as the attack surface. Recent work has also shown the potential dangers of mobile malnets [3]. Malnets are botnets created from routers, cellphones, and other non-traditional computational Wi-Fi devices. The main focus of malnet research has been in the area of router attacks, such as presented by McDaniel [4].

In this paper we show how recent advances in mobile devices can be used to combine the most dangerous aspects of these two worlds - mobile malnets and Wi-Fi worms. We will show how the inclusion of mobile hotspot capabilities in virtually all new mobile devices opens the door to leverage the evil twin attack[1] to create a mobile malnet capable of spreading from device to device. Unlike in previous work we do not require vulnerabilities in the Wi-Fi access points, nor it is a problem that most private and corporate Wi-Fi networks now use WPA to encrypt their traffic. To test the feasibility of our approach we implemented the key components needed to create the malnet for iOS. In lab experiments we analyze the basic parameters and requirements of this malware mechanism and measure infection and propagation times in a controlled environment.

The rest of the paper is organized as follows: In section 2 related work in the field of mobile malware and evil twins is presented. Section 3 describes how mobile evil twin malnets work and which weaknesses of todays smartphones are utilized to render this type of attack possible. Additionally our prototypical implementation is described in-depth. In Section 4 we present the simulations we used to study the spreading characteristics, their assumptions and results. Section 5 concludes the paper and gives an overview of possible future work in this research area.

2 Related Work

There is a large body of related work discussing the spread of mobile device malware. Most of this work [5,6,7,8,9,10,11] relies on Bluetooth or user errors as an infection vector. One prominent example for Bluetooth-based worm research is described in a paper from Wang et al. [12] where they analyzed the spreading patterns of mobile viruses. They considered Bluetooth and multimedia messaging as distribution channels for this kind of worms. They predict serious threats to mobile phones once an operating system reaches high enough market shares. How vulnerable even *feature phones*[2] are, is shown in the work by Collin et al. [11]. They studied the vulnerabilities of these mobile phones and possible attacks against the mobile network using SMS and Bluetooth.

In 2007 Akritidis et al. [1] presented a simulated study of Wi-Fi-based malware in metropolitan area networks. The work explores the idea to utilize the proximity of Wi-Fi routers to spread malware. Akritidis et al. present two modes

[1] Evil twin is a term for a malicious Wi-Fi access point that appears to be a legitimate one by spoofing its SSID.

[2] Phones that are not considered to be smartphones, but have additional functionality beyond standard mobile services.

of infection labeled push and pull. Push infection resembles traditional worm infection methods most closely. With push infection a vulnerability in a device connected to a vulnerable hotspot is exploited to infect that device, which then goes on to infect further devices. The second method is the pull method. In this case the infected node waits for a device connected to a vulnerable access point to make some form of network request and then injects the worm code into this connection. The advantage of this method is that instead of relying on a service vulnerability, the attacker exploits vulnerabilities, such as browser vulnerabilities which are much more frequent and patched at a slower rate. In the studies case both the push and the pull method rely on visible, open and unencrypted access points to launch the attacks against other connected devices.

Another source of related work and inspiration for our work are malnets. Malnets are botnets created from non-traditional Wi-Fi devices such as mobile phones, but in particular wireless routers. In their work McDaniel [4], Hu et al. [13] and Traynor et al. [14] show that traditional network routers can be used to build up large malnets in areas with a high density of routers. The attack uses weak or default passwords to compromise wireless routers, which are in range, which then in turn attempt to infect other vulnerable routers in range. They developed epidemiological models and showed that in densely populated areas with enough vulnerable routers an epidemic spread of the malnet is completed within hours. While not strictly being a malnet the research presented by Akritidis et al. above has similar properties. But since these works only consider static environments where APs themselves have to be infected we describe in this work a new kind of infection and explore an alternative route to creating malnets, which does not require vulnerable routers.

The attack vector we utilize is based on the evil twin attack presented by Bauer et al. [15]. They showed that it is possible to trick a wireless client into associating with a rouge (evil twin) access point. The paper describes how an attacker can execute MITM attacks against selected victims by tricking them into connecting with the evil twin hotspot instead of the legitimate one. While this was viable and serious even then, it did not receive a great deal of attention. We assume this lack of interest was due to the fact that at the time of the attack it was only discussed in the context of infrastructure-based Wi-Fis, where targeted attacks against specific locations were the goal. The proliferation of WEP, WPA/WPA2 made this attack significantly harder in most scenarios. Also the effort of setting up the evil twin and the targeted nature of the attack reduced the mass-effect of the approach. The changes in the mobile computing landscape and our research into extending this attack by leveraging the capabilities of current smart mobile devices to create a potentially massive mobile malnet increases the threat significantly.

3 Mobile Evil Twin Malnets

In this section we show that the currently dormant threats shown above can be combined and extended to once again be a serious threat in the new mobile

computing landscape. There are two key factors which enable the approach presented in this paper; the proliferation of Universal Authentication Mechanism (UAM) Wi-Fi hotspots [16] accessed through captive portals and the capability of modern smart mobile devices to act as a mobile hotspot. Before we present the concept of the mobile malnet, we discuss the usage and security environment which has now made this type of malware possible.

3.1 Wi-Fi Hotspot Usage and Security

Wi-Fi access has become a commodity service in most urban areas, such as malls, coffee shops, hotels, airports and other public locations. A relatively small number of Wireless Internet Service Providers (WISPs) offer Wi-Fi access for either ad-hoc usage or bundled with other services such as mobile phone plans.

While most Wi-Fi routers now usually come pre-configured with WPA/WPA2 or similar channel encryption mechanisms, public Wi-Fi is often unencrypted. The lack of channel encryption for public Wi-Fi is mostly due to the fact that WISPs have not found a way to incorporate the necessary configuration steps into their business and usage model. While the lack of encryption brings with it a number of security problems such as impersonation, credential theft and other violations of a Wi-Fi user's privacy, the lack of strong mutual authentication during the connection to a public Wi-Fi hotspot and the unsecured use of SSIDs (Service Set Identifiers) for both ESS (Extended Service Set) and IBSS (Independent Basic Service Set) networks is the main problem.

In most public Wi-Fis authentication of users is handled by the UAM. With UAM, any device is allowed to associate with the Wi-Fi access point (AP) and is issued an IP address and other network information such as the standard gateway automatically via DHCP. After association, the user opens a web browser and enters any URL. A transparent HTTP proxy (also called captive portal) on the AP (or the underlying infrastructure) captures the request and redirects it to a login page. In the case of free Wi-Fi, the user is usually just required to accept the terms of use; in the case of subscription or pay-as-you-go Wi-Fi the user needs to provide valid credentials or purchase access on the fly. The credentials entered in the login form are forwarded to a back-end processing facility (AAA server, credit card processor, etc.) and after successful validation the user's session is started. Now the user's primary HTTP request is sent and the response is delivered to the user. Critically, the content of this portal is controlled entirely by the WISP. Later in this paper we will show that this issue creates several dangerous problems.

Today there is no practical way to authenticate APs wherefore it is neither mandatory nor automated and common. If at all, it needs to be performed by the user checking a SSL/TLS certificate. Unlike the use of certificates in traditional client/server Internet applications (which also can be fraught with difficulty), this measure has very little security effect in practice, when applied to hotspot security. SSL certificates are issued for a FQDN (fully qualified domain name). However, since there is no verifiable or even any form of specified link between the FQDN and the SSID, the certificate-based approach is not feasible. Thus,

the vast majority of users will not be able to associate a hotspot SSID with the "correct" host name for the hotspot's authentication service.

The final problem in current Wi-Fi security is the fact that for increased usability many mobile devices are configured to automatically connect to known Wi-Fi hotspots, where "known hotspot" is defined only by the SSIDs being equal. Since SSIDs are not authenticated this opens up the possibility of the evil twin attack [17].

3.2 Mobile Evil Twin Attack

If a device has been connected to a hotspot with a given SSID at any time in the past, it is susceptible to an "evil twin" attack. An attacker positioned outside the Wi-Fi range of the real hotspot or with a stronger signal can create an evil twin hotspot by spoofing its SSID. In the previous section, we showed that there is currently no practical way for users of a UAM-enabled Wi-Fi hotspot to reliably determine its authenticity. So if an evil twin broadcasts a "known" SSID and a device connects to it, the evil twin is then capable of executing man in the middle attacks (MITMAs) against that device. In this paper we introduce an adaptation of the evil twin attack to create a far more potent attack type, the Mobile Evil Twin attack (MET). Unlike the evil twin attack, the MET does not aim at subverting a specific AP to carry out targeted attacks, but to misuse the UAM weaknesses as an infection channel and replicate itself.

In order to spread, any malware needs to find vulnerable hosts and an infection vector. The environment described above offers both. In a lot of countries, mobile hotspots are ubiquitous in many establishments like Starbucks, Barnes&Noble and McDonald's. These hotspots usually have a single SSID per brand to allow customers to easily connect to hotspots across any of their locations. In some cases even the WISP's default SSID is used (i.e. by BT in the US and the UK, which use the SSID `BTOpenzone`). While the default SSID for public Wi-Fi makes sense from a usability perspective, it opens the door to the MET attack since it creates a large user base which has connected to a well known SSID. In the following, we will show the design of the MET attack and the prototypical implementation of the key components. The implementation extends the jailbreakme.com jailbreak to deploy all components for a self-replicating mobile malnet. All components were separately tested in the lab using iOS 4.3.3 running on an iPhone 4, an iPad 1 and an iPad 2. The principle can be ported to any OS with a root vulnerability. However, the usability feature of an automatic browser-based captive portal popup makes iOS the more attractive target. The whole attack process consists of four steps that are graphed in Figure 1 and described below.

Step 1: Masquerading and Injection. The first step of the MET attack is the exploitation of the auto-reconnection feature to known hotspots. The MET attack starts with the initial infection host broadcasting an SSID corresponding to a well used SSID such as those used by public hotspots like *tmobile*. If a mobile

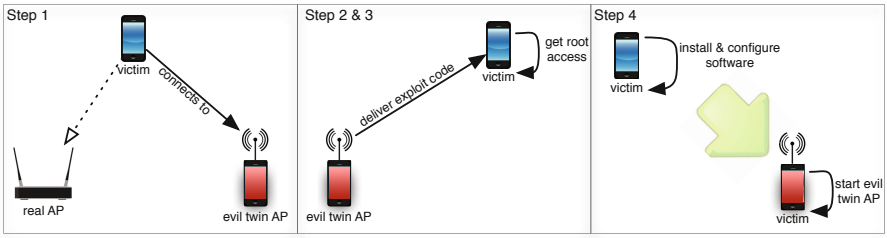

Fig. 1. MET attack overview

device has previously been connected to a legitimate AP of that SSID label, it is likely that it will automatically connect to the MET hotspot, unless there is a competing AP with higher signal strength or a SSID with higher priority.

Once the mobile device is connected to the MET hotspot all Internet activity of the device goes via the hotspot. With this setup, we can run a rogue Wi-Fi hotspot which can not only sniff, but also modify all unencrypted communication sent through it and inject malicious content.

The proof of concept malware presented in this paper uses `pf`, the iOS internal firewall and packet forwarding engine, to redirect all incoming traffic on port 80 to a locally deployed web server where the malicious payload is hosted. A more detailed description will follow later in this paper.

Step 2: Application Exploit. Once the mobile device is connected to the MET hotspot the second step of infection can occur. Since we use iOS for the prototypical implementation, which creates a MobileSafari popup window during the login process on the captive portal we can utilize a browser-based attack.

For all of the relevant smartphone platforms, i.e. iOS, Android, Palm/HP WebOS, BlackBerry and Symbian, the preinstalled web browsers are based on the WebKit [18] rendering engine. The WebKit engine is under constant scrutiny by security researchers and has had a number of security issues in the past (e.g.. [19,20,21]). Many of these security issues could be exploited by an attacker to execute arbitrary code on the phone with the same permissions as the web browser. Based on the long years of experience with desktop browsers and the recent history of mobile browsers, it can be assumed that mobile browsers will continue to be affected by security issues that allow arbitrary code execution for a fair while. For the proof of concept implementation we used the jailbreakme.com jailbreak which makes use of a PDF rendering vulnerability of the CoreGraphics framework in iOS 4.3.3 [22,23].

Step 3: Kernel Exploit. Executing code with the browser's permissions is rarely sufficient to install malware on a device. Compartmentalization and "sandboxing" of processes is a typical security approach for modern operating systems, including those on smartphones. Usually this means that the web browser – which runs under

an unprivileged user id – does not have privileges to install software on the phone. Therefore, another successful exploit on the operating system level is necessary to obtain full administrative privileges. Although harder to find than bugs in WebKit, these exploits exist and have existed for most mobile operating systems and it is likely that they will be available for future OS versions.

Step 4: Crossing-Over into the Evil Twin Malnet. Once a kernel-level exploit was executed on a mobile device the attacker gains complete control over that device, including the ability to control network and radio functions. This control is necessary for the propagation concept which we will outline in the following section. It makes use of the "personal hotspot" feature (Apple iOS $\geq$4.3). This hotspot functionality can be used by the device owner to share their 3G Internet connection with others by creating a mobile Wi-Fi access point. It conveniently bundles a small DHCP server, routing capabilities and Wi-Fi channel management and the setup is trivially easy. During the MET attack the mobile hotspot is activated using the target SSID making the device a new MET carrier.

By deploying *lighttpd* as a light weight web server on the infected mobile phone and rerouting all HTTP traffic of connected devices to it, we can deliver malicious code within any HTTP response and thus infect more mobile devices. Now, the infection cycle starts back at step 1.

3.3 Automating Infection

In the steps above an infection only occurs when a user manually triggers an event that receives content from the Internet into which the attack code can be injected. However, iOS has a convenient usability feature which enables almost instantaneous infection. When the iOS device is active, it automatically connects to the MET captive portal and as soon as any application requires Internet access an automated pop-up meant for credential input is shown.

This login page contains content which is fully controlled by whoever runs the Wi-Fi hotspot. It is, moreover, rendered with the same browser engine that is used for all other browsing activities. Therefore, virtually all exploits and security issues that exist for the WebKit browser family can also be exploited on that login page. Since the login page is displayed automatically, the MET infection routine can be executed as soon as the iOS device requests data from the MET hotspot.

Under the hood the following happens. After successfully associating with an AP and receiving an IP address, the iPhone's network stack sends one HTTP request to a specified URL. In the majority of cases the request goes to `http://www.apple.com/library/test/success.html`.

The response is then evaluated by iOS and if it consists of a valid HTML document containing only HTML metadata and exactly the word "Success" in the title tag, the network stack uses this as an indicator for the Internet