

## Blog Export: Exile From the Herd, <http://www.privateworld.com/>

Tuesday, September 4, 2007

### iPhone unlocked on the Fido network using Turbo Sim card

I bought a turbo Sim card on the aftermarket (at a pretty inflated price), from PDAPlaza, it arrived pretty quickly.

I was going to wait for a software unlock but I broke down before the weekend and grabbed a couple turbo SIMs.

There are a lot of docs out there on what to do, I thought I'd post a quick summary of what you need to do to get iPhone unlocked on a Canadian network like Fido or Rogers (I guess the point is this works on pretty well any GSM network)

In a nutshell, here's what you need to do to unlock your iPhone with a Turbo Sim card:

Get an iPhone. I had a friend ship up a couple from Chicago via FedEx a few weeks ago and have been using it as a WiFi device, without the phone, until now.

Get a turbo sim card. They are manufactured by Bladox, out of the Czech Republic. But the bad news is, since it was discovered these work on iPhones, demand skyrocketed and they are perpetually sold out. You may end up having to buy one on the aftermarket. Prices range from \$100+ (check Craigslist), I got mine from PDAPlaza.ca for \$175. On eBay prices are as high as \$600+. It'll be interesting to see what the software unlock will do to these prices.

Make a note of your IMEI and your ICCID. If your phone has been hacktivated (this means you have activated your iPhone without signing up with AT&T and can use everything except the phone), you will find it under Settings -> General -> About if you've already activated your phone, or by pressing the "i" icon on an unactivated phone.

Jailbreak and activate your phone, or activate and then jailbreak. The activation, like I said, gets you using the iPhone features except the phone, jailbreak allows you to install other applications and mods on the phone (which you'll need to install the turbo sim stuff). I used this guide for the Mac OS X to activate and jailbreak with iActivate.

You need to then install ssh, I found two tutorials, this one worked for me, and this one does a good job of explaining how what you are doing to the phone actually works, although after I completed the latter tutorial, I couldn't ssh into my phone, but I could after following the first one. (it took me a couple go rounds at this, I started from scratch twice, by resetting my iPhone to the factory defaults via iTunes)

When the tutorial goes through the bit about compiling iPhuck, don't. I found a binary download somewhere for the Mac, although I can't find the link at the moment. When I do I'll update this.

I didn't bother using Cyberduck or Fugit, I'd just scp stuff over from the Mac shell.

Once you're done that, you're finally ready to install the turbo sim. There are two methods of doing so, "ISA" and "applesaft", this is the definitive guide for the applesaft method, I couldn't find a tutorial for the ISA method. The difference between the two is the ISA method ONLY requires that you have to cut your target sim card to attach the turbo sim, the applesaft method requires you cut both your target sim and the ATT sim which came with your iPhone. The applesaft method basically is you put the turbo sim in with the original ATT sim, and install a turbo sim software bundle, then remove the sim combo, and attach the turbo sim to your intended sim card (in my case, my Fido sim, with the corner cut out) and after a power cycle of the phone, away you go. The turbo sim came with a cardboard diagram of how to cut the sim which I used as a defacto template/pattern. Not really my forte (anything bordering on a hardware mod) but I muddled through.

Update: May as well activate the iPhone for the Edge with Fido, so when I'm not in a WiFi hotspot I can at least access the internet over GPRS. Under Settings -> General -> Network -> Edge enter:

APN: internet.fido.ca  
user: fido  
pass: fido

So now I have a fully functional iPhone on Fido. I wonder what tomorrow's big announcement from Apple will be and whether they will address the groundswell of unlocking activity the iPhone has generated.

Posted by Mark Jiftovic in Hacking tech at 23:40

Tuesday, August 21, 2007

### **easySPF: an ajax enabled SPF wizard**

Creating Sender Policy Framework (SPF) data is a task that naturally lends itself to ajax. As you tweak the parameters of your policies, the spf record changes on the fly. This was a good opportunity for me to learn jquery and I was pretty impressed with how fast jquery enabled me to get a basic framework up and running.

What took me the longest was figuring out how to get the contents of my form processing via an external php script before being rendered into my inline , after awhile I gave up and asked one of my programmers. Turns out it's the load() event handler, but the jquery page doesn't really make it evident in the documentation that what I want to do is this:

```
$("#my_div_id").load("some_script.php", { myvar: myvar } );
```

Then I can access the myvar and anything else passed in the parameter list via the php \$\_REQUEST structure.

Anyhoo, the result is easySPF: an ajax enabled SPF wizard.

Posted by Mark Jeftovic in Hacking tech at 20:13

Monday, June 4, 2007

### Migrating your feed subscribers to Feedburner

I spent a few hours friday working on a blog widget that would parse the number of subscribers out of the HTTP\_USER\_AGENT strings of the various feedreaders so I could easily see the aggregate number of subscribers I have. It only took me a few hours to realize that what I was trying to do, wasn't gonna work.

My plan to use remote javascript may have worked fine for human readers to the main index, a la MyBlogLog (which I use), but of course, duh, the feedreaders won't load the main index and they likely won't interpolate the javascript. I doubted an iframe would work either.

So my idea for a widget wasn't going to fly, I realized the only way to do this was to actually run the entire feed URL through the counter system, the way FeedBurner does it. It was only a coincidence that this happened on the same day Google bought FeedBurner for 100 Million.

So the blog widget is a non-starter, I still want to easily know my aggregate subscribers but now I don't want to start counting from zero, I want to be able to factor in my existing readers.

My existing readers subscribe to my feed via <http://mark.jeftovic.net/feeds/index.rss2>, which is the same URL feedburner will use to load the feed. So simply redirecting my existing feed URL to my new Feedburner feed would create an infinite loop when FeedBurner tries to load the old feed.

We need to always redirect the old feed URL to the new feed URL except when the remote client is Feedburner.

My blog platform is serendipity, and it implements all the RSS and XML urls by a URL rewrite to rss.php, so at the very top of that file, we just add this:

```
if(!ereg("^FeedBurner",$_SERVER['HTTP_USER_AGENT'])) {
    Header("Location: http://feeds.feedburner.com/jeftovic/JVMu");
    exit;
}
```

This should allow Feedburner to load the RSS from the blog, and redirect all the other feedreaders to Feedburner. Hopefully I haven't overlooked anything. This is somewhat of a test post and if all goes well, it should show up in my Bloglines reader transparently.

To migrate to FeedBurner from other blog platforms including Wordpress and Drupal, there is this thread on the FeedBurner forums.

Posted by Mark Jeftovic in Hacking tech at 22:33

Saturday, May 20, 2006

### **Useless mash-up between pdns pipe backend and md5**

I've had the idea to do some tinkering with powerdns' pipe backend for awhile and had decided to rig up a quick and dirty MD5 encoder/decoder database on a domain I've had for awhile MD5.ORG  
So it was pretty easy. Now anybody can quickly get an MD5 hash on any string that'll fit inside a DNS query packet by doing something like:

```
host -t txt .to.md5.org
```

or the slightly less readable

```
dig -t txt .to.md5.org
```

Then for completeness you can try and see if we already know what string creates a given MD5 hash and retrieve it using

```
host -t txt .from.md5.org
```

There is not a lot of practical value to this, it was just a neat hack to learn the basics of the pdns pipe backend, which I like a lot.

Posted by Mark Jeftovic in Hacking tech at 16:20

Friday, December 16. 2005

### Ebay's move to free and grokking PHP's Services\_Ebay

Ebay made a very smart move recently and removed most of the costs associated with their developer platform. I think this will bring a lot of innovative new services to the market as the barrier to entry is now largely technical and not financial.

One of the reasons Amazon's AWS and affiliate program is so ubiquitous across the net is largely due to two things: 1) the depth and stickiness of the Amazon site and 2) the free access to their affiliate and AWS services.

Ebay has one of the few sites whose depth and prevalence approaches that of Amazon, although a back-of-the-napkin comparison shows Amazon dwarfs Ebay in backlinks by about 28 million to 6 million.

I started playing with PHP's pear class Services\_Ebay and after some struggling, I finally have a handle on it and thought I'd post some of the "aha" links which helped get me there. Programming eBay Web Services with PHP 5 and Services\_Ebay covers getting up and running. Here are a couple of handy calls to know:

```
$session = Services_Ebay::getSession($devId, $appId, $certId);
$session->setToken($token);
$call = Services_Ebay::loadAPICall('GetCategories');
$call->describeCall();
$result = $call->call($session);
print_r($result);
```

The describeCall() describes any method you care to load via loadAPICall() and you can see the complete list of methods in the package via getAvailableApiCalls()

The describeCall() output links to the eBay Developers documentation on the given call.

For example, getCategory() is documented here, which explains which parameters to pass to obtain the version number of the current category list and what to pass to dump the entire category tree.

One silly problem I ran into was, how do I pass the parameters to the getCategory() call? It took me awhile because I couldn't find it via googling so I had to experiment.

Turns out it's

```
$session = Services_Ebay::getSession($devId, $appId, $certId);
$session->setToken($token);
$ebay = new Services_Ebay($session);
try {
    $args = array('DetailLevel'=>1, 'ViewAllNodes'=>1,
    );
    $result = $ebay->GetCategories($args);
    print_r($result);
} catch (Exception $e) {
    echo "Something went wrong.";
    echo $e;
}
```

If you're going to cut your teeth on the getCategory() call like I did, remember to lift the default 8M memory limit on your script when you dump the entire tree.

So I'm looking forward to experimenting with this API.

Posted by Mark Jeftovic in Hacking tech at 22:21