

Details of a Real Data Breach

It took a breach at Barracuda Networks to prove the necessity of continuous Barracuda Web Application Firewall™ security



By Oliver Wai and Bob Matlow

April 9, 2011

Introduction

In the era of automated Web attacks, the time when an administrator could leave a Web site unguarded for even a few hours has passed into history. On Saturday, April 9, 2011, an unusual set of conditions aligned that resulted in some bad actors successfully breaching a marketing database at Barracuda Networks through an SQL injection attack. A Barracuda Web Application Firewall that had been put into Passive Mode recorded a detailed audit trail of the probe and the subsequent successful attack. This logging gave us the forensic data we needed to quickly analyze the breach, contain the damage and reach out to the people whose names and email addresses had been exposed. Analysis revealed that the attack was most likely launched by “gray hat” hackers with limited criminal intent. This paper will explore how the breach happened, what we learned, and how we unintentionally demonstrated the effectiveness of our own Barracuda Web Application Firewall in preventing further damage and stopping an actual application-layer attack in progress.

Why Bad Actors Attack Web Applications

Web application traffic is designed to move transparently through network firewalls. Traditional layer 4 network firewalls are ineffective in detecting and blocking layer 7 (application layer) attacks. However, many organizations – regardless of size – haven’t recognized that layer 4 security measures are outdated. This leaves these organizations vulnerable to application layer attacks from a variety of bad actors whose motives range from professional organized crime, government-sponsored cyber warfare and corporate espionage to interest-oriented political activists and amateurs seeking “street creds” among a loosely knit community of security hacking enthusiasts. The data breach at Barracuda Networks was most likely performed by “gray hat” not-for-profit enthusiasts looking to score points with their peers. How we came to this conclusion is discussed below.

Regardless of the motive, attacks against Web applications and specifically SQL injection attacks have proven to be the most effective way to penetrate networks for stealing data.

- Web application attacks account for only 54 percent of all data breaches, but 92 percent of stolen records
- SQL injection attacks account for only 25 percent of Web application attacks, but 89 percent of stolen records¹

Conditions at Barracuda Networks that Resulted in a Breach

Barracuda Networks employs redundant security procedures to prevent attacks on Web applications. Despite this, a series of events occurred that resulted in the data breach:

1. An error writing PHP code on our Web Site
2. A planned code vulnerability scan was not performed in a timely manner on the part of the Web site that contained the PHP code error
3. The Barracuda Web Application Firewall responsible for safeguarding the Web site was placed in passive mode through human error during a maintenance window

¹Verizon and US Secret Service Breach Study, 2009

Details of a Real Data Breach

With vulnerable code and the Barracuda Web Application Firewall in passive mode, it was only a matter of time before an attack occurred. Based on the Barracuda Web Application Firewall's logging and reporting features, here is how the attack occurred:

The Attack Timeline

Date/Time	Description
2011-04-10 00:07:59 GMT	First IP address started initial probe of the Web site root page
2011-04-10 00:16:15 GMT	Attack found a vulnerable URL after trying 175 URLs, starts probing database
2011-04-10 03:10:43 GMT	Second IP address started probing the vulnerable URL
2011-04-10 10:10:00 GMT	Attack started trying to retrieve database users
2011-04-10 10:16:00 GMT	Attack gave up on users, tried to retrieve database list and schema
2011-04-10 10:19:00 GMT	Attack started stealing data
2011-04-10 17:30:00 GMT	We discovered the attack
2011-04-10 17:37:00 GMT	We discovered that the Barracuda Web Application Firewall was not in active mode on these services
2011-04-10 17:39:59 GMT	We placed relevant services on the Barracuda Web Application Firewall back into active mode, no further attacks reached server
2011-04-10 21:00:00 GMT	Attacks from these IP addresses stop hitting the Web Application Firewall

Analysis of the Attack

Barracuda Web Application Firewall logs allowed us to determine that our bad actors used two clients to probe and attack the barracudanetworks.com Web site:

The screenshot shows the Barracuda Web Application Firewall log display interface. The interface includes a navigation menu with tabs for BASIC, SECURITY POLICIES, WEBSITES, ACCESS CONTROL, and ADVANCED. Below the menu, there are sub-tabs for Status, Services, Default Security, Certificates, IP Configuration, and Administration. The main content area is titled 'Log Display' and shows a list of logs for the time range 2011-04-09 20:13 to 2011-04-09 20:20. The logs are filtered to show SQL injection attacks on the URL www.barracudanetworks.com. The log entries are as follows:

Time	Client IP	Client Port	Service	IP Port	Action	Follow Up	Severity	URL	Method	Attack	Detail	Rule Type	Rule
2011-04-09 20:19:59.91	115.134.249.15	24858			LOG	None	Alert	www.barracudanetworks.com/hs/custom	GET	SQL Injection in	type="sql-injectio	Global	security-policy
2011-04-09 20:19:59.88	87.106.220.57	49636			LOG	None	Alert	www.barracudanetworks.com/hs/custom	GET	SQL Injection in	type="sql-injectio	Global	security-policy
2011-04-09 20:19:58.98	87.106.220.57	49635			LOG	None	Alert	www.barracudanetworks.com/hs/custom	GET	SQL Injection in	type="sql-injectio	Global	security-policy
2011-04-09 20:19:58.15	115.134.249.15	24855			LOG	None	Alert	www.barracudanetworks.com/hs/custom	GET	SQL Injection in	type="sql-injectio	Global	security-policy
2011-04-09 20:19:58.01	87.106.220.57	49634			LOG	None	Alert	www.barracudanetworks.com/hs/custom	GET	SQL Injection in	type="sql-injectio	Global	security-policy
2011-04-09 20:19:55.01	87.106.220.57	49633			LOG	None	Alert	www.barracudanetworks.com/hs/custom	GET	SQL Injection in	type="sql-injectio	Global	security-policy
2011-04-09 20:19:54.21	115.134.249.15	24850			LOG	None	Alert	www.barracudanetworks.com/hs/custom	GET	SQL Injection in	type="sql-injectio	Global	security-policy
2011-04-09 20:19:54.11	87.106.220.57	49632			LOG	None	Alert	www.barracudanetworks.com/hs/custom	GET	SQL Injection in	type="sql-injectio	Global	security-policy
2011-04-09 20:19:52.91	87.106.220.57	49631			LOG	None	Alert	www.barracudanetworks.com/hs/custom	GET	SQL Injection in	type="sql-injectio	Global	security-policy
2011-04-09 20:19:52.01	87.106.220.57	49630			LOG	None	Alert	www.barracudanetworks.com/hs/custom	GET	SQL Injection in	type="sql-injectio	Global	security-policy
2011-04-09 20:19:51.51	115.134.249.15	24848			LOG	None	Alert	www.barracudanetworks.com/hs/custom	GET	SQL Injection in	type="sql-injectio	Global	security-policy
2011-04-09 20:19:51.11	87.106.220.57	49629			LOG	None	Alert	www.barracudanetworks.com/hs/custom	GET	SQL Injection in	type="sql-injectio	Global	security-policy
2011-04-09 20:19:50.11	87.106.220.57	49628			LOG	None	Alert	www.barracudanetworks.com/hs/custom	GET	SQL Injection in	type="sql-injectio	Global	security-policy
2011-04-09 20:19:50.01	115.134.249.15	24846			LOG	None	Alert	www.barracudanetworks.com/hs/custom	GET	SQL Injection in	type="sql-injectio	Global	security-policy
2011-04-09 20:19:49.21	87.106.220.57	49627			LOG	None	Alert	www.barracudanetworks.com/hs/custom	GET	SQL Injection in	type="sql-injectio	Global	security-policy
2011-04-09 20:19:48.21	87.106.220.57	49626			LOG	None	Alert	www.barracudanetworks.com/hs/custom	GET	SQL Injection in	type="sql-injectio	Global	security-policy
2011-04-09 20:19:48.11	115.134.249.15	24844			LOG	None	Alert	www.barracudanetworks.com/hs/custom	GET	SQL Injection in	type="sql-injectio	Global	security-policy
2011-04-09 20:19:47.31	87.106.220.57	49625			LOG	None	Alert	www.barracudanetworks.com/hs/custom	GET	SQL Injection in	type="sql-injectio	Global	security-policy
2011-04-09 20:19:46.41	87.106.220.57	49624			LOG	None	Alert	www.barracudanetworks.com/hs/custom	GET	SQL Injection in	type="sql-injectio	Global	security-policy
2011-04-09 20:19:46.31	115.134.249.15	24842			LOG	None	Alert	www.barracudanetworks.com/hs/custom	GET	SQL Injection in	type="sql-injectio	Global	security-policy

Details of a Real Data Breach

Using the information reported by the Barracuda Web Application Firewall, we were able to quickly filter and find the corresponding entries on our Web server logs:

```
2011-04-10 03:19:17 GET /ns/customers/customer_verticals.php v=12"%20and%20ascii(substring((database()),13,1))=99%20and%20"x"="x 80 - 87.11
2011-04-10 03:19:17 GET /ns/customers/customer_verticals.php v=11"%20and%20ascii(substring((SELECT%20distinct%20schema_name%20from%
2011-04-10 03:19:18 GET /ns/customers/customer_verticals.php v=12"%20and%20ascii(substring((database()),13,1))=98%20and%20"x"="x 80 - 87.11
2011-04-10 03:19:18 GET /ns/customers/customer_verticals.php v=12"%20and%20ascii(substring((database()),13,1))=97%20and%20"x"="x 80 - 87.11
2011-04-10 03:19:19 GET /ns/customers/customer_verticals.php v=11"%20and%20ascii(substring((SELECT%20distinct%20schema_name%20from%
2011-04-10 03:19:21 GET /ns/customers/customer_verticals.php v=11"%20and%20ascii(substring((SELECT%20distinct%20schema_name%20from%
2011-04-10 03:19:24 GET /ns/customers/customer_verticals.php v=11"%20and%20ascii(substring((SELECT%20distinct%20schema_name%20from%
2011-04-10 03:19:26 GET /ns/customers/customer_verticals.php v=11"%20and%20ascii(substring((SELECT%20distinct%20schema_name%20from%
2011-04-10 03:19:28 GET /ns/customers/customer_verticals.php v=11"%20and%20ascii(substring((SELECT%20distinct%20schema_name%20from%
2011-04-10 03:19:31 GET /ns/customers/customer_verticals.php v=11"%20and%20ascii(substring((SELECT%20distinct%20schema_name%20from%
2011-04-10 03:19:32 GET /ns/customers/customer_verticals.php v=11"%20and%20ascii(substring((SELECT%20distinct%20schema_name%20from%
2011-04-10 03:19:33 GET /ns/customers/customer_verticals.php v=11"%20and%20ascii(substring((SELECT%20distinct%20schema_name%20from%
2011-04-10 03:19:37 GET /ns/customers/customer_verticals.php v=11"%20and%20ascii(substring((SELECT%20distinct%20schema_name%20from%
2011-04-10 03:19:39 GET /ns/customers/customer_verticals.php v=11"%20and%20ascii(substring((SELECT%20distinct%20schema_name%20from%
2011-04-10 03:19:41 GET /ns/customers/customer_verticals.php v=11"%20and%20ascii(substring((SELECT%20distinct%20schema_name%20from%
2011-04-10 03:19:46 GET /ns/customers/customer_verticals.php v=11"%20and%20ascii(substring((SELECT%20distinct%20schema_name%20from%
2011-04-10 03:19:48 GET /ns/customers/customer_verticals.php v=11"%20and%20ascii(substring((SELECT%20distinct%20schema_name%20from%
2011-04-10 03:19:48 GET /ns/customers/customer_verticals.php v=12"%20and%20Length((SELECT%20distinct%20schema_name%20from%20infor
2011-04-10 03:19:49 GET /ns/customers/customer_verticals.php v=12"%20and%20Length((SELECT%20distinct%20schema_name%20from%20infor
2011-04-10 03:19:51 GET /ns/customers/customer_verticals.php v=11"%20and%20ascii(substring((SELECT%20distinct%20schema_name%20from%
2011-04-10 03:19:51 GET /ns/customers/customer_verticals.php v=12"%20and%20Length((SELECT%20distinct%20schema_name%20from%20infor
2011-04-10 03:19:52 GET /ns/customers/customer_verticals.php v=12"%20and%20Length((SELECT%20distinct%20schema_name%20from%20infor
2011-04-10 03:19:53 GET /ns/customers/customer_verticals.php v=11"%20and%20ascii(substring((SELECT%20distinct%20schema_name%20from%
2011-04-10 03:19:53 GET /ns/customers/customer_verticals.php v=12"%20and%20Length((SELECT%20distinct%20schema_name%20from%20infor
2011-04-10 03:19:54 GET /ns/customers/customer_verticals.php v=12"%20and%20Length((SELECT%20distinct%20schema_name%20from%20infor
2011-04-10 03:19:54 GET /ns/customers/customer_verticals.php v=11"%20and%20ascii(substring((SELECT%20distinct%20schema_name%20from%
2011-04-10 03:19:54 GET /ns/customers/customer_verticals.php v=12"%20and%20Length((SELECT%20distinct%20schema_name%20from%20infor
2011-04-10 03:19:55 GET /ns/customers/customer_verticals.php v=12"%20and%20ascii(substring((SELECT%20distinct%20schema_name%20from%
2011-04-10 03:19:57 GET /ns/customers/customer_verticals.php v=12"%20and%20ascii(substring((SELECT%20distinct%20schema_name%20from%
2011-04-10 03:19:57 GET /ns/customers/customer_verticals.php v=11"%20and%20ascii(substring((SELECT%20distinct%20schema_name%20from%
2011-04-10 03:19:57 GET /ns/customers/customer_verticals.php v=12"%20and%20ascii(substring((SELECT%20distinct%20schema_name%20from%
```

Note: The Web server logs use Greenwich Mean Time (GMT) whereas the Web Application Firewall uses Pacific Daylight Time (PDT)

Drilling down into each log entry on the Barracuda Web Application Firewall gave us clues about the attackers and the tools they used in the attack:

Web Firewall log Details		Help
Time	2011-04-09 17:17:18	
Client IP	115.134.249.155	
Client Port	9551	
Service IP:Port	10.8.0.132:80	
Action	LOG	
Follow Up Action	None	
Severity	Alert	
URL	www.barracudanetworks.com/ns/customers/customer_verticals.php?v=11%20and%201=1	
Method	GET	
Attack	SQL Injection in Parameter	
Detail	type="sql-injection-medium" pattern="sql-tautology-conditions-simple" token=" and 1=1" Parameter="v" value="11 and 1=1"	
Rule Type	Global	
Rule	security-policy	
Protocol	HTTP	
Session ID		
User Agent	Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; SV1; .NET CLR 2.0.50727)	
Proxy IP	115.134.249.155	
Proxy Port	9551	
Authenticated User		
Referer		

Details of a Real Data Breach

The first attack started at 5:07pm PDT on April 9. The attackers' IP address 115.134.249.15 resolved to the area of Kuala Lumpur, Malaysia. This log entry confirmed online reports that the attacks originated from Malaysia. We also noticed that the attackers used a modified version of a pentest tool designed by "white hats" to probe Web sites for SQL injection vulnerabilities. This log entry correlated with reports that the hacking team responsible for this attack frequented "white hat" online communities. We also found the same entries on our Web server logs. These log entries let us trace what types of attacks were attempted and which attacks succeeded on our backend systems.

```
ex11041000.log:2011-04-10 00:17:18 GET /ns/customers/customer_verticals.php v=11 80 - 115.134.249.155 Mozilla/4.0+(compatible;+MSIE+7.0;+W
ex11041000.log:2011-04-10 00:17:20 GET /ns/customers/customer_verticals.php v=-9,9 80 - 115.134.249.155 Mozilla/4.0+(compatible;+MSIE+7.0;+V
ex11041000.log:2011-04-10 00:17:22 GET /ns/customers/customer_verticals.php v=11%20and%201=1 80 - 115.134.249.155 Mozilla/4.0+(compatibl
ex11041000.log:2011-04-10 00:17:24 GET /ns/customers/customer_verticals.php v=11%20and%201=0 80 - 115.134.249.155 Mozilla/4.0+(compatibl
ex11041000.log:2011-04-10 00:17:25 GET /ns/customers/customer_verticals.php v=11%20and%20'x'='x 80 - 115.134.249.155 Mozilla/4.0+(compati
```

Note: The Web server logs use GMT whereas the Web Application Firewall uses PDT

It became clear that the first attacker used an automated tool to recursively crawl through the barracudanetworks.com Web site and blindly inject a series of SQL commands against each input parameter to find potential vulnerabilities. The SQL injection tool found the first vulnerability at 5:16pm PDT but continued to probe the Web site. At 8:10pm PDT a second client using the IP address of 87.106.220.57 joined the attack. The second IP address resolved to a server in Germany. It is unclear at this time if the server was a relay point or if it was a second attacker. The Barracuda Web Application Firewall also recorded and logged activities from the second IP address.

Web Firewall log Details	
Time	2011-04-09 20:14:09
Client IP	87.106.220.57
Client Port	49420
Service	
IP:Port	10.8.0.132:80
Action	LOG
Follow Up	
Action	None
Severity	Alert
URL	www.barracudanetworks.com/ns/customers/customer_verticals.php?v=12%20and(select%201%20from(select%20count(*),concat((select%20(select%20concat(0x7e,0x27,Hex(cast(database()%20as%20char))0x27,0x7e)))%20from%20information_s
Method	GET
Attack	SQL Injection in Parameter
Detail	type="sql-injection-medium" pattern="sql-select-command" token="(select 1 from(select count(*)\"concat((select (s, Parameter='v' value='12\" and(select 1 from(select count(*)\"
Rule Type	Global
Rule	security-policy
Protocol	HTTP
Session ID	
User Agent	Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; SV1; NET CLR 2.0.50727)
Proxy IP	87.106.220.57
Proxy Port	49420
Authenticated	
User	
Referer	

The following is the corresponding Web server log:

```
2011-04-10 03:14:11 GET /ns/customers/customer_verticals.php v=12"%20UNION%20ALL%20SELECT%20null,null,null,null,null,null,null,null,null,null,
2011-04-10 03:14:11 GET /ns/customers/customer_verticals.php v=11"%20and%20ascii(substring((SELECT%20distinct%20schema_name%20from%
2011-04-10 03:14:12 GET /ns/customers/customer_verticals.php v=12"%20and(select%201%20from(select%20count(*),concat((select%20(select%2
2011-04-10 03:14:12 GET /ns/customers/customer_verticals.php v=12"%20and(select%201%20from(select%20count(*),concat((select%20(select%2
2011-04-10 03:14:14 GET /ns/customers/customer_verticals.php v=11"%20and%20ascii(substring((SELECT%20distinct%20schema_name%20from%
2011-04-10 03:14:14 GET /ns/customers/customer_verticals.php v=12"%20and%20Length((database()))<32%20and%20"x"='x 80 - 87.106.220.57 Mc
2011-04-10 03:14:15 GET /ns/customers/customer_verticals.php v=12"%20and%20Length((database()))<16%20and%20"x"='x 80 - 87.106.220.57 Mc
```

Note: The Web server logs use GMT whereas the Web Application Firewall uses PDT

From the logs captured in the Barracuda Web Application Firewall, it seems that the attacker used the second client to launch manual attacks against the discovered vulnerability while the primary attack script continued to scan the Web site to find other vulnerabilities. Ultimately, the attackers focused their efforts on a single line of weak code in a peripheral Web page where the input parameters had not been properly sanitized. Here is the pseudo-code of the underlying vulnerability:

```
<?=Foo_Function($_GET['parameter'])?> //Takes user input
```

By not sanitizing the input value, this code error let attackers inject SQL commands into the HTML input parameter to attack the underlying database.

Details of a Real Data Breach

Developers are taught to never trust user inputs; all inputs must be sanitized before sending them to underlying servers. However, you can see from the example above that it is not often obvious to the naked eye that something is wrong with the code. This is why, in addition to defensive coding, Barracuda Networks uses code scanners and our Barracuda Web Application Firewall to guard against possible vulnerabilities. Because of automated attacks, in a Web site of tens of thousands of lines of code, all it takes is a single mistake for an attack to succeed. We have since added a line of code that sanitizes inputs on the affected page to protect against future attacks:

```
$parameter = @is_sanitized($_GET['parameter']) ? $_GET['parameter'] : 0;  
<?=Foo_Function( $parameter)?>
```

From Vulnerability to Breach

After the attackers found the vulnerable page, they attempted to steal database user accounts. Over the next ten hours, the attackers tried several methods to break into the underlying database, but failed each time. At 3:06am PDT, the attackers changed strategy to focus on the underlying database schema. This proved to be the correct strategy. By 3:19am PDT the attackers had stolen the first email accounts.

A Barracuda Networks' systems administrator discovered the breach at 10:30am PDT and re-enabled the Barracuda Web Application Firewall to active mode at 10:39am PDT. The Barracuda Web Application Firewall immediately blocked all subsequent attacks from the 115.134.249.15 IP address. The attackers continued to cycle through attacks against the remaining Web pages for the next few hours with the Barracuda Web Application Firewall blocking all of the attacks. This attack profile supports our conclusion that the attackers used an automated pentest tool to blindly inject SQL commands. In all, the attackers sent a total of 110,892 SQL injection commands from both attacking IP addresses against 175 URLs at a rate of 42 per minute.

In tracing the Web Firewall and Access logs on the Barracuda Web Application Firewall, we determined that the attackers compromised a marketing database and stole two sets of records. A total of 21,861 names and emails were stolen from the database. Since there were a number of duplicates in the two sets and many of the entries were from users who are no longer with their original organizations, the number of affected users is substantially lower than the total stolen records.

Any breach is a serious issue. Although the team executing this attack appears to be fairly benign, data breaches like this one have been used to enable spearphishing attacks against affected users. We have already reached out to affected users with documentation and have advised of possible precautions they may wish to take. We believe that the users affected by the breach are at minimal risk. We do not store any sensitive information in our marketing database other than names and email addresses. Moreover, since Barracuda Networks primarily uses this customer data to send emails on upcoming events, Webinars, or corporate news updates, the risk of spearphishing is low as the all of communications are one-directional and informational in nature. Finally since most users are existing Barracuda Spam & Virus Firewall customers, the vast majority of potential spam would likely be blocked.

Conclusion

In what would be called a before-and-after experimental design — had we been doing an intentional field experiment — we found that our Barracuda Web Application Firewall provided complete protection against SQL injection attacks for our Web site which contained a PHP code vulnerability, until we unintentionally turned off the active protection mode. Upon detection of the attack, resetting the Barracuda Web Application Firewall to active protection mode stopped subsequent attacks within seconds. The Barracuda Web Application Firewall's logging and reporting capabilities helped in forensic analysis of the attack and in simplifying responses for contacting the affected parties as well as handling inquiries from the media, research and analyst communities. While careful coding and vulnerability testing are important parts of current network defense postures, Barracuda Web Application Firewalls should be the first line of defense against layer 7 attacks.

Barracuda Web Application Firewall Features

Comprehensive protection against widely used application layer attacks:

- SQL injection flaws
- Application denial of service
- Cross Site Scripting (XSS)
- Malicious probes/crawlers
- OS command injections
- Cookie/session tampering
- Site reconnaissance
- Path traversal
- Session hijacking
- Outbound filtering to prevent information leakage (DLP)

Barracuda Web Application Firewalls also have features that organically optimize network performance:

- SSL Offloading
- SSL Acceleration
- Load Balancing

Contact a representative at Barracuda Networks today at: 1-888-ANTI-SPAM

or visit: <http://www.barracudanetworks.com/ns/products/web-site-firewall-overview.php>

Also available as virtual appliances.

For questions about the Barracuda Web Application Firewall, please visit <http://www.barracuda.com/waf> or call Barracuda Networks for a free 30-day evaluation at 1-888-ANTI-SPAM or +1 408-342-5400. For more information on our other security and productivity solutions, please visit <http://www.barracuda.com/products>.

About Barracuda Networks Inc.

Barracuda Networks Inc. combines premises-based gateways and software, virtual appliances, cloud services, and sophisticated remote support to deliver comprehensive content security, data protection and application delivery solutions. The company's expansive product portfolio includes offerings for protection against email, Web and IM threats as well as products that improve application delivery and network access, message archiving, backup and data protection.

Coca-Cola, FedEx, Harvard University, IBM, L'Oreal, and Europcar are among the more than 130,000 organizations protecting their IT infrastructures with Barracuda Networks' range of affordable, easy-to-deploy and manage solutions. Barracuda Networks is privately held with its International headquarters in Campbell, Calif. For more information, please visit www.barracudanetworks.com.