

Miracle on Thirty-Hack Street - Answers and Winners

Greetings, challenge fans! It's time (at long last) to announce the WINNERS for our holiday-themed challenge, Miracle on Thirty-Hack Street. I've gotta say, we received a huge number of high-quality responses. KJ0 (one of my nicknames for my challenge co-author, Kevin "Frickin'" Johnson) and I felt kinda like we were in the scene from the movie where they dump all the mail on the judge. I apologize for not getting these answers done sooner, but a family medical emergency in January and February consumed much of my attention those months. But, we're back in action and ready to roll.

Dedicated little elves that we are, Count Kevula and I read every last word of every entry. Actually, we read the first word, too. Oh, and all of the ones in between. (Well, except for one entry, in which we read only every other word. It was kinda confusing, quite honestly. But, since that submission came from Don Donzal, who is ineligible to win, we figured it was OK to skip those words.)

--Ed Skoudis

EthicalHacker.net Challenge Master

Author of Counter Hack Reloaded, Co-Founder, InGuardians, SANS Instructor

[del.icio.us](#)

[Discuss in Forums {mos_smf_discuss:December 2009 - Miracle on Thirty-Hack Street}](#)

SANS vLive SEC 504 GCIH w/ Skoudis & Strand Starts May 25

SANS vLive SEC 617 GAWN w/ Josh Wright Starts May 19

Advertisement

Miracle on Thirty-Hack Street

Answers & Winners

By Ed Skoudis and Kevin Johnson

Anyway, first off, let's review some answers that the Kevinator and I prepared for you.

1) "What is the name of the following mathematical property? If $a=b$ and $b=c$, then $a=c$."

This is the transitive property of equality. On the surface, this may seem like an oddball question. But, Kev-mo and I planted it as a big, whopping hint. We were hoping you'd map the concept of transitive equality into transitive trust. While highly convenient, transitive trust can cause serious security holes when not applied very carefully. For many social network systems, transitive trust is implicit among friends and friends of friends and friends of friends of friends of friends and... oh heck:

```
C:\> FOR /L %i in (1,0,2) do @echo friends of
```

In other words, if Kevtastic is friends with Ed, and Ed is friends with Larry, then Kevtastic can typically see information from Larry's account. Like that overly friendly girl in high school that your parents warned you about, a promiscuous friender can spread stuff among a whole community of people that is best left private. Let's see how we can use this concept to answer the questions below.

2) What FQL query or API call can be used to retrieve information about vacations from Kris Cringle's (uid

10000565751882) Facebook account?

There are a number of ways to get information about Kris Cringle. The first step is to friend Fred Gailey, the promiscuous friender in our little story. Because Facebook's terms of service didn't allow us to write a script to automate the accepting of friends, Kevitation and his dear wife actually played the part of Fred Gailey. Throughout the challenge period, they'd login as Fred every hour or so and blindly accept friend requests from all ya'll. And, for all of you who tried to friend Kris, you aren't able to do so directly since he has lost his account password, so he couldn't really login and accept your request. Instead, after Fred accepted your request, you could ride on the train of transitive trust to get access to Kris's private account information.

After friending Fred, you could visit the tools section from <http://developers.facebook.com/tools.php>, which allows you to try out FQL queries or API calls. This super-awesome page is designed so that Facebook application developers can test out their API calls and query fu, but Kevithan and I like to use it for ganking data out of Facebook accounts in a flexible fashion (always assiduously following the terms of service carefully explained to us by our lawyer via a fascinating session of interpretive dance). Anyway, that's just what we wanted you to do (ganking the data from Kris's account, not the interpretive dance part).

Kevticipation and I weren't expecting you to be Facebook developer ninjas or ultimate street fighters or anything. Instead, we were hoping you'd do a little research in the documentation that Facebook so helpfully makes available. Some of the best docs about FQL and the Facebook API are available at:

* <http://wiki.developers.facebook.com/index.php/FQL>

* http://wiki.developers.facebook.com/index.php/API#Data_Retrieval_Methods

Note that Kevikinesis and I specifically asked about a "FQL Query or API call". As it turns out, there were (at least) two different ways of getting the data you needed from Kris's account. One method relied on FQL Queries. The other relied on API calls.

Let's look at the FQL method first. First, you can run the following FQL query to retrieve lots of information about Kris:

```
SELECT uid, first_name, last_name, name, pic_small,  
pic_big, pic_square, pic, affiliations,  
profile_update_time, timezone, religion, birthday,  
birthday_date, sex, hometown_location, meeting_sex,  
meeting_for, relationship_status, significant_other_id,
```

```
political, current_location, activities, interests,  
is_app_user, music, tv, movies, books, quotes, about_me,  
hs_info, education_history, work_history, notes_count,  
wall_count, status, has_added_app, online_presence,  
locale, proxied_email, profile_url, email_hashes,  
pic_small_with_logo, pic_big_with_logo,  
pic_square_with_logo, allowed_restrictions, verified,  
profile_blurb, family, family, website, is_blocked FROM  
user WHERE uid="100000565751882"
```

This query will return quite a bit of information including a count of various posts to the account. For example, we get back the count of notes posted and wall updates. Good penetration testers (always operating within the rules of engagement and meticulously following the Facebook terms of service) should know that Facebook account notes often contain quite a bit of information of interest, so we will retrieve those.

When we look at the `note_count`, we see that Kris has posted one note (Kevinstein and I didn't want you to have to dig through a bunch of junk to find it). We can then grab that note by performing the following FQL query:

```
SELECT uid, note_id, created_time, updated_time, content,  
title FROM note WHERE uid="100000565751882"
```

This query returns the contents of Kris' note regarding his vacations. He refers to a bunch of different cities and countries that he has visited, with some special remarks about "norway". As it turns out, that word, with its glorious lower-case n, was his passphrase. Why norway? I dunno... it just felt right. I've made a lot of friends from Norway over the past few years. One time, I went out for beers with a chap from there who had done very well in the Capture the Flag game from my SANS class. I asked him what in his background allowed him to do so well. He told me that it was the fact that he was from Norway. He explained, "Ed, it's so quiet and cold up there, we can focus on getting really good at a few things. Folks from Norway tend to be really good at drinking, making love, and hacking." I can confirm the first and third of the items in his list, but have no personal experience about the second one. Anyway, given that description of the fine people from Norway, it seemed like the kind of place that Santa would like.

Some folks who played the challenge wrote nifty little shell scripts around GnuPG to take each word from a list they compiled from Kris' note, and use it as a passphrase to try to decrypt the attachment. You gotta like a little Command Line Kung Fu now and then. Many others just manually typed in a few words (or

even got lucky and tried "norway" on their first shot). Both approaches (shell fu or just typing) would yield the answer pretty quickly.

Alternatively, you could have used the API instead of the FQL query. For this approach, we could first call `users.getinfo()` to retrieve the same information as the first query above. The API call would look like the following:

```
$facebook->api_client-  
>users_getInfo(100000565751882,'uid, first_name,  
last_name, name, pic_small, pic_big, pic_square, pic,  
affiliations, profile_update_time, timezone, religion,  
birthday, birthday_date, sex, hometown_location,  
meeting_sex, meeting_for, relationship_status,  
significant_other_id, political, current_location,  
activities, interests, is_app_user, music, tv, movies,  
books, quotes, about_me, hs_info, education_history,  
work_history, notes_count, wall_count, status,  
has_added_app, online_presence, locale, proxied_email,  
profile_url, email_hashes, pic_small_with_logo,  
pic_big_with_logo, pic_square_with_logo,  
allowed_restrictions, verified, profile_blurb, family,  
family, website, is_blocked');
```

This blob would again return quite a bit of information including the `notes_count` of 1. We would then retrieve the note information by calling `notes.get()`. The code to do this looks like:

```
$facebook->api_client->notes_get(100000565751882,");
```

As with the FQL, this call returns the contents of the note, with "norway" being a prominent word.

3) What Facebook privacy setting allowed this data leakage? What is the default value of this setting?

The privacy option within Facebook that allow such access is the "Friends of Friends" setting on the note when it was posted. When the challenge started, the default was for just friends of friends to see notes. About 3 days after we posted the challenge, Facebook changed their privacy defaults. Kevinopolous and I were really concerned when we first heard that Facebook had changed its privacy policy. We had thought that they might have completely blown up our challenge by tightening things up.

But we shouldn't have worried at all. The general drift of privacy settings in social networks is inexorably toward more and more sharing and less and less privacy, until the world is nothing but a big ball of public information and the concept of privacy is dumped in the trashbin of history. Or sumpin's like that. Anyway, after Facebook's default policy change right in the middle of our challenge, the default for a note posting is "Everyone". Yowza! Our challenge would still work, and the default was even weaker.

4) What is the text from the decrypted message from the Judge?

By decrypting the PDF with the passphrase of "norway" we get the original PDF which appears to be a Christmas letter from the Judge to Santa.

December 9, 1901

Dear Mr. Santa,

My mom asked me to write you this letter with my Christmas Wish List. I've always wanted a Righteous Bison Indivisible Particle Smasher for Christmas. I will use it for good - I promise! Here is a picture:

I've tried to be a very good boy all year.

Thank you,

Henry X. Harper

Age 10

P.S. My middle name is "X-mas"!

So, what is this note all about? Well, Kevilicious and I wrote it to try to instill a backstory to the whole challenge. You see, in our formulation, the judge (who was named Henry X. Harper in the movie) had written a letter to Santa way back in 1901. If you check out the IMDB entry for the original 1947 movie (<http://www.imdb.com/title/tt0039628/>), you can see that the character's name was really Henry X. Harper. When writing the challenge, Kevquistador and I were musing about the middle initial X., and decided that it would be cool to have it expand to X-mas. Furthermore, note that the letter was written in 1901. That is based on the fact that Gene Lockhart, the actor who played Judge Harper, was born in 1891 (<http://www.imdb.com/name/nm0516876>). We wanted him to have written the letter at the age of 10, so 1901 was the way to go. One wonders what challenge writers did before IMDB came around.

In the letter itself, young Henry asks Santa for a Righteous Bison Indivisible Particle Smasher for Christmas. What the heck is that? Why, it's a steampunk raygun, one of Dr. Grordbort's fine weapons available for purchase from Weta (director Peter Jackson's special effects group, which sells props from movies and other cool stuff at <http://www.wetanz.com/the-righteous-bison-indivisible>). I bought one of these for my steam punk office redesign, in fact, and simply love it. I have it hanging above the door in the Secret Room over the passage that leads back into the Main Laboratory. The photo you see in the PDF is actually a shot taken from within the Secret Room looking out into the Main Laboratory with the bookcase hide-a-door (<http://www.hideadour.com/>) opened.

So, we wrote the challenge with the idea that Judge Harper had an unrequited longing for a raygun that he had asked Santa for 46 years earlier. To prove that Kris was the real Santa, the Judge simply asked him again for the gift he had requested so many years before.

What does this have to do with the challenge? Well, everything and nothing. Kevilingus and I like to put in a good back story to these things because it makes them seem more real, gives them a larger surface for people to explore, and it's just plain fun. Oh, and we're freaks. Can't forget that. Especially Kevington.

Bonus Question: What other information can you pull from the Kris Cringle Facebook account (uid 100000565751882)?

Whenever we write these challenges, we like to have an open-ended question at the end, so that folks can feel free to explore deeper on their own, and help differentiate their answers from others. This question is based on the fact that we can get quite a lot of information from user accounts such as their wall postings, notes, photos, and captions. We are also able to retrieve lists of their friends and some information about those friends. Although Kris has not filled in much beyond what we have already retrieved above, we are able to get the list of albums Kris has set up and the photos within those albums. We tried to populate that list with some fun pictures of Santas from around the world.

So, as you can see, the whole reason we wrote this challenge was to underscore the privacy concerns associated with social networking sites, and the ease with which large amounts of sensitive data could be harvested using FQL and/or the Facebook API. We use these tools all the time while conducting reconnaissance during penetration tests (always done with permission, of course, and following the site's terms of service).

As for our own usage of social networking sites, both Kevspastic and I treat all data we put on the sites as public, and we recommend that you do the same. The security and privacy of data on today's social networking sites is questionable at best, completely broken at worst. You should not expect a site that is designed to share information far and wide (between users as well as with advertisers) to protect your information, even if it seems like they might be pretending to pay lip service to some abstract concept of privacy. Instead, follow the old adage: Only put information about yourself on your social networking sites you'd want published on the front page of the morning newspaper.

And now... on to our winner announcements!

First off, please remember that we had so many very high quality entries, it was very difficult to pick the best one. Even those folks who garnered an honorable mention did fantastic work. So, don't be disappointed if you got an honorable mention. You still did great! I sometimes get kinda sad when people tell me that they only got an honorable mention. Hey, celebrate! Trumpet it from the mountaintops. Put it on your resume! Tell everyone that Kevspedia and I said you are a genius. Also, there are some really solid answers even among those that didn't get an honorable mention. So, please don't grumble if you didn't win or get an honorable mention. Kevauditory has very sensitive ears and can actually hear you. Anyway, without further adieu...

Technical Winner

Our Technical Winner is... Mark Baggett! His answer was awesome, and he really dug into the analysis, and tested his theories about why the transitive trust was working the way that it was. He also wrote a bit of Perl to use the Facebook API. Please do read his answers (link coming soon).

Technical Honorable Mentions go to:

- Brandon Knight
- Mr. Long
- Todd Last
- Wayne Dawson
- Dmitry Akselrod
- Kevin White
- Ajit Gaddam

- Mark Hillick
- Daniel Kennedy

Creative Winner

Our next category is Creative Winner. This is really my favorite one, because Kevsadorian and I get to watch you expand upon our little tale in fun and interesting ways. You really should read the winning answer here, plus some honorable mention entries for a good chuckle along with some great technical insight.

Envelop please…

Our Creative Winner is… Tor Inge Skaar! He grew the back story to our challenge in a quirky and fun way, and offered up some really slick command-line kung fu with his answer. You really should read his answers (link to answer coming soon).

Creative Honorable Mentions go to:

- Suzie Walker (nice name… sounds kinda familiar): Suzie built a Facebook application in her answer.
- Timothy Everson: This answer includes a description of Walla Wall Washington (www) telling us where those strange town names come from and their tie-in to Santa's Elves. Kevogany literally laughed out loud as he read the whole thing.
- John Jarocki: John provides some really good narrative to the backstory of the unfortunate Rudolph/Grandma incident. And, he managed to work in the phrase "Turn a buck" into his answer!
- Eugenio Delfa: This answer included a video showing how its author approached the problem, describing the various steps in formulating a solution. It is the first ever entry I've seen with a Vimeo video for any of these challenges, and it was very well done.

Random Draw Winner

Now, on to our final category… the random winner. Random number provided courtesy of my Windows box:

```
C:\> set /a %random%%100 + 1
```

That command gives us a random number between 1 and 100, which I'll apply modulo style to our entries.

And, the winner is... Number 34! Woohoo! Congrats Number 34.

Uh... who was Number 34?

(/me frantically counts through the entries)...

Ahem... and, the Random Draw winner is: John Poulin

Congrats to all our winners. Great job folks.

Thank you--

--Ed and Kevipendicitis.