What Happened Part 2 (2013)

The Questions

Upon entering their emails into the website, the solvers were given a set of test questions, and requested not to publish them. There were 19 questions in total, with three different types. Here are the screenshots of questions as they were on .onion site: http://imgur.com/a/YFA1a

The first type of question gave a statement and then a multitude of answers, which were:

- True
- False
- Indeterminate
- Meaningless
- Self-Referential
- Game Rule
- Strange Loop
- None of the above

These statements were the following:

There is no truth

What you are is more important than what you do

You cannot step into the same river twice

Observation changes the thing being observed

This sentence is false

I am the voice* inside my head (You undoubtedly just thought "I don't have a voice inside my head." *That* is the voice the question is referring to)

Disregarding color blindness, any arbitrary color looks the same to all people

If A is not true, then it must be:

1 = 0.9 recurring

People who only study material *after* a test do better than those who do not study at all

Grass is only green due to a relationship between the grass, the light and your mind

All things are true

We get hundreds of millions of sensations coming into our minds at any moment. Our brain cannot process them all so it categorises these signals according to our belief systems. This is why we find evidence to support our beliefs and rarely notice evidence to the contrary.

The second type of question included an input box with a question. These questions were:

What does the word 'it' refer to in this sentence: It is dark outside?

The mathematical operation known as addition is modeled after what?

Explain, in your own words, what mathematical operation is relied upon for the security of Shamir's Secret Sharing Scheme?

Name similarities between the concept and reality of the 'News Feed' on Facebook?

In the programming language of your choice, write a function that returns the value 3301.

The final type of question only appeared once, and it had different radio buttons to the first type. This question was:

Two people are standing by a lake. One says, "That's a lovely reflection in the water." The other says "I see no reflection, but it's a fascinating assortment of fish, plants and rocks within the water."

Which one is lying?

The answers to this question were:

- The one who sees the reflection
- The one who sees the fish
- Neither
- Both

It has been noted that the abstractness of these questions is very similar to the questions that Google supposedly asks its interviewees for serious roles at the company. They can supposedly be used to determine a person's personality and type.

Each question in the test was timed to prevent one from externally researching questions, and the questions chosen were in a random order from the above pool.

This page also saved two cookies on the user's computer, which were:

- 167=6941f707ff39d259ff71657a79cb6b54c184d2f0455810109c1a960860bde0e6;
- 761=7bc1e7805ccfa518920f0d94fc4e8f7dbd83287a03b337b89109cd2287befae5;

Note that 167 and 761 are ~~palindromic~~ EMIRPS, primes which appeared earlier in the puzzleset (*The Instar Emergence* song is 2:47 long, or 167 seconds. Name of the file: 761.mp3.

> *An **emirp** (prime spelled backwards) is a prime number that results in a different prime when its digits are reversed.*
> *A **palindromic prime** (sometimes called a **palprime**) is a prime number that is also a palindromic number.*
> *A **palindromic number** or **numeral palindrome** is a number that remains the same when its digits are reversed.*
> *Palindromic prime used in puzzle was one in twiter account: https://twitter.com/1231507051321*

The Servers

After completing the test each solver was sent the following email to the address they had indeed yes inputted. Please note that the GPG signature has been removed, but multiple sources have confirmed that they received this email.

```
In the programming language of your choice build a TCP server
that implements the protocol below.  The server code must be
written by you and you alone, although you are free to use any
modules or libraries publicly available for the selected
programming language.

Once you have done this, make it accessible as a Tor hidden
service.  Then provide us with the onion address and port
via a GPG-encrypted email to this address.

You have until 0:00 UTC on 3 Feb, 2013.  Any emails received
after that time will be ignored.

Good luck.

3301

=====================================================================


1. INTRODUCTION

   The TCP server MUST listen on an arbitrary port, and send and
   receive plain text with lines separated by <CRLF> (representing
   a carriage return followed by a line feed).  The TCP server MUST
   disregard the case of input.

   In the examples below, lines sent by the server will be preceded
   with "S:" and lines sent by the client will be preceded by "C:"

   Each message sent by the server MUST conform to the format:

      [CODE] [RESPONSE NAME] [RESPONSE (optional)]<CRLF>
```

Where [CODE] and [RESPONSE NAME] is one of:

CODE	RESPONSE NAME
00	Welcome
01	Ok
02	Error
03	Data
99	Goodbye

2. PROCEDURES

a. Remote Connection

Upon receiving a remote connection, the server MUST greet the client with a 00 WELCOME message. The RESPONSE of a welcome message MAY contain arbitrary text. The arbitrary text MUST at the very least contain the name of the programming language used to implement the server.

Upon receiving a 00 WELCOME message, the client may begin initiating procedures.

Example:

S: 00 WELCOME [ARBITRARY RESPONSE TEXT]<CRLF>

b. RAND [n]

Upon receiving a "RAND" request by the client, the server will first send a 01 OK response, and will then provide the client with [n] cryptographically random numbers within the range of 0-255. Each number MUST be followed by <CRLF>. After the last number has been sent, the server MUST send a dot (.) on a line by itself.

Example:

C: RAND 3<CRLF>
S: 01 OK<CRLF>
S: [first random number]<CRLF>
S: [second random number]<CRLF>
S: [third random number]<CRLF>
S: .<CRLF>

c. QUINE

Upon receiving a "QUINE" request by the client, the server will first send a 01 OK response, and will then provide the client with a quine in the programming language used to implement the server. This quine does not have to be original. After the last line of code has been sent, the server MUST send a dot (.) on a line by itself.

Example:

C: QUINE<CRLF>

S: 01 OK<CRLF>
S: [quine code]<CRLF>
S: .<CRLF>

d. BASE29 [n]

Upon receiving a "BASE29" request by the client, the server will send a 01 OK response followed by the number [n] converted into its base 29 representation.

Example:

C: BASE29 3301<CRLF>
S: 01 OK 3QO<CRLF>

e. CODE

Upon receiving a "CODE" request by the client, the server will send a 01 OK response followed by its own source code. After the last line of code has been sent, the server MUST send a dot(.) on a line by itself.

Example:

C: CODE<CRLF>
S: 01 OK<CRLF>
S: [Server Source Code]<CRLF>
s: .<CRLF>

f. KOAN

Upon receiving a "KOAN" request by the client, the server will send a 01 OK response followed by a koan. After the last line of the koan, the server MUST send a dot (.) on a line by itself.

Example:

C: KOAN<CRLF>
S: 01 OK<CRLF>
S: A master who lived as a hermit on a mountain was asked by a<CRLF>
S: monk, "What is the Way?<CRLF>
S: "What a fine mountain this is," the master said in reply<CRLF>
S: "I am not asking you about the mountain, but about the Way.<CRLF>
S: "So long as you cannot go beyond the mountain, my son, you<CRLF>
S: cannot reach the Way," replied the master<CRLF>
S: .

g. DH [p]

Upon receiving a "DH" request by the client, the server will proceed to perform a Diffie-Hellman key exchange using [p] as the prime modulus. The server will then select a base [b] to use in the protocol, as well as its secret integer. The server will then compute its exponent result [e] as specified within the Diffie-Hellman key exchange protocol.

```
The server MUST then respond with a 01 OK response followed by the
selected base [b] and computed exponent [e] separated by white space.

The client MUST respond with its exponent result [e2], and the client and
server will follow the rest of the Diffie-Hellman key exchange protocol.

The server MUST then compute the resulting secret key, and provide it
using 03 DATA [k].

Example:

    C: DH 23<CRLF>
    S: 01 OK 5 8<CRLF>
    C: 19<CRLF>
    S: 03 DATA 2<CRLF>


j. NEXT

Upon receiving a "NEXT" request by the client, the server will respond
with 01 OK and then listen for text data to be provided by the client.
The client will send a dot (.) on a line by itself after the last line
of text.  The server MUST record this.  This data will be the next set
of instructions.  Once the data is received the server will respond
with 01 OK.

Example:

    C: NEXT<CRLF>
    S: 01 OK<CRLF>
    C: -----BEGIN PGP SIGNED MESSAGE-----<CRLF>
    C: [MESSAGE CONTENTS]<CRLF>
    C: -----END PGP SIGNATURE-----<CRLF>
    C: .<CRLF>
    S: 01 OK<CRLF>


i. GOODBYE

Upon receiving a "DH" request by the client, the server MUST respond with
99 GOODBYE and then gracefully close the connection.

Example:

    C: GOODBYE<CRLF>
    S: 99 GOODBYE<CRLF>
```

The solvers began work on their TCP server programs and submitted them by the deadline presented in the email. An example server coded in Go is here, and an example server coded in python is here.

There was no response until two weeks later, when finally the TCP servers were pinged.

A log of one of the server testings is below:

```
2013/02/25 14:32:01 server is running under address [::]:3307
```

```
2013/03/03 10:57:48 got connection from 127.0.0.1:42483
2013/03/03 10:58:05 executing 'rand 3' for 127.0.0.1:42483
2013/03/03 10:58:09 executing 'rand 3' for 127.0.0.1:42483
2013/03/03 10:58:18 executing 'rand 0' for 127.0.0.1:42483
2013/03/03 10:58:29 executing 'rand 1' for 127.0.0.1:42483
2013/03/03 10:58:56 executing 'quine' for 127.0.0.1:42483
2013/03/03 10:59:10 executing 'base29 1033' for 127.0.0.1:42483
2013/03/03 10:59:14 executing 'koan' for 127.0.0.1:42483
2013/03/03 10:59:16 executing 'koan' for 127.0.0.1:42483
2013/03/03 10:59:18 executing 'koan' for 127.0.0.1:42483
2013/03/03 10:59:21 executing 'koan' for 127.0.0.1:42483
2013/03/03 10:59:28 executing 'dh 3301' for 127.0.0.1:42483
2013/03/03 10:59:56 executing 'dh 3301' for 127.0.0.1:42483
2013/03/03 11:00:29 executing 'dh 3301' for 127.0.0.1:42483
2013/03/03 11:00:58 executing 'next' for 127.0.0.1:42483
2013/03/03 11:01:11 executing 'dh' for 127.0.0.1:42483
2013/03/03 11:01:18 executing 'goodbye' for 127.0.0.1:42483
2013/03/03 11:01:18 closing connection to 127.0.0.1:42483
```

The End?

There were reports that stated they received a message similar to the 'leaked email' from 2012's puzzle. These reports cannot be confirmed as no email was leaked.

It was after this that everything went silent, again.

Hopefully only for another year this time...