Problem Set 1 Solutions

These problems (and all future ones) are graded on a zero, check{plus/minus} basis. A zero means the solution was simply wrong. A minus means that it was lacking in relevant information, while a check means it was a good answer. A check-plus means that you have demonstrated a more in-depth understanding or have done a very good job at answering the question.

These were then summed up (this particular problem set was out of 9), and the total was written on the first page.

Problem 1-1. Digital Postage

William Chuang
Mark Silis
Serena Chan

A: What features do users want?

saves time	quick purchasing and printing of stamps
ease of use	simple software, few steps, multiple letter types
saves money	get exact postage, instead of having to deal with multiple stamp prices
receipts	to track usage (for personal records)
cost	low initial cost, and low overhead stamp cost
reliable	few failures

Essentially users care about a few basic things – if the product is reliable, easy to use, and cheap. Some users may also want to track their usage for billing purposes.

B: What protections do users want?

privacy	the Post Office should not track a user's mailing habits
security	other users/attackers cannot steal credit card or bank account information
refunds	if the printer fails/jams or the software fails, or the dog eats the Security Device

When it comes to technology, users also are interested in privacy and security. They'd also like to know that they won't be billed if something fails, say if their printer jams or their letter indiciums are unreadable due to rain or snow.

C: What features does the US Post Office want?

automation	removes need for more personnel
low maintenance	requires few repairs and usage instruction

The US Post Office sorts, routes, and delivers mail. This is an ideal job for automation, which it realized a decade or two ago. The conversion to a mostly digital system precedes a fully digital future system, which theoretically should cut down on mistakes (heh).

The E-Stamp system should be integrated into the US Post Office's existing sorting/routing hardware. It should also be scalable to accomodate heavier mail loads.

D: What protections does the US Post Office want?

payment	ensure all digital postage is paid for
fraud control	Detect fake or repeated usage of a digital postage stamp
audit trail	log all scanned digital stamp "barcodes" (for refunds, tracking, checking stamp uniqueness, etc)

The US Post Office provides a service (mail delivery) for a fee. Since it's a private entity now (after the US government spun it off as a private company [though still under federal regulation]) the Post Office wants to make sure that it actually gets paid to deliver each piece of mail – and doesn't get defrauded by fake or repeated usage of digital stamps.

As a side effect of this, the US Post Office needs an "audit trail" – a database of all the used digital stamps, so they can check for duplicates.

E: What features is E-Stamp missing?

If we examine the E-Stamp product we can see that it fulfills these requirements. However, in some cases it doesn't have the flexibility that some users may desire:

Mass Mailings prepaid postage: Convenience is an issue for some users, particularly those who send out mass mailings. These customers would like to be able to pre-print large quantities of digital postage stamps and then address them to anyone. Unfortunately for them, E-Stamp encodes the recipient's address in the digital postage stamp (the "indicium") and the US Post Office sorting/routing hardware actually verifies this information.

International Mail: E-Stamp doesn't provide a way to encode foreign mailing addresses, and most foreign mail systems can't handle indicia yet. This limits its use to the US (and probably Canada).

Possible Problems: The E-Stamp software actually attempts to verify the recipient's postal address against a database of all US addresses (on an included CD). This could cause update issues/problems – say if a new ZIP code was created, streets renamed, cities rezoned, etc – update distribution and handling of errors in the database are the major problems.

Apparently E-Stamp and the US Post Office decided this was useful as it would allow E-Stamp to print the 5+4 ZIP codes for better routing.

Problem 1-2. Break the Two-Time Pad

Chris Peikert
Brandon DuRette
Anthony Young-Garner
Roger Dingledine
Nick Mathewson
Brian Sniffen

Cryptanalysis reveals that Ben Bitdiddle has been been misusing a one-time pad in an attempt to conceal his furtive poetical interests. The two texts are the last stanzas of *The Road Not Taken* by Robert Frost and *Do Not Go Gentle into that Good Night* by Dylan Thomas.

```
I shall be telling this with a sigh
Somewhere ages and ages hence:
Two roads diverged in a wood, and I--
I took the one less traveled by,
And that has made all the difference.
```

and

```
And you, my father, there on the sad height,
Curse, bless, me now with your fierce tears,
I pray. Do not go gentle into that good night.
Rage, rage against the dying of the light
```

The key is one of the following sequences. (It is not possible to determine the key uniquely, since it is not possible to definitely match either plaintext to either ciphertext.)

```
6b 08 1c 1d 44 2c 4c 78 55 2f 50 01 02 15 7b 5f 35
62 54 20 42 26 0c 03 05 39 44 2b 07 26 15 4d 08 6b
19 3a 7c 4d 17 5c 27 57 1f 3f 58 7d 16 23 22 20 4b
36 1a 17 22 72 02 15 25 5f 26 08 6c 3a 34 2b 5f 51
62 31 5f 26 54 0c 69 59 39 10 17 71 62 60 0d 13 15
1e 4c 51 0b 07 65 3e 20 61 63 45 66 70 7c 7d 7d 71
2a 36 0d 1d 28 01 5d 14 62 0c 5e 5a 07 6c 00 68 61
10 19 62 25 2b 7c 3f 6f 71 0f 24 58 35 0f 12 71 71
05 24 4b 3a 2e 7f 29 51 0b 6f 68 15 04 3a 41 55 5b
57 3c 45 0d 11 1c 43 06 20 77 29 30 77 61 0d 3c 50
42 09 42 33 0a 23 6a 48
```

```
63 46 0b 55 5c 2f 55 74 17 27 09 55 01 18 63 5e 3e
77 58 74 5e 27 1a 51 17 70 5f 2d 07 33 5d 5b 41 7f
10 54 0f 4a 1f 50 37 57 0e 61 37 1e 02 36 34 36 47
77 16 1f 67 60 16 5c 76 12 2b 4d 6c 36 26 31 22 6c
61 36 5f 2d 54 18 7f 0a 7f 1d 1b 75 64 77 4a 02 14
5f 57 4c 07 6c 0c 69 3f 7c 66 10 68 31 56 76 7d 56
68 6f 27 33 67 55 55 1e 67 58 46 57 42 25 01 72 6b
10 01 6f 37 2c 7c 2c 72 7f 1d 61 5a 39 0c 5a 67 26
```

```
23 7c 6b 33 2f 73 7d 4b 0b 7c 2d 5d 04 2e 00 51 54
40 2d 45 18 15 15 43 16 31 7b 67 33 3e 68 0d 79 56
4f 02 01 3a 4d 64 22 1c
```

We made some observations before starting our guesswork on the ciphertexts.

- Guessing a particular character M_i^1 in message 1 yields a corresponding guess in M_i^2 in message 2, by the following: $M_i^2 = K_i \oplus C_i^2 = (M_i^1 \oplus C_i^1) \oplus C_i^2 = M_i^1 \oplus (C_i^1 \oplus C_i^2)$, where K_i is the (guessed) i^{th} byte of the secret pad and C_i^1 and C_i^2 are the i^{th} ciphertext bytes of messages 1 and 2, respectively.
- The uppercase letters A-Z have values 0x41 through 0x5A, while the lowercase letters a-z have values 0x61 through 0x7A. Therefore an uppercase letter's ASCII code differs from its lowercase counterpart by only one bit. This fact lessened our guesswork by one half, since the case of the characters was irrelevant. Guessing a correct letter, but the wrong case, would only reverse the case of the corresponding character in the other message. It was easy to assign the proper case to each character after all the plaintext was known.
- Whitespace characters like space and linefeed have low ASCII values. Taking the exclusive OR of each pair of corresponding characters would therefore be very valuable. Any result greater than 0x2F would indicate that at least one of the two characters was special, i.e. not a letter. This gave us useful information about word lengths in the plaintext.
- The problem stated that, if necessary, one of the messages would be padded with spaces to equalize their lengths. We guessed that one message would end in several spaces, and by examining the corresponding guessed characters in the other message, we were able to determine the last three letters of both messages.

After forming a plan of attack based on the above facts, we wrote a perl script that allowed us to guess a particular letter of either message, examine the consequences, and commit the guess if we thought it was correct. Based on the discovered word lengths (described in the third item above), we were able to guess `I␣`, which yielded `An` in the other message. After guessing `And␣`, we had `I␣sh` in the first message. A bit of trial and error, and alternating between messages, yielded `I␣shall␣be␣` in the first message. At this point our literary expert recalled the entire quote from memory. Filling in the rest of the quote and discovering the second message was a trivial application of the perl script.

There was virtually no pure guesswork involved in this exercise. Using the above observations and one team member's sharp sense of prose, our "in the dark" guessing was limited to a few minutes. The entire exercise took about two hours to complete.

Problem 1-3. Snake Oil or Good Crypto?

Chris Peikert
Brandon DuRette
Anthony Young-Garner

The security claims made by TriStrata are based upon their use of a one-time pad (OTP) to provide encryption. However, the weakness in their security model is independent of the use of OTP; instead it is dependent upon the security of their authentication scheme and other implementation details.

Much of TriStrata's marketing information is deceptive. A great amount of text is devoted to explaining the perfect theoretical security of the Vernan Cipher, or one-time pad. It was difficult to even determine TriStrata's claims about their system. For instance, the front page of TriStrata's "Products" section claimed that their security was "unbreakable", but that "defeating one TriStrata-encrypted message would take $3.5 * 10^{33}$ years." These claims contradict one another.

Very little attention is given to implementation details, which is where most security systems fail. For example, the following areas were left untouched or were explained inadequately:

- **Security of seals.** The TriStrata documentation states that "Along with the permit, the TESS also sends a seal. The seal is attached to the encrypted document. Only the TESS can open the seal - the data in the seal is not accessible to the user." We reasoned that the seal is a unique message identifier or key offset pointer. If it is not a reference to information stored on the TESS, then the claim that "the data ... is not accessible to the user" is dubious.
- **Security of permits.** The documentation continues, "To decrypt a file, a request for decryption is sent to the TESS along with the seal. A permit is returned from the TESS allowing the decryption to be performed locally." If the decryption is performed locally, then the permit must contain the key material that is used in decryption. There is a problem with this approach: the key material must be transmitted confidentially and with authentication. The security of the system is only as strong as this link.
- **Authentication at TESS.** A user requesting a permit to decrypt a message must identify himself to the TESS and be authorized to decrypt that message. Otherwise, any user could obtain a permit for any message, making the encryption useless. The methods of authentication and authorization were unspecified. Unless these methods also offer unconditional security, TriStrata's claims about the unconditional security of their entire system are fallacious.
- **Authentication of messages.** TriStrata makes no claims about authentication of user messages. In many systems, authentication is more important than confidentiality. The utility of a security system that does not provide authentication is low.
- **Storage of the key material.** Apparently the one-time pad is either stored on the server or generated by the information in a seal. Either method leaves vulnerabilities. If the key material is stored, an attacker could concentrate his efforts on the TESS server. If the key material is generated, then the attacker need only get the random number generator algorithm in order to decrypt any message.
- **Random bit generator.** The only documentation relating to the random bit generator says, "With RKS, a Random KeyStream derived from a physical random number generator is used as the cipher key." A physical generator is preferred to a pseudo-random generator of course,

but the details of this generator are not disclosed. A weak generator can compromise the unconditional security of a OTP system.

We contacted TriStrata about some of these issues but they did not reply by press time. Based on our analysis and uncertainty about TriStrata's algorithms and protocols, we would not recommend this system to a large institution. There are too many potential vulnerabilities that need to be explored. For a casual user, this system is entirely inappropriate; it requires dedicated servers and complicated management.