

Re: Coding Theory Question, I think

Source: <http://sci.tech-archive.net/Archive/sci.math/2005-09/msg03016.html>

- *From:* Jyrki Lahtonen <lahtonen@xxxxxx>
 - *Date:* Tue, 13 Sep 2005 15:29:17 +0300
-

Thinus Pollard wrote:

Nope, I did some abstract algebra a few years ago where we did some coding theory.

The system issues tickets. Each ticket contains a PIN. This PIN needs to be unique and only I should be able to issue valid PINs.

The PIN contains an issue date, an expiry date, a unique serial number and an amount. All this data should be encoded into a 16 *digit* string.

Validation should take the following form:

1. When entered the system should check if the entered string is valid. This is easy, use 15 digits and append a check as the 16th.
2. When verified the system should check if the data inside the PIN "makes sense".

I was thinking about packing all the data into some bits and then encrypting it with a cipher (public key or symmetrical). The problem with this idea is how to map the data to a 15 digit string, without losing information? From the 16 digits you should be able to extract the information if you have the cypher keys.

I apologize if the original post was a bit vague.

regards,
T

15 digits means that you have a space of 10^{15} combinations. 10^{15} is slightly smaller than 2^{50} , so we may also say that

Re: Coding Theory Question, I think

you have 49 bits to play with. 49 bits or 15 digits is probably enough to contain the necessary amount of data.

I sorta got the feeling that the your main problem was not to pack the data into this space, but to get protection against somebody claiming that his/her random sequence of digits is a valid ticket simply because it passes the check equation.

I am no expert on cryptographic protocols or whatever you might call these schemes, but would the following fit your needs:

Issuing a ticket

- 1) pack the ticket data (dates+PIN) into say 10 digits
- 2) compute the value of a suitable hash function using that 10 digit string as input. Pick a hash function whose values can be expressed in 5 digits.
- 3) encrypt the 10+5 digit string consisting of the ticket data plus the hash
- 4) compute 1 check digit

Validation of a ticket

- 1) verify that the check digit(s) is correct
- 2) decrypt the 15 digit string.
- 3) check that the 5 last digits of the string are the hash value of the first 10 digits. If yes, then the ticket is accepted. Otherwise the ticket just contains a random string.
- 4) At the entrance gate, you probably must keep a database of the PINs of the used tickets (otherwise a ticket can be simply copied).

The idea is that changing one bit/digit of the actual data will result in an unpredictable change in the value of the hash function, so many bits will always change (you could also use an error-correcting code for the same purpose). Tagging the hash along, of course, also means that not all the 15 digit strings (without the check digit) are valid. Because after decryption the last 5 must match the first 10.

Even if the bad guys break your encryption, they still need to know also figure out your hash function. Ok, if they get their hands on your algorithm, you are screwed anyway.

I'm no expert so there may be some immediately obvious holes in this scheme. 5 digits (i.e. 16 bits) is also probably a bit short, but may work against unskilled opponents.

Re: Coding Theory Question, I think

Digital signatures could be used to add many features (e.g. ticket dates and PIN would be certified by your PGP signature), but I don't think any such reliable systems exist that would fit into 49 bits :)

Ask people at sci.crypt for better hints!

Cheers,

Jyrki Lahtonen, Turku, Finland

.