

## Subliminal Channels

**Subliminal channels** are covert ways that an attacker can send a second message hidden within a normal message. A simple example of this is the bit string formed by the parity of the number of letters in each word of a message. A carefully constructed, innocent sounding message may contain another message hidden in this bit string.

As another example, a message may be hidden in certain bits of a digitized image. Some cryptographic algorithms having subliminal channels are schemes that choose random numbers used just once, such as the Digital Signature Algorithm and the signature algorithms of El-Gamal and Ong-Schnorr-Shamir.

Here is a subliminal channel for the DSA. Alice is a spy who sends many plaintext messages to her contact Bob. Alice's employer reviews these messages to ensure she is not divulging any secrets. All of the messages are signed by the DSA. Alice and Bob secretly agree on a prime  $p_1$ , different from any parameter of the DSA. When Alice signs an innocuous message  $M$ , she hides a subliminal bit in it. If she wants to send a 1 bit, she tries different random numbers  $k$  until the  $r$  parameter of the signature is a quadratic residue modulo  $p_1$ . For a 0 bit, she makes  $r$  a quadratic nonresidue modulo  $p_1$ . Since half of the  $r$  are quadratic residues and half are quadratic nonresidues, she doesn't have to try many random  $k$  to do this.

Bob checks each DSA signature to be sure the message came from Alice. Alice's employer could make the same check and find nothing amiss. Bob would read the subliminal bit in each message by evaluating the Legendre symbol  $(r/p_1)$ .

In fact, Alice could send several subliminal bits per message.

Suppose Bob and Alice agree on  $j$  secret primes  $p_1, \dots, p_j$ . Then  $j$  subliminal bits could be sent per message, with the  $i$ -th bit being 0 or 1 according as  $r$  is a quadratic residue or non-residue modulo  $p_i$ .

On average, Alice would have to try  $2^j$  values of  $k$  to get an  $r$  with the required properties, so she can't make  $j$  too big or her employer might wonder why her DSA was so slow. She could choose  $j = 16$  and send two bytes per message.

An evil implementer of DSA could use the subliminal channel to leak Alice's private key.

Mike sells cheap DSA chips with a 14-bit subliminal channel using fourteen secret primes. When the chip signs a message, the user gives it her private 160-bit key  $x$ .

Alice buys a DSA chip from Mike and signs messages with it. The chip breaks the 160-bit key  $x$  into 16 ten-bit pieces. It chooses a random 4-bit number  $e$  and sends the subliminal message ( $e$ , the  $e$ -th piece of  $x$ ).

Mike deduces ten bits of Alice's private key from each signature. After many signatures, he learns many ten-bit pieces of  $x$ . When he knows most pieces he can compute the remaining bits by trying all possibilities. Then he can forge Alice's signature on messages.

Even if Alice knew that Mike was stealing DSA private keys this way, she could not prove it unless she knew Mike's 14 secret primes.