

4. Systems designed to improve text message processing can see the OTT.

Possible Redesigns

Information disclosure threats can be addressed by adding cryptographic functions. Simplified versions of ways to do that include the following:

- Send a nonce, encrypted to a key held on a smartphone, then send the decrypted nonce to the authentication server (message 1 = $e_{\text{phone}}(\text{nonce}_n)$, message 2 = nonce_n). Then the server checks whether the nonce_n is the one that it encrypted for the phone, approving the transaction if it is.
- Send a nonce to the smartphone, and then send a signed version of the nonce to the server [message 1 = $\text{nonce}_{\text{phone}}$, message 2 = $\text{sign}_{\text{key}(\text{phone})}(\text{nonce}_{\text{phone}})$]. The server validates that the signature on the $\text{nonce}_{\text{phone}}$ is from the expected phone's key, and is a good signature on the expected nonce. If both those checks pass, then the server approves the transaction.
- Send a nonce to the smartphone. The smartphone hashes the nonce with a secret value it holds, and sends back the hash.

For each of these, it's important to manage the keys appropriately, and it's probably useful to include time stamps, message addressing, and other elements to make the system fully secure. Including those in this discussion makes it hard to see how cryptographic building blocks could be applied.

The key in all design changes is understanding the differences introduced by the changes, and how those changes interact with the software requirements as a whole.

Redesigns that focus on using the phone as a processor preclude the use of an old-fashioned telephone, or even a mobile phone (of the kind where the end user can't easily install software). Because such phones still exist, it may be that the threats just enumerated are considered acceptable risk, or even an improvement over traditional passwords.

Sample for You to Model

You can use the models presented above as training models with answer keys. (That is, use the software model in Figure E-1 and the operational model in Figure E-2 and find threats against them yourself. You can treat the example threats