

**Common Hacks and SHK Counter Attacks**

<i>Attack Name</i>	<i>Attack Definition</i>	<i>The Impact</i>	<i>How To Prevent</i>	<i>SafeNet Solution</i>
Brute Force Attacks	Brute force attacks exhaustively attempt every security combination until the secret being attacked is compromised.	A brute force attack can compromise the algorithms that are central to the security of the dongle. This affects all dongles for a particular software vendor. It is not a generic hack.	Preventing a brute force attack can be accomplished when data is large enough to make cracking all combinations virtually impossible.	The SHK's security kernel is the Device Update Key, a 128-bit AES key, to encrypt and protect the algorithms central to the dongle.
Device Emulation	Device Emulation occurs when a piece of software emulates the hardware dongle. The software emulates the device and presents itself to the application as if it were the hardware dongle.	Device emulation is a generic hack on the dongle. A generic hack means that the dongle itself is compromised. No amount of implementation improvement can counter a generic hack. All applications protected by the hacked dongle type are threatened.	Device emulation attacks the secrets of the dongle, typically via a brute force attack. Brute force attacks attempt to unlock every possible security combination until all choices are exhausted. This results in ultimately cracking the secret. Device emulation can be prevented by securing the storage area on the dongle, making the dongle virtually impossible to compromise using a brute force attack.	The SHK uses a 128-bit AES key which is impossible to attack using brute force with today's computing power. For a description of the significance of key size, please see the following link: <a href="http://en.wikipedia.org/wiki/Key_size">http://en.wikipedia.org/wiki/Key_size</a> (Wikipedia entry: "key size")
Record/Playback	In a record/playback attack, all information is recorded between the application and the dongle. A middleware (software application) is then created by the hacker to mimic the responses provided by the dongle.	When a record/playback attack has been conducted, the application has been compromised. The vendor is forced to redo their implementation in order to combat this type of attack. It is not a generic hack, but affects the application in which a record/playback attack exists.	Record/playback attacks can be prevented by encrypting and randomizing the communication between the application and the dongle.	We use a random 128-bit key to encrypt the communication between the application and the SHK.
Stealing Secrets	Stealing secrets occurs when a security password is obtained, providing unauthorized access to the dongle.	Once access to the dongle is obtained, hackers are able to manipulate the application using the dongle.	Passwords should never be communicated without being encrypted.	We encrypt the data that is being sent to SHK. The passwords and other secrets are never sent "in the clear."
Device Sharing	Device sharing implies that multiple PCs are able to share one dongle.	One dongle can be used by several machines on the same network. While the application itself is not in jeopardy, the software vendor will not receive the anticipated revenues for use of multiple copies of their application.	Software vendors need to ensure that only one computer or device driver can access the dongle, unless specifically allowed by the vendor.	The secure tunneling of the SHK safeguards against multiple devices or machines trying to access the token. If multiple machines access the token, the application will not work.
Hardware Cloning	Hardware cloning occurs when hardware tokens are duplicated, which authorize the protected application.	To duplicate hardware token memory, the hacker clones the tokens by copying memory contents from the original token. The fake tokens still come from the token vendor.	The software vendor needs to ensure that the token's memory is encrypted.	In this case, SafeNet's solution is encrypting memory by token-unique 128-bit AES key.