# Meet-in-the-Middle Attacks

Stephane Moore

November 16, 2010

A meet-in-the-middle attack is a cryptographic attack, first developed by Diffie and Hellman, that employs a space-time tradeoff to drastically reduce the complexity of cracking a multiple-encryption scheme. To illustrate how the attack works, we shall take a look at an example.
Let $E_K$ and $D_K$ denote encryption and decryption functions using the key $K \in \{0,1\}^n$. Similarly, let $E'_K$ and $D'_K$ denote encryption and decryption functions using the key $K \in \{0,1\}^m$. Consider the following simple double-encryption scheme which computes a ciphertext message $C$ from a plaintext message $P$ using two keys $K_1 \in \{0,1\}^n$ and $K_2 \in \{0,1\}^m$:

$$\begin{aligned} C &= E'_{K_2}(E_{K_1}(P)) \\ P &= D_{K_1}(D'_{K_2}(C)) \end{aligned}$$

A naive attack on this double-encryption scheme, covering the entire search space of $\{0,1\}^n \times \{0,1\}^m$, would require $O(2^{n+m})$ encryptions. However, exhaustive searches to crack $E_K$ and $E'_K$ individually would only take $O(2^n)$ and $O(2^m)$ encryptions, respectively. There is an important derivation from this double-encryption scheme that we can exploit to construct a more sophisticated attack.

$$\begin{aligned} C &= E'_{K_2}(E_{K_1}(P)) \\ D'_{K_2}(C) &= D'_{K_2}(E'_{K_2}(E_{K_1}(P))) \\ D'_{K_2}(C) &= E_{K_1}(P) \end{aligned}$$

This derivation *meets in the middle* of the double-encryption scheme and allows us to use exhaustive searches over $E_K$ and $E'_K$ in a more efficient chosen-plaintext attack. Consider one possible approach based on computing the following sets:

$$\begin{aligned} H &= \{(K, E_K(P)) : K \in \{0,1\}^n\} \\ S &= \{(K_i, K_j) : K_i \in \{0,1\}^n \wedge K_j \in \{0,1\}^m \wedge (K_i, D'_{K_j}(C)) \in H\} \end{aligned}$$

Here, we precompute the set of all possible encryptions of the plaintext $P$ using $E_K$ and store a lookup table $H$. Afterwards, we compute the set of all possible decryptions of the ciphertext $C$ using $D'_K$ and check for membership in the lookup table. The intersections between the two described sets will contain the correct key pair $(K_1, K_2)$. If there are multiple key pairs in the intersection, then we can test the candidate key pairs using additional plaintext-ciphertext pairs and quickly isolate the correct key pair. This constitutes a much more efficient attack on this double-encryption scheme.
This meet-in-the-middle attack requires $O(2^n + 2^m)$ encryptions to compute the two sets instead of the $O(2^{n+m})$ encryptions required by an exhaustive search. We do incur $O(2^n)$ or $O(2^m)$ space overhead, depending on the approach, in storing the lookup table; however, with modern resources, the space overhead is typically not unreasonable. Meeting in the middle reduces the search space drastically and points out that cracking the double-encryption scheme is computationally similar to cracking the encryption functions that compose it. The math becomes even more alarming in the case where $n = m$, as this discrepancy becomes $O(2^{2n})$ encryptions for the naive attack and $O(2^{n+1})$ encryptions for the meet-in-the-middle attack, which is only twice what it would take to crack $E_K$. For this reason, simple multiple-encryption schemes tend to provide considerably fewer bits of effective security than the actual number of key bits used in the encryption scheme.

**Example: E-D-E Triple DES**

For a more applied example of a meet-in-the-middle attack, we shall focus on *E-D-E* triple encryption using the Data Encryption Standard (DES) cipher algorithm. This encryption scheme is a keying option to Triple DES (3DES) that uses three 56 bit keys. If $E_K$ and $D_K$ denote DES encryption and decryption functions using a key $K \in \{0,1\}^{56}$ then our *E-D-E* triple encryption can be described as follows:

$$\begin{aligned} C &= E_{K_3}(D_{K_2}(E_{K_1}(P))) \\ P &= D_{K_1}(E_{K_2}(D_{K_3}(C))) \end{aligned}$$

From this encryption scheme, we can derive the following:

$$\begin{aligned} C &= E_{K_3}(D_{K_2}(E_{K_1}(P))) \\ D_{K_3}(C) &= D_{K_3}(E_{K_3}(D_{K_2}(E_{K_1}(P)))) \\ D_{K_3}(C) &= D_{K_2}(E_{K_1}(P)) \end{aligned}$$

From this derivation, similar to the double-encryption scheme detailed before, we can describe a meet-in-the-middle attack on the *E-D-E* triple encryption scheme using DES as follows:

$$\begin{aligned} H &= \{(K, D_K(P)) : K \in \{0,1\}^{56}\} \\ S &= \{(K_a, K_b, K_c) : K_a, K_b, K_c \in \{0,1\}^{56} \wedge (K_c, D_{K_b}(E_{K_a}(C))) \in H\} \end{aligned}$$

Here, we construct a lookup table using $O(2^{56})$ encryptions and store it in $O(2^{56})$ memory. Next we find candidate keys using $O(2^{112})$ encryptions and then isolate the correct key. In this way, the meet-in-the-middle attack allows us to find the correct keys $K_1$, $K_2$, and $K_3$ in roughly $O(2^{112})$ encryptions. For this reason, *E-D-E* triple encryption using DES, which is the strongest keying option of 3DES, is considered to have at most 112 effective bits of security despite having 168 key bits.