SANS Institute
InfoSec Reading Room

S-Box Modifications and Their Effect in DES-like Encryption Systems

This paper presents the substitution boxes (s-boxes) found in many block ciphers, and more specifically in DES-like encryption systems. It begins with a brief history of the Data Encryption Standard (DES) and the first public question on the chosen s-boxes. An outline of the DES algorithm is presented, along with a more detailed look of the cipher function that uses the s-boxes. The major methods of cryptanalysis are reviewed, including how they use the s-boxes for their attacks, and how the risk can be mitigated by al...

S-Box Modifications and Their Effect in DES-like Encryption Systems

Joe Gargiulo

GSEC v1.4 option1

July 25, 2002

1.0 Summary

This paper presents the substitution boxes (s-boxes) found in many block ciphers, and more specifically in DES-like encryption systems. It begins with a brief history of the Data Encryption Standard (DES) and the first public question on the chosen s-boxes. An outline of the DES algorithm is presented, along with a more detailed look of the cipher function that uses the s-boxes. The major methods of cryptanalysis are reviewed, including how they use the s-boxes for their attacks, and how the risk can be mitigated by alternate schemes. Potential changes to s-boxes described, as well as how these changes may or may not strengthen DES-like encryption systems. Finally, there is a brief example of how some researchers underwent rigorous DES-like s-box construction testing.

2.0 DES and S-Box History

In the early 1970s, the National Bureau of Standards (now known as the National Institute of Standards and Technology) requested from the public any proposals for a cryptographic algorithm following certain specific criteria.[1] Some of the requirements included that the algorithm must be completely specified, and available to all users. A team from IBM responded to the request, and the details of the algorithm were published after the National Security Agency (NSA) evaluated its suitability. This NSA involvement raises some questions in the design of the mysterious s-boxes used in the algorithm, which will be described in more detail in the following sections. Regardless, DES was approved as a federal standard "Data Encryption Standard (DES)" FIPS PUB 46 and later by the National Standards Institute (ANSI) as the "Data Encryption Algorithm" ANSI X3.92.[2]

After approximately 20 years of substantial governmental and commercial use, DES no longer is considered secure. However, the cryptanalysis of DES, and more specifically, the analysis of the s-boxes remains useful today. Many of the newer cryptographic algorithms use DES-like s-boxes, and are implemented widely as they are still considered very secure.

3.0 The DES Algorithm

Before the s-boxes can be analyzed in detail, a general look at the DES ANSI X3.92 algorithm is needed. There are many block ciphers to choose from that use substitution boxes (s-boxes); however, DES is one of the most widely used and researched encryption algorithms, making it valuable to study.

DES uses a secret key of 56 bits (effective size) and plaintext of 64 bits to create 64 bits of ciphertext. This is done by the following steps of the enciphering computation as shown in figure 1 (the details of some trivial permutations have been intentionally left out).

1. Create the key schedule of 16 compressed keys *(K_i where $1 \bullet i \bullet 16$)*, by permutations and shifts of the original secret key *(K)*. These 48-bit keys will be used later for each of the 16 rounds of the DES function.

2. Perform an initial permutation *(IP)* of the plaintext, and split it into 32-bit left and right halves *(L_0, R_0)*.

3. For each of the 16 rounds:
   $L_i = R_{i-1}$ and
   $R_i = L_{i-1} \oplus f(R_{i-1}, K_i)$, where $f(R_{i-1}, K_i)$ is the Cipher Function (described in more detail in section 3.1)

4. Exchange the final blocks L_{16} and R_{16}.

5. Perform the inverse permutation *(IP^{-1})* and the output is the ciphertext.

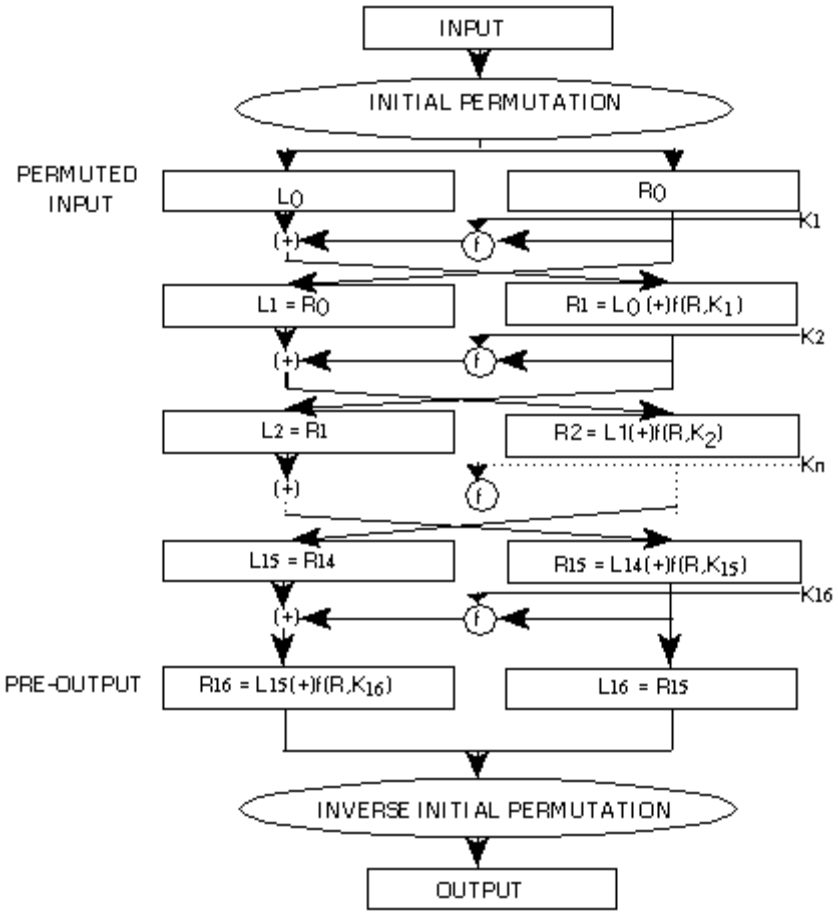

Figure 1: Enciphering Computation [3]

3.1 Cipher Function *f(R,K)*

The encryption algorithm gets its strength from the Cipher Function (refer to figure 2), the steps of this function are described below:

1. Take the 32-bit right half input (*R*), and reorder the bits into eight 6-bit blocks from table 1 below. This is called the expansion function:

	Bit 1	Bit 2	Bit 3	Bit 4	Bit 5	Bit 6
Block 1	32	1	2	3	4	5
Block 2	4	5	6	7	8	9
Block 3	8	9	10	11	12	13
Block 4	12	13	14	15	16	17
Block 5	16	17	18	19	20	21
Block 6	20	21	22	23	24	25
Block 7	24	25	26	27	28	29
Block 8	28	29	30	31	32	1

Table 1: The expansion function E(R) [4]

2. XOR the 48-bit output with the 48-bit key $(E(R_{i-1}) \oplus K_i)$.

3. Run those eight 6-bit blocks though the s-box, yielding eight 4-bit bocks. This is described in more detail in section 3.2.

4. Do one final permutation of the 32-bit output.

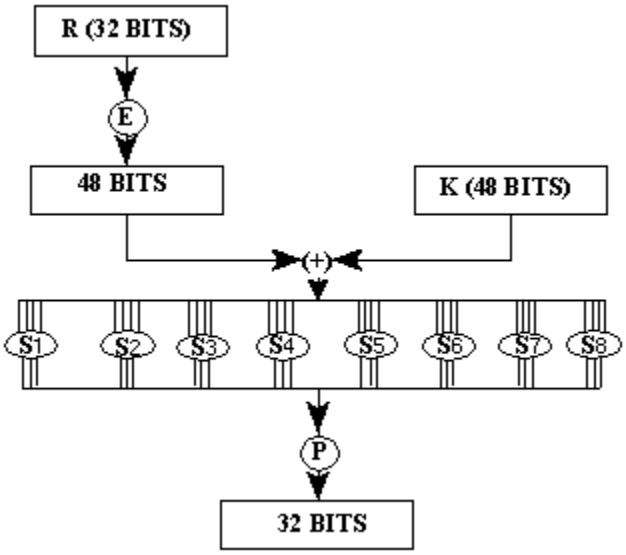

Figure 2: Calculation of f(R, K) [3]

3.2 The S-Box Function

The s-box is where the Cipher Function gets its security. The details of how the s-box works are as follows:

The 48-bit input (from $E(R_{i-1}) \oplus K_i$) is separated into eight 6-bit blocks *(B_{1-8})*. Each block is subjected to a unique substitution function *(S_{1-8})* yielding a 4-bit block as output. This is done by taking the first and last bits of the block to represent a 2-digit binary number *(i)* in the range of 0 to 3. The middle 4 bits of the block represent a 4-digit binary number *(j)* in the range of 0 to 15. The unique substitution number to use is the one in the i^{th} row and j^{th} column, which is in the range of 0 to 15 and is represented by a 4-bit block.

For example:

The first s-box (S_1) is shown in table 2 below.

	Col 0	Col 1	Col 2	Col 3	Col 4	Col 5	Col 6	Col 7	Col 8	Col 9	Col 10	Col 11	Col 12	Col 13	Col 14	Col 15
Row 0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
Row 1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
Row 2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
Row 3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

Table 2: The first DES s-box (S_1) [4]

If the first block of the input *(B_1)* is 011010, then the row is 00 (or simply 0) and the column is 1101 (or decimal 13). By looking up the result in the first s-box *(S_1)*, it is found that the resultant unique 4-bit number is 9 (or binary 1001).

It is these carefully designed s-boxes that create the properties of the ciphertext in DES-like encryption systems. The other parts of the algorithm (expansions, permutations, etc.) are mathematically linear, and can be picked apart very simply. It is imperative that the cryptanalyst understand this process fully before attempting to create new s-boxes. In the next section, it is shown that even the slightest changes have potential to weaken the strength of the encryption standard.

4.0 Cryptanalysis Techniques

With the adoption of DES 20 years ago, many critics questioned the involvement of the NSA in generating the s-boxes. Some questioned if a back-door, or at the very least, a reduced capability open to cryptanalysis, was hidden in the boxes. This prompted several researchers to find alternate s-boxes that would be secure without worry of the NSA potentially cracking them. Since the creation of DES, plenty of research has gone into finding better s-boxes and better techniques to break them.

The following sections 4.1-4.4 review the most successful cryptanalysis techniques on DES-like encryption systems. These techniques have two major components: known plaintext/ciphertext pairs, and processing overhead (or complexity).

Known plaintext/ciphertext pairs are blocks of encrypted data where the plaintext is already known. These pairs are obtainable by any number of methods. A common method would be a guess at the file type of the encrypted document, which would give away several blocks since the headers are usually known. Intercepting ciphertext that is of a known document would be a large advantage, as several plaintext/ciphertext pairs would then be known. Still, some of the best cryptanalysis techniques detailed below requires more than 2^{43} known plaintext/ciphertext pairs. This large amount of known data keeps many methods in the theoretical realm.

Processing overhead can be mitigated by several means. Dedicated machines could be built specifically for cryptanalysis. Parallel computers could be used in a group to reduce time constraints. There are also many papers on the evolution of processor speed and the predictions for faster processors in the future. If a method of cryptanalysis takes one year to perform today, then in five years it certainly will be performed much quicker.

4.1 Exhaustive Search

The most basic method of cryptanalysis is the *exhaustive search* method, also known as the *brute force* and *known plaintext* attack. This method uses merely one block of known plaintext and the resultant ciphertext to find the secret key. With this pair of known plaintext/ciphertext, it could take 2^{56} maximum DES calculations to find the correct secret key in DES encryption. With today's processing power this not only is possible; indeed it has been performed time and time again.[5]

Although exhaustive search is related entirely to the DES secret key length of 56 bits rather than s-boxes, this information is valuable as a baseline for the usefulness of other cryptanalysis techniques.

4.2 Linear Cryptanalysis

Linear cryptanalysis was first openly published as a means for attacking DES by Mitsuru Matsui in EUROCRYPT'93.[6] His method attempts to find a linear relation among the plaintext, ciphertext, and keys as they pass through the s-boxes. With enough known plaintext/ciphertext pairs as data, a relation with a high enough probability can be used to find the key.

Matsui generated linear approximation tables for the 8 DES s-boxes and found the strongest linearity in S_5 (the fifth s-box). The tables were created by analyzing all the combinations of the input and output bits of the s-boxes. Since there are 6 input bits and 4 output bits, there are 1024 ($= 2^6 \cdot 2^4$) entries in his tables for every s-box. A linear approximation is stronger if it is significantly greater or less

than 50% probability. That particular entry in S_5 had a value of -20, representing a probability of 12/64 (= *1/2 - 20/64*). This value is considered strong enough, and it allows the linear cryptanalysis on DES to be possible.

In order to achieve approximately an 85% success rate using this attack method, 2^{43} known plaintext/ciphertext pairs are needed. However, processing overhead is less than the exhaustive search method at 2^{43}.[7]

Eli Biham took this one step further to help define restrictions on s-boxes to make them more resistant to linear cryptanalysis.[8] He found that increasing the number of output bits of an s-box can endanger the s-box significantly to linear cryptanalysis. More precisely, he found that in an *m·n* s-box, where *m* is the number of input bits and *n* is the number of output bits, if $n \bullet 2^m\text{-}m$, the s-box *must* have a linear property of the input and output bits.

4.3 Differential Cryptanalysis

Eli Biham and Adi Shamir published their new method of cryptanalysis in 1990 called Differential Cryptanalysis.[9] They define this as:

> A method which analyses the effect of particular differences in plaintext pairs on the differences of the resultant ciphertext pairs. These differences can be used to assign probabilities to the possible keys and to locate the most probable key.[10]

The idea behind differential cryptanalysis is to throw out key choices that are unlikely, and keep choices that are very likely. From this reduced subset, a cryptanalyst can run an exhaustive search to find the correct key.

In order for differential cryptanalysis to be successful, 2^{47} chosen plaintext/ciphertext pairs are needed. The processing overhead also is less than exhaustive search at 2^{47}.[11] However, Schneider suggests that "the enormous time and data requirements to mount a differential cryptanalytic attack put it beyond the reach of almost everyone."[12]

There are various ways to secure DES from this type of attack. One method relating to s-boxes is to increase the number of output bits[13] *n* of the s-box whereas $m < n$, using care to not increase it too far, as described above, putting it at risk for linear cryptanalysis.

4.4 Improved Davies' Attack

Eli Biham and Alex Biryukov updated what they called a purely theoretical cryptanalysis method called the Davies' Attack in 1997.[14] This attack finds an interesting trait in the distribution of data based on the expansion function *E(R)*. This occurs due to the expansion operation duplicating data bits. Each s-box shares two bits with the s-boxes on each side of it (refer back to table 1).

As an example, in the second block of six data bits the first two bits (4 and 5) are shared with the first block, and the last two bits (8 and 9) are shared with the third block.

The original description of the Davies' Attack required $2^{56.6}$ known plaintexts. Biham and Biryukov's improvements to the method claim to require only 2^{50} known plaintexts and has a processing overhead of 2^{50}. Although this is better than exhaustive search, it still falls behind linear and differential cryptanalysis.

They also found that s-boxes can be formed to be resistant to this attack, by paying close attention to the distributions in them. The distribution of the outputs of pairs of adjacent s-boxes must be uniform, thus not lending themselves to a relation that can be used.

5.0 S-Box Design

With the primary modes of attack on DES-like algorithms defined, rules can be established for how s-boxes are designed and used. Researchers can also examine how other's s-box designs match up against those cryptanalysis techniques. There are several ways of making better s-boxes than the ones specified in DES, however, Schneider states that "… blindingly choosing new s-boxes isn't a good idea."[15]

Among the common and well-documented features of s-boxes that are considered viable are those that permit the algorithm to follow the Strict Avalanche Criteria (SAC). The avalanche effect was first published in the cryptography world by Horst Feistel.[16] In that study, it was determined that when an input bit goes through the system, an equal number of 1's and 0's on average are the resultant output. This was taken one step further by Webster and Tavares[17], requiring exactly half of the output bits to change when one input bit changes.

Another consideration is the *size* of the s-box. From the above discussions on cryptanalysis, a large box would be better than a small one. A large number of output bits are needed to protect against differential attacks; however, a corresponding large number of input bits are also needed to protect against linear cryptanalysis. Obviously, a balance of the two is needed.

Finally, there are three requirements regarding the *values* in the s-box. First, the distributions of outputs must be checked for uniformity to protect against the Davies' Attack. Second, the outputs must have no linearity in their function to the input. Third, there must be unique values in every row of the s-box. There are several other requirements; however they are beyond the scope of this paper.

Even with these requirements for s-boxes, there are still many choices. Just a small sample of what some researchers have tried is expanded in sections 5.1-5.5 below.

5.1 Random S-Boxes

One of the easiest ways to change DES is to create random s-boxes. Unfortunately, this is also one of the easiest ways to make the s-boxes prone to differential cryptanalysis. This is due to the fact that with random s-boxes (unlike DES s-boxes), it is likely that there are two different inputs with different middle bits, which have the same output. This is a characteristic that enables the attack. Biham and Shamir found an example in which with a single change to a single s-box in DES, only 2^{37} pairs of plaintext/ciphertext were needed to conduct an attack.[9]

Biham and Shamir also tried an approach called Random Key-Dependent S-boxes, where the boxes were generated by the secret key. This was cracked with only 2^{29} plaintext/ciphertext pairs.

These findings are significant enough to render the use of random s-boxes useless. However, Schneider found a study that shows if the s-boxes are large enough (such as 12 input bits,) then it may be secure.[2]

5.2 Change the Order of the S-Boxes

If creating random s-boxes is not an option, another method is to just change the fixed ordering of the existing s-boxes. However, it is not safe to assume that this is a secure scheme. Biham and Shamir proved that even a small modification to the ordering can make DES weaker.[9] For instance, it was found that if the first three s-boxes are in the order of S_1, S_7, and S_4, and the other s-boxes are in any order, it is much more susceptible to differential attacks.

Haphazardly changing the order of the s-boxes is not safe, yet the ordering in DES is not optimized for all attacks as it is defined. By analyzing all possible orders (8!), Biham and Biryukov found best one[18], most resistant to the above attacks: S_2, S_4, S_6, S_7, S_3, S_1, S_5, and S_8. This order does not provide much additional protection against differential cryptanalysis, but it does have a significant improvement against linear cryptanalysis and the improved Davies' Attack.

5.3 Key-Dependent Reordering of the S-Boxes

Not only did Biham and Biryukov find the best ordering of the s-boxes, they also found a pool of 32 strong ones.[18] They created a method in which an extra five bits of a key would choose the ordering from this list. This DES-like algorithm is obviously strengthened against differential, linear, and Davies' Attacks, and it is also much more effective against exhaustive search, due to the extended key length.

5.4 Key-Dependent S-Box Transformations

Key dependent s-box transformations are yet another method described by Biham and Biryukov.[18] This method uses six bits of the key to transpose the rows and columns of an existing s-box. The value is then XORed with another four bits of the key. They proved that this method is not influenced by linear or differential attacks, and they claim that the improved Davies' Attack becomes more complicated.

5.5 S-Box Design Using Bent Sequences

Another approach to generating s-boxes is to base the design on mathematical properties. Boolean functions seem to be the most promising due to the binary nature of s-boxes. However, only a subset of these functions is usable, since a primary characteristic of acceptable s-boxes is the Strict Avalanche Criteria (SAC) that was described above. Adams and Tavares determined that this class of functions that satisfy the SAC is "… identical to the class of functions known in combinatorial theory as 'bent' functions".[19] Three years later, they found that bent sequences alone could not guarantee protection against attack. They summarized that "without extra precautions, systems incorporating such s-boxes are vulnerable to differential cryptanalysis".[20]

6.0 s^nDES Attempts

Over the course of many years, several researchers around the world have developed a DES-like variant commonly known as s^nDES. The goal was to make the best possible alternative s-boxes, taking into consideration the recent research in cryptanalysis described above.

The first implementation (s^2DES) was described by Kwangjo Kim in 1991.[21] The design was to be checked for the Strict Avalanche Criteria, nonlinearity, bijection, and several other criteria deemed important at that time. He created the s-boxes using bent Boolean functions, and published some examples. He stated "the differential characteristics of s^2DES s-boxes are found to exhibit better than those of DES s-boxes. This suggests that s^2DES can resist better than DES against differential cryptanalysis."[22] However, he also recommended that more research was needed in this area.

In 1992, it was found that s^2DES was in fact *more* susceptible to differential attack then DES. Kim and his team went back to work and in 1993, published updates to the s-boxes and called it s^3DES.[23] When applying the latest methods of differential attack to their s-boxes, it was found that the processing overhead now ranged from 2^{96} to 2^{112}, which is far more complicated than exhaustive search. They suggested that linear cryptanalysis of s^3DES was still an "open problem."

In 1994, Biham and Biryukov (masters of s-box ordering at the time) found that reversing the orders of the first two s^3DES s-boxes would make it considerably

more difficult to perform a linear attack.[18] Kim and his team quickly adopted the changes, and set out to define new s-box criteria.

Kwangjo Kim, Sangjin Lee, Sangjun Park, and Daiki Lee released the specifications for s^5DES in 1995 (s^4DES was never formally distributed).[24, 25] The new criteria created s-boxes considerably more immune to differential, linear, and Davies' Attacks. More specifically, they found that differential cryptanalysis had a processing complexity of 2^{96}, linear cryptanalysis had a complexity of $2^{57.88}$, and the improved Davies' Attack was not even possible. Their final recommendations were to study s^5DES against any new cryptanalysis techniques, and use a longer secret key length in order to secure it from exhaustive search methods. More research continues on this subject.

7.0 Conclusion

S-boxes in DES-like encryption systems can be modified. However, a significant amount of mathematical knowledge, creativity, and understanding of cryptanalysis is needed for them to be secure. Although secret key size is important, a poor design subject to attacks described above can make any key length irrelevant. In addition, continued research in newer cryptanalysis methods is needed. The three attacks described in this paper are more computationally efficient than exhaustive search, yet they still require a significant amount of data to be performed. New techniques should focus on requiring fewer known plaintext/ciphertext pairs for the method to be successful.

References

1. J. Grabbe, “The DES Algorithm Illustrated,” <http://www.aci.net/kalliste/des.htm> (12 July 2002).

2. Bruce Schneider, *Applied Cryptology: protocols, algorithms, and source code in C, Second Edition* (John Wiley & Sons, Inc., 1996), 265-267, 289, 350.

3. “FIPS PUB 46-2,” *Federal Information Processing Standards Publication* (1993), <http://www.itl.nist.gov/fipspubs/fip46-2.htm> (14 July 2002).

4. “DES Encryption,” *Tropical Software*, <http://www.tropsoft.com/strongenc/des.html> (12 July 2002).

5. “Has DES been broken?” *RSA Laboratories' Frequently Asked Questions about Today's Cryptography, Version 4.1***,** <http://www.rsasecurity.com/rsalabs/faq/3-2-2.html> (12 July 2002).

6. Mitsuru Matsui, “Linear Cryptanalysis Method for DES Cipher,” *EUROCRYPT’93,* (1993).

7. A. Menezes, P. Van Oorschot, S. Vanstone, *Handbook of Applied Cryptology* (CRC Press, 1996), 258-259.

8. Eli Biham, “On Matsui’s Linear Cryptanalysis,” *Technion Technical Report CS0813* (Israel: 1994), 8-9.

9. Eli Biham, Adi Shamir, “Differential Cryptanalysis of DES-like Cryptosystems,” *Technical Report CS90-16* (Weizmann Institute of Science, 1990).

10. Biham, Shamir, “Differential Cryptanalysis,” 8.

11. Susan Landau, “Standing the Test of Time: The Data Encryption Standard,” *Notices of the American Mathematical Society* 47, no. 3 (March 2000), <http://www.ams.org/notices/200003/fea-landau.pdf> (16 July 2002).

12. Schneider, *Applied Cryptology*, 289.

13. Carlisle Adams, Stafford Tavares, “Designing S-boxes for Ciphers Resistant to Differential Cryptanalysis (Extended Abstract),” (Ontario, Canada: 1993).

14. Eli Biham, Alex Biryukov, “An Improvement of Davies’ Attack on DES.” *Journal of Cryptology* 10, no. 3 (1997). 195-205.

15. Schneider, *Applied Cryptology*, 296.

16. H. Feistel, “Cryptography and Computer Privacy.” *Scientific American* (1973). 15-23.

17. A. F. Webster, Stafford Tavares, “On the Design of S-Boxes,” *CRYPTO ’85* (1985).

18. Eli Biham, Alex Biryukov, “How to Strengthen DES using Existing Hardware,” *ASIACRYPT: International Conference on the Theory and Application of Cryptology* (1994).

19. Carlisle Adams, Stafford Tavares, “The Use of Bent Sequences to Achieve Higher-Order Strict Avalanche Criterion in S-Box Design,” *Technical Report TR 90-013* (Ontario, Canada: 1990). 5-6.

20. Adams, Tavares, “Designing S-boxes,” Conclusions.

21. Kwangjo Kim, “Construction of DES-like S-boxes based on Boolean Functions Satisfying the SAC,” *ASIACRYPT’91* (1991).

22. Kim, “Construction of DES-like,” Comparisons.

23. Kwangjo Kim, Sangjun Park, Sangjin Lee, “Reconstruction of s^2DES S-boxes and their Immunity to Differential Cryptanalysis,” *Joint Workshop on Information Security and Cryptology* (Korea: 1993).

24. Kwangjo Kim, Sangjin Lee, Sangjun Park, Daiki Lee, “How to Strengthen DES against Two Robust Attacks,” *Joint Workshop on Information Security and Cryptology* (Japan: 1995).

25. Kwangjo Kim, Sangjin Lee, Sangjun Park, Daiki Lee, “How to Strengthen DES against Three Robust Attacks,” *Electronics and Telecommunications Research Institute* (Korea: 1995).

Last Updated: October 24th, 2017

Upcoming SANS Training

Click Here for a full list of all Upcoming SANS Events by Location

SANS Seattle 2017	Seattle, WAUS	Oct 30, 2017 - Nov 04, 2017	Live Event
SANS Gulf Region 2017	Dubai, AE	Nov 04, 2017 - Nov 16, 2017	Live Event
SANS Amsterdam 2017	Amsterdam, NL	Nov 06, 2017 - Nov 11, 2017	Live Event
SANS Milan November 2017	Milan, IT	Nov 06, 2017 - Nov 11, 2017	Live Event
SANS Miami 2017	Miami, FLUS	Nov 06, 2017 - Nov 11, 2017	Live Event
SANS Paris November 2017	Paris, FR	Nov 13, 2017 - Nov 18, 2017	Live Event
Pen Test Hackfest Summit & Training 2017	Bethesda, MDUS	Nov 13, 2017 - Nov 20, 2017	Live Event
SANS Sydney 2017	Sydney, AU	Nov 13, 2017 - Nov 25, 2017	Live Event
GridEx IV 2017	Online,	Nov 15, 2017 - Nov 16, 2017	Live Event
SANS San Francisco Winter 2017	San Francisco, CAUS	Nov 27, 2017 - Dec 02, 2017	Live Event
SANS London November 2017	London, GB	Nov 27, 2017 - Dec 02, 2017	Live Event
SIEM & Tactical Analytics Summit & Training	Scottsdale, AZUS	Nov 28, 2017 - Dec 05, 2017	Live Event
SANS Khobar 2017	Khobar, SA	Dec 02, 2017 - Dec 07, 2017	Live Event
SANS Austin Winter 2017	Austin, TXUS	Dec 04, 2017 - Dec 09, 2017	Live Event
SANS Munich December 2017	Munich, DE	Dec 04, 2017 - Dec 09, 2017	Live Event
European Security Awareness Summit & Training 2017	London, GB	Dec 04, 2017 - Dec 07, 2017	Live Event
SANS Bangalore 2017	Bangalore, IN	Dec 11, 2017 - Dec 16, 2017	Live Event
SANS Frankfurt 2017	Frankfurt, DE	Dec 11, 2017 - Dec 16, 2017	Live Event
SANS Cyber Defense Initiative 2017	Washington, DCUS	Dec 12, 2017 - Dec 19, 2017	Live Event
SANS Security East 2018	New Orleans, LAUS	Jan 08, 2018 - Jan 13, 2018	Live Event
SANS SEC460: Enterprise Threat Beta	San Diego, CAUS	Jan 08, 2018 - Jan 13, 2018	Live Event
SANS Amsterdam January 2018	Amsterdam, NL	Jan 15, 2018 - Jan 20, 2018	Live Event
Northern VA Winter - Reston 2018	Reston, VAUS	Jan 15, 2018 - Jan 20, 2018	Live Event
SEC599: Defeat Advanced Adversaries	San Francisco, CAUS	Jan 15, 2018 - Jan 20, 2018	Live Event
SANS San Diego 2017	OnlineCAUS	Oct 30, 2017 - Nov 04, 2017	Live Event
SANS OnDemand	Books & MP3s OnlyUS	Anytime	Self Paced