DESnuts – (DES, Never Use This Scheme)

Breaking DES Encryption

By Cody Ohlsen (codyohl@uw.edu) - Dylan Johnson (dgj16@uw.edu) - Seokmin Kim (smkim95@uw.edu)

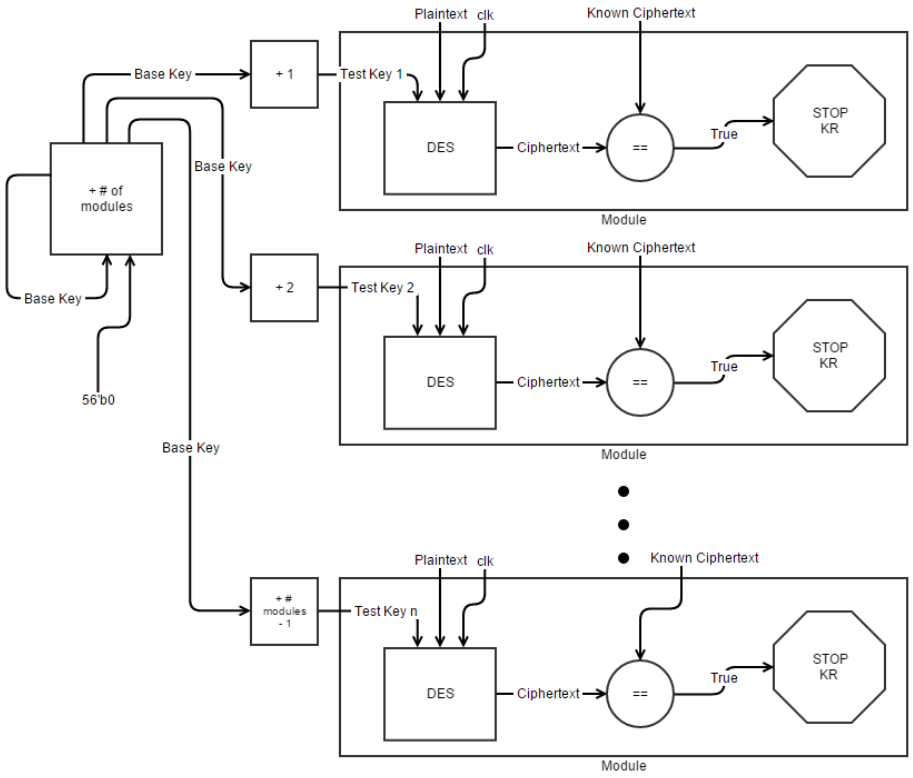
Abstract

The purpose of our project is to investigate past key recovery attacks on the single DES encryption scheme. We began by reviewing published papers on the security, history, and cracking of DES. We designed a suitable attack given the time and hardware resources we had available. Using the hardware description language Verilog, we implemented our design on a DE1-SoC FPGA. The benefits to a hardware based attack include speed and parallelism. By revisiting this key recovery attack, we demonstrate the transiency of once modern cryptographic schemes; what used to be considered the standard in cryptography is now obsolete and is easily broken using cheap and widely available hardware.

DESign

Our first design decision was to determine the characteristics of a fast and parallel DES encryption module. In the research paper, by Richard Clayton and Mike Bond, on "Using low-cost FPGA's to break DES encryption", a hardware based key recovery attack is described that utilizes a pipelined version of DES (1). This DES module takes advantage of the inherent parallelism found in DES by loading a key into the module every clock cycle. After 16 cycles, the encrypted plain text is seen on its output. Using a 50 Mhz clock on the DE1-SoC, we can theoretically check 50 million keys per second.

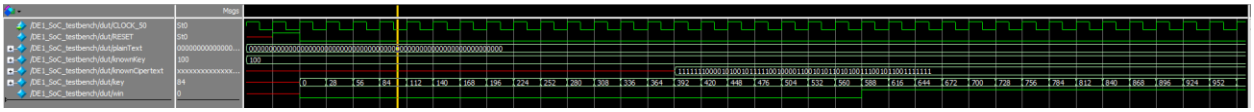
Additionally, we designed our system to exploit parallelism further by filling the logic array blocks on the DE1-SoC with multiple instantiations of DES modules. Each of the modules ran and cooperated together to recover the key. They also ran in synchronization with a base key, causing them to collaboratively search the key space. A high level model of how these DES modules work is shown below.



Results

The system was developed in Verilog and sent to the DE1-SoC Cyclone V FPGA. We implemented the system using as many of the pipelined DES module possible. 28 DES modules running together is the hardware limitation in our system – we ran out of logic array blocks on the FPGA at 29. With the system implemented, running at 50 MHz and testing 28 keys per clock cycle, it can test nearly 1.5 billion keys per second, which will recover any DES key by exhaustion after 606 days, with an expected KR time of 303 days. We were able to verify our system by physically loading a key using the 9 switches as input to the most significant bits of the key to be recovered. Our system ran as expected, and was able to reach benchmarks within a one percent margin of predicted times. Even for keys as high as 2^40, the system was able to recover the keys approximately in an hour.

Shown below, we have a waveform simulation of a KR with a key of the decimal value 100, taking place after only 4 clock cycles of key comparisons, or 1/12500000 of a second.



Future Work and Optimizations

There are a variety of alternative designs discussed that we believe could optimize and allow for even faster key recovery attacks on DES:

- The implementation and use of a 56 bit LFSR used as a key rather than a counter could possibly minimize the number of logic array blocks on the FPGA. This is because the use of many adders on the FPGA is expensive, and the order from 0 to 2^56 is not necessary, rather only checking each of these values in this sequence is. There are also other issues associated with this scheme, such as what to initialize the LFSR counters to. It is infeasible to be able to place each DES module evenly spaced apart because doing so requires computation to find the 2^56 states. However, placing the DES modules at random states provides the same expected value of KR on average case because the expected distance between DES modules is the same as if they were evenly spaced apart, under the assumption that the random function generates truly random 56 bit streams. However, in the worst case where all LFSR maps to the states right next to each other, the time it takes to break the DES is large because it is equivalent to running a single DES module.
- The use of a more expensive FPGA could easily cut the time on a large scale. For example, many FPGA systems, such as the Virtex 5, have onboard 100 MHz clocks. We believe that the pipelined DES implementation can reliably decrypt at this clock speed and this would immediately cut the KR time in half by testing keys at twice the speed. Additionally, the Virtex 5 provides over a factor of five number of logic gates, allowing for many more DES module instantiations and a higher key testing rate. With only a slightly more expensive FPGA, our DES cracking design could provide expected KR times within the course of a few weeks.
- Some DES KR schemes have been attempted with the use of a high-end GPU. This is useful because of the amount of parallelism that can be exploited and only a small amount of shared memory needed. We decided to avoid these schemes, because GPU's are not very affordable or easily accessible and usable for the average user and we wanted to be able to easily benchmark our KR rate through hardware clocking.

References

(1) Clayton, Richard, and Mike Bond. "Experience Using a Low-Cost FPGA Design to Crack DES Keys." *Using a Low-Cost FPGA Design to Crack DES Keys.* University of Cambridge. Web. 05 Mar. 2016. <http://www.cl.cam.ac.uk/~rnc1/descrack/DEScracker.html>.