COMP 360 Group Project:
RSA Implementation and Exploration

Matt Adelman, Evan Carmi, Adam Forbes

October 8, 2012

Brief History:

RSA is a public key encryption algorithm developed by Ron Rivest, Adi Shamir, and Leonard Adleman at MIT in 1977. Rivest and Shamir are both computer scientists that were working on an "unbreakable" public key encryption method. Rivest and Shamir worked on many different codes, and would pose them as a challenge to Adleman. Forty two of these codes were presented, and Adleman broke them all. Finally on attempty number 43, they created what is now known as the RSA scheme. Incidentally, English Professor CLifford Cocks developed the exact same encryption system in 1973, but it was classified as top-secret, so it was not released until 1997. The RSA algorithm was released for public domain in 1997.

The algorithm operates by using two distinct, large prime numbers to generate public and private keys. Anyone can use the public key to encrypt a message, but only someone with the private key can decrypt it. The algorithm is hard to break, because if the prime numbers are large enough, the factorization is exponential in time.

Implementation:

To further explore the RSA algorithm and the difficulties in an actual implementation we will write the algorithm in JavaScript, embedded in a web page. There has been some research done on web based implementations and (http://www-cs-students.stanford.edu/ tjw/jsbn/) may serve as a resource. Additionally, we will explore possible speed improvements, such as Chinese remainder theorem, and error checking. If possible, providing a visualization of the process would also be a goal of ours. In terms of difficulty implementation the algorithm shouldn't be too difficult, although further improvements and a clean interface may provide interesting challenges. Ideally we wouldn't simply replicate previous JavaScript implementation's of RSA, rather creating a new, clean, fast and explicative version which would provide the base of further experimentation.

Common Implementation Flaws:

Because the security implications of an improperly implemented RSA algorithm are so dire we would like to include within the scope of our project common mistakes made throughout the industry. These aren't trivial mistakes by any means, but rather exploitations of the properties of prime numbers, poor random and pseudorandom choices leading to weak seeds,

selecting seeds of low entropy, generating keys using the same factor, and other implementation flaws. A full understanding of the most common implementation mistakes is essential in crafting a strong algorithm that is up to contemporary cryptographic standards. Therefore we'd like to include this research as it not only serves as an interesting area of exploration in decryption, but also proves the strength of our algorithm.

Final Notes:

Additionally, we have all expressed interested in having this project serve as a example to show on our résumé or CV. To that end, having a project written in JavaScript, that runs in a web browser, would be both a portable and easily presentable choice of tools.

Our List of primary references is as follows:

http://en.wikipedia.org/wiki/RSA_(algorithm)
http://www-cs-students.stanford.edu/ tjw/jsbn/