Task 2.1

CVE-2015-7547

The glibc package provides common system code in one place, which can be shared by many programs. The package includes the standard c library, which includes functions all GNU/Linux programs are using, so every program is linked with that library. The libresolv library, included in glibc, provides the DNS functions for mapping domain names to IP addresses.

The vulnerability occurs in the send_dg() and send_vc() functions when calling getaddrinfo() for resolving a DNS domain name. The resolving response sends both A (IPv4) and AAAA (IPv6) DNS queries in parallel (dual A/AAAA DNS queries).

With specially crafted DNS responses an attacker could force a stack-based buffer overflow. Therefore a 2048 byte fixed size buffer for the DNS communication is located on the stack. But the attacker was able to write at most 65535 bytes to this buffer and overflow the stack. Then he could place an exploit on the overflowed stack and it is executed with the permissions of the user running the library.

NX – Non executable stack

The main idea is to separate the executable memory from the writable memory. Therefore the stack is only used for writing operation and any execution of code is forbidden. So if attack code is placed on the stack through an overflow, it cannot be executed. But a stack overflow is still possible and e.g. local variables can still be overwritten.

ASLR – Address Space Layout Randomization

The addresses of the stack and the linked libraries are randomized. So the attacker cannot execute his own code on the stack, because he does not know where it lays. Even other libraries can’t be used for exploiting, because their addresses are not accessible.

Task 2.2

Heartbleed

Missing bound check in the OpenSSL cryptography library of the Transport Layer Security protocol. With the heartbeat function, the client or server can check if the connection is still valid by sending a payload, with arbitrary content that will be send back.

For the exploit, an attacker sends e.g. 16 kB of data (payload + padding) to its opposition and states the size of the date is 64 kB (`payload`). The server writes the 16 kB size data onto the buffer (`pl`), but reserves 64 kB of size (`payload`).

```
buffer = OPENSSL_malloc( 1 + 2 + payload + padding);

bp = buffer;

memcpy(bp, pl, payload);
```
[1]

The response of the server is now 16 kB of the payload data plus 48 kB data that was also reserved and is lying behind the payload on the buffer (`pl`). This can be any kind of data (user names, passwords, keys…) that are currently in the buffer.

What/who is/was affected?

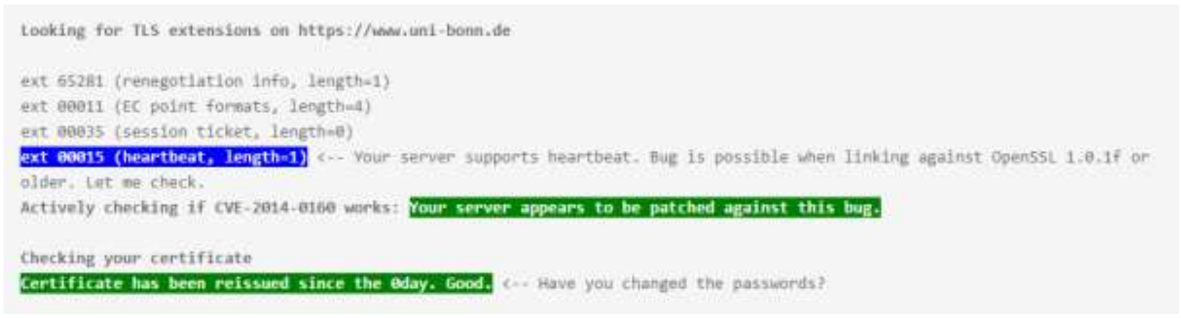
Since OpenSSL is so popular everyone could be direct or indirect affected. Every website using OpenSSL version 1.0.1 to 1.0.1f was concerned. The attacker could either access the private keys for secure communication and can try to observe the communication, or just encrypt the massive load of information with user sensitive information in it.

Are you affected (if so, how did you react)?

Of course I was affected. Changed the login credentials for every website that used OpenSSL. For my own website I just waited for the update, since there was no specific information on it (Just the basic configurations).

[1] https://git.openssl.org/gitweb/?p=openssl.git;a=blobdiff;f=ssl/d1_both.c;h=0a84f957118afa9804451add380eca4719a9765e;hp=89338e9430f2865f21a70da0e2c3fd4ec1a9dc35;hb=4817504d069b4c5082161b02a22116ad75f822b1;hpb=84b6e277d4f45487377d0159e82c356d750e1218

<u>May the university be affected?</u>



```
Looking for TLS extensions on https://www.uni-bonn.de

ext 65281 (renegotiation info, length=1)
ext 00011 (EC point formats, length=4)
ext 00035 (session ticket, length=0)
ext 00015 (heartbeat, length=1) <-- Your server supports heartbeat. Bug is possible when linking against OpenSSL 1.0.1f or
older. Let me check.
Actively checking if CVE-2014-0160 works: Your server appears to be patched against this bug.

Checking your certificate
Certificate has been reissued since the 0day. Good. <-- Have you changed the passwords?
```

<u>Which sources of information have you chosen and what do you think about the quality of the sources?</u>

https://www.owasp.org/index.php/Heartbleed_Bug

- Non-profit organization “Open Web Application Security Project”
- Often my first contact point in security concerns
- Same quality as Wikipedia for researching

https://cve.mitre.org/cgi-bin/cvename.cgi?name=cve-2014-0160

- Good source for cybersecurity vulnerabilities
- References include some useful websites with further information
- But not every link is always useful
- In it’s entirely you can obtain every information you need from the different sources

http://heartbleed.com/

- Linked to a profit organization
- Bunch of FAQs with answers
- Not useful on a technical level
- More a normal user information about the exploit than a technical specification

Task 2.3

The correct password is „cellophane“.

To run our code you can just run the „runHack.sh“ in the „Task2_3“ directory.
The script installs a python extension and runs the python script which ultimately shows the password.

Task 2.4

Part a) Dictionary Attacks:

1. The RockYou word list is a dictionary for possible passwords. It consists of passwords that were stolen or leaked. Therefore they seem to be legit to use for cracking passwords.

2. The most passwords are made up only from standart characters and are quite short. Some of them (the most) are chosen sloppy and are normal words or a combination of two words. In many cases using a regular library as a word list would be enough.

3. a "strong" password is a long password consisting of different charakters out of different charactersets without forming a regular word.

a "weak" password is short and does use only a single or few charactersets.

4. No its not.

Part b) Brute-Force Attacks:

The number of possible passwords is P^n so that increasing the length of the password n causes the number of possible passwords to grow faster.

A password with length 5 and P = U + L has about 254 million possible combinations.
Adding D to P makes it about 656 million possibilities.

Having P = U + L again and increase n by one (|P| = 48 and n = 6) increases the amount of possible passwords to 12 billion.

Task 2.5

the pcap file contains a record of network packets during a network communication.

The target seems to be a website or web application where it is mandatory to login.

The attacker is using a sql injection trying to get user information.

No, the attack did not ultimately compromis the target system. Therefore the leaked information must be used to get access to the System. Furthermore the attacker was not successfull i think. Usernames were leaked but the passwords not.

Securing the system from user input is mandatory to be safe from sql injections.