Cyber Security Base 2021 - University of Helsinki

Timo Schneider

August 18, 2022

Contents

Chapter I

Introduction to Cyber Security

1 Cyber security in the real world

What are the costs when dealing with cyber security attacks?

Cyber security attacks can be expensive. Especially since, once an attacker compromises your system, that attacker can then often steal multiple databases filled with sensitive information. In such a case the aftermath could be legal claims from customers, whose data have been stolen. Those would be direct costs.

Stealing is not the only form of damage that can be caused by an attack though. (Distributed) Denial of service attacks (DDoS) are just as bad. In this case, the attackers aim is not stealing valuable data to then sell it, but to cause you harm by shutting down the service that you are providing. This means that you loose money, with every minute the servers are down or compromised.

Both examples will have serious consequences, direct and indirect. After such an incident your future revenues could be influenced heavily. Customers may loose trust in your company and choose not to renew their contract with you. You can also see losses in the company stocks, if the company is present at the stock market. In both cases you will have to prevent future incidents and recover lost data and compromised servers. This also results in more costs. Insurances can help in those cases, however they come at a monthly/yearly price too, which would raise indirect costs.

The importance of cyber security and whose job it is

Technology is ever evolving and in recent years, software has overtaken hardware in the pace at which it is being revolutionized. And with every new piece of software, there come new ways of abusing it and new vulnerabilities to find. With an ever-growing pool of software products cyber security becomes more important as well, since it is vital to protect data from the end users.

Firstly to achieve a constant level of security in new software products, coders and developers need to receive better training, information and context on the subject. They often don't think about what their software shouldn't do, but focus too much on the features it should have. Confidentiality, integrity and availability should be prominent while developing new products.

However it is not solely the developers job to keep you safe, the end user himself plays a big role as well. Since technology keeps getting better, it is also -in some cases at least- harder to breach the security systems. In these cases, the easiest target is an employee or simply an end user. Therefore it is important to provide enough information and education to employees and end users, so they don't fall for phishing attacks or other social engineering attacks.

If a company would have its servers breached, what would be the possible reasons for the company not wanting to disclose the breach?

If someone breaches the servers of a company, said company would most likely not want to provide this information to the media, since such news are no longer minor articles in news reports. These reports are now front page stories that users and potential customers all around the world will read. Imagine the stress that a cyber attack causes. In such a case damage control should have first priority. Having to deal with all the external drama, such as news headlines and a crumbling reputation would make this scenario even worse.

In conclusion, a company that falls victim to a cyber attacks most likely want's to protect it's reputation and therefore wouldn't want to disclose this information to the media.

2 Identifying threats

2.1 Dealing with the human factor

Propose your own list on how to reduce the human factor when dealing with cyber security

As discussed, humans are a factor to consider when talking about cyber security. To reduce the risks posed by social engineering or sheer ignorance from employees, I am going to list 5 practices to improve security.

Firstly users should be able to identify sensitive data and handle it accordingly. It isn't all a matter of the right practices, but also about where and how to apply those practices. For this to work properly there should be a policy, which identifies different classifications and applies this to the data.

Secondly there should be a secure form of authentication implemented. Authentication is the act of validating a persons identity. This prevents users from accessing information that they don't have clearance for and aren't permitted to read or alter. This can be achieved by multiple-factor-authentication, which often requires something the user has (e.g., a smartphone or notebook), something the user knows (e.g., a password or PIN) or something the user is (e.g., bio metrics like a finger print).

After successful authentication there should be a second step, so the person trying to get access to the data is not only authenticated but also authorized. This should be conducted as an explicit check. The authorization depends on the privileges associated with the authenticated user account. It might also depend on the time of the request or the location of the user during this request.

One of the most important tools for having a secure system is the use of cryptography. This protects data from unauthorized access and therefore ensures confidentiality. Getting cryptography right however is no easy task.

Lastly the user should always be taken in account when designing or developing a system. This means that all necessary actions to keep the system working in the secure way it was designed to, should be easily manageable for the end user and sysadmins, who need to keep it updated.

In conclusion we see, that there are many steps to keeping a system secure, some responsibilities lie in the hands of developers others in the hands of the end user.

3 From Brain to Stuxnet: selected history of malware

3.1 Botnets

How can botnets be used for malicious purposes?

Botnets are very powerful tools to disrupt or compromise services of the victim. They can be used for many different attacks.

One of those attacks, probably the most common or most well known, is the DDoS attack. during a DDoS attack (or distributed denial of service attack) a server is bombarded with too many requests for it to handle, until it crashes. This causes the system to not be reachable anymore by actual users and in turn means, that the company running this server/service is loosing lots of money during that downtime.

Another form of attack (though a little more harmless) involving a botnet would be ad fraud. This means that the attacker or botmaster commands thousands of infected PCs to visit fraudulent websites, that were created by the criminals, and click ads, which are placed there. With every click, they get a small percentage of the ad fees.

Lastly I want to mention snooping. This technique provides the attackers with intelligence, by monitoring network traffic. By monitoring this traffic, the botnets can either passively gather intelligence on possible victims, steal credentials or actively inject malicious code into HTTP traffic. By Domain Name System (DNS) snooping the attackers can map IP addresses to domain names in order to discover what queries are being made, which domains might be good targets or what mis-typed domains might be worth purchasing.

In conclusion, this reveals how versatile botnets can be used and how dangerous they actually are. Since attacks aren't coming from one specific client, they are much harder to defend.

4 How does the internet work

4.1 Deep packet inspection

Describe 2 different scenarios where deep packet inspection can be used

Deep packet inspection is a technique to examine data to identify and filter out malware and other malicious or unwanted traffic. This method investigates not only the different headers of packets, but also the actual information and content that it carries. This tool is essential for advanced cyber security, since it can decide whether to let a packet through, or redirect it to another destination, based on different pre-set criteria.

The first scenario where this could be incredibly useful would be in an corporate environment. Deep packet inspection (DPI) can be vital in preventing worms, spyware and viruses from getting into the corporate network through work computers used by employees. DPI can detect these dangers even if common firewall techniques overlook them. Also, since this method is based on rules and policies defined by yourself, it can also detect any prohibited uses of an application.

It can also be used by network managers to help ease the flow of network traffic. For example, you could set up policies, which let high priority messages pass through instantly, over lower priority messages. Or prioritize packets, which are critical to use cases or projects over common browsing packets.

These to scenarios make it apparent, that DPI is a great improvement over traditional routers, which don't inspect packets above the transport layer. However there are also some concerns about privacy, since this inspection can easily detect recipients and sender of content.

5 Encryption algorithms

Explain why the Caesar cipher is not very secure

The Caesar cipher is considered to be one of the weakest forms of encryption - it is also one of the oldest forms there are. This is because there is only a very limited number of shifts available. The common English alphabet has 26 letters. Since the Caesar cipher works by shifting letters, you can only move by up to 26 digits, however moving it by 26 doesn't make any sense, since you would end up with the original letter and the encryption wouldn't be working as it would just be the original plain text. So you have 25 possible integers to shift by. Since, to decrypt the message, we only need to now by what number the letters have been shifted, we can easily try out 25 different variations on even just one or two words of the message until we get something that sounds right.

Now if we also know the language in which the original message has been authored we can already start guessing words without even needing to try out different shifts. If we can guess a word right, then we can just apply the corresponding shift to all other words and have successfully decrypted the message.

Let's look at an example:

Plain Text: I am studying Data Encryption Key: 4 Output: M eq wxyhCmrk Hexe IrgvCtxmsr

Imagine we know, that the original message is written in English, we can already almost certainly conclude, that the first to words are 'I am'. The very first word only has one character, the only two one-letter-words in English are 'a' and 'I' so two tries and we decrypted the whole message.

One modification to improve this cipher could be different shifts, for different characters of the message. For example instead of shifting every letter by 4, we could shift the first letter by 4, second by 23, third by 15 and then repeat the cycle. The key could be a plain text word, which corresponds to the shifts 4, 23, 15 ('EXP' would work if a=0, b=1, and so on..). This key obviously has to be transmitted separately from the message itself, otherwise it would be easily cracked.

Chapter II

Securing Software

1 Part 1

1.1 Ports and Applications

Portscanner

```python
#!/usr/bin/env python3
import sys
import socket


def get_accessible_ports(address, min_port, max_port):
  found_ports = []

  s = socket.socket()
  for i in range(min_port, max_port+1):
    result = s.connect_ex((address,i))
    if result == 0:
      found_ports.append(i)
  #result = s.connect_ex((address, max_port))
  #if result == 0:
  #   found_ports.append(max_port)
  return found_ports


def main(argv):
  address = sys.argv[1]
  min_port = int(sys.argv[2])
  max_port = int(sys.argv[3])
  ports = get_accessible_ports(address, min_port, max_port)
  for p in ports:
    print(p)

# This makes sure the main function is not called immediatedly
# when TMC imports this module
if __name__ == "__main__":
  if len(sys.argv) != 4:
    print('usage: python %s address min_port max_port' % sys.argv[0])
  else:
    main(sys.argv)
```

Listing II.1: write the get_accesible_ports method

1.2 Web Servers and Web Applications

Hello Web!

```
#!/usr/bin/env python3
from django.http import HttpResponse

def homePageView(request):
    return HttpResponse('Hello Web!')
```

Listing II.2: create a server response

Calculator

```
#!/usr/bin/env python3

from django.http import HttpResponse


# Create your views here.

def addPageView(request):
        first = int(request.GET.get('first'))
        second = int(request.GET.get('second'))
        return HttpResponse(first+second)


def multiplyPageView(request):
        first = int(request.GET.get('first'))
        second = int(request.GET.get('second'))
        return HttpResponse(first*second)
```

Listing II.3: add or multiply the first and second parameter

Hello Templates!

```

from django.http import HttpResponse
from django.template import loader


# Create your views here.

def homePageView(request):
    template = loader.get_template('pages/index.html')
    return HttpResponse(template.render())

def videoPageView(request):
    template = loader.get_template('pages/video.html')
    return HttpResponse(template.render())
```

Listing II.4: return content using a template

Hello List!

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
    <head>
        <title>Hello List!</title>
    </head>
    <body>

        <h1>Hello List!</h1>

        <ul>
            {% for item in itmes %}
            <li>{{item}}</li>
            {% endfor %}
        </ul>

        <form action="/" method="POST">
            {% csrf_token %}
            <input type="text" name="content"/>
            <input type="submit"/>
        </form>

    </body>
</html>
```

Listing II.5: print the contents of a list

Notebook

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
    <head>
        <title>Hello Notes!</title>
    </head>

    <body>
        <h1>Hello Notes!</h1>

        <ul>
            {% for item in items %}
            <li>{{item}}</li>
            {% endfor %}
    </ul>



        <h2>Add content to list</h2>
        <form action="/" method="POST">
            {% csrf_token %}
            <input type="text" name="content"/>
            <input type="submit"/>
        </form>

        <h2>Clear everything</h2>
        <form action="/erase" method="POST" name="content">
            {% csrf_token %}
            <input type="submit" value="erase" />
        </form>

    </body>
</html>
```

Listing II.6: template for the notebook application

```
from django.shortcuts import render

# Create your views here.

def addPageView(request):
  items = request.session.get('items', [])
  item = request.POST.get('content', '').strip()
  if len(item) > 0:
    items.append(item)
  if len(items) > 10:
    del items[0]
  request.session['items'] = items
  return render(request, 'pages/index.html', {'items' : items})


def erasePageView(request):
  items = []
  request.session['items'] = items
  return render(request, 'pages/index.html', {'items' : items})


def homePageView(request):
  # use sessions (the data is stored in a database db.sqlite that is then accessed
    using a cookie)
  items = request.session.get('items', [])

  # shorter way of writing instead of loader
  return render(request, 'pages/index.html', {'items' : items})
```

Listing II.7: create a notebook application

2 Part 2

2.1 Web Applications continued...

[TBC]