IT-UNIVERSITETET I KØBENHAVN

Forside til skriftligt arbejde/projekt

Kursusafleveringer

Kursus ved ITU-studienævn	Kursus ved EBUSS-studienævn
□	□ Synopsis □ Miniprojekt □ Anden skriftlig opgave

Projekt (med projektaftale)

ITU-hovedvejleder	EBUSS-hovedvejleder
□ Speciale □ Afsluttende projekt ☑ Projekt	□ Speciale □ Projekt □ Sommerprojekt □ 4-ugers projekt □ 12-ugers projekt □ 16-ugers projekt

Titel på kursus, projekt eller speciale:

ITU Infrastructure Analysis

Kursusansvarlig/Vejleder(e):

Joseph Roland Kiniry

Navn(e):	Fødselsdato og år:	ITU-mail:
1. Ian Qvist	16-09-85	ianq @
2. Michael Bisbjerg	04-10-91	mgre @
3.		@
4.		@
5.		@
6.		@
7.		@

Kun for kurser med e-portfolio:

Link til e-portfolio:

Table of Contents

Abstract

In the digital age with global access to the Internet, security is more important than ever. Since the rise of the information age, we are storing more and more confidential information on servers. As ordinary human beings, we hope the data will remain secure and accessible to only those who need it; however, the truth is that the data is stored on a server that is accessible by millions of people. We therefore we put our data at a risk of hackers destroying, or worse; abuse the data.

With that in mind, we will present a real attack scenario against the IT University of Copenhagen, the methods that were used and how it resulted in a total takeover of the networks, servers and data. The results we present also made it possible to gain physical access to all restricted areas that requires a key card.

Introduction

In the wake of recent data breaches of medium[1] to large companies[2], it has become obvious that they put themselves at a risk of being hacked. With outdated infrastructures, missing security policies and sloppy handling of data, they are at a security disadvantage and therefore a high priority target for hackers. In this paper, we will focus on the security of the infrastructure at the IT University of Copenhagen (ITU) and how we can exploit it using advanced hacking techniques. We will exploit security vulnerabilities, crack passwords and traverse networks in order to expose the data they protect and further gain access to closed systems in order to take full control of the infrastructure.

[1] http://datalossdb.org/incidents/7005-2-4-million-voters-names-addresses-genders-dates-of-birth-and-voting- information-compromised-by-disappearance-of-two-usb-memory-drives

[2] http://datalossdb.org/organizations/5806-shanghai-roadway-d-b-marketing-services-co-ltd

Data Protection and Compromise Policy

An agreement was made with the security board that any attacks that could disrupt the normal functionality of the services that ITU was offering, had to be approved before being executed under the supervision of the network staff. We took on a read-only policy where no changes of any kind can be made to the systems. Each of the items in our policy is described in details underneath here.

Read Only Policy

Any changes to the compromised systems were strictly prohibited. This includes, but is not limited to, changes to configuration files, uploading of malware and altering files that could cause disruption to any of the services running on the machine.

Secure Storage of Collected Data

Any data that was to be collected, had to be encrypted and stored in on redundant hard drives. Transferring of the data between machines was done using TLS with AES 256 bit encryption[3] and the data arrived in a data container that used AES 256 bit encryption[4] with a very strong password to further secure the data.

Minimum Damage Policy

When intrusive attacks were allowed, we were using a minimum damage policy that ensures that whatever damage that could come from the attack, was limited to a small area or service. This included, but was not limited to; heavy SQL queries, long running tasks and DoS based buffer overflows.

[3] https://www.dropbox.com/help/27/en

[4] https://boxcryptor.desk.com/customer/portal/articles/565947-how-does-boxcryptor-encrypt-files-

The Importance of Security

Recently a lot of universities have been the target of malicious hackers. The San Jose State University[5], University of Florida[6] and Yale University[7] are among the latest to get hacked and their data sold on the underground hacker markets. The universities mentioned only lost a few thousand records, which is nothing compared to recent breaches of Yahoo[8] and Nvidia[9] where 453,492 and 420,000 records were stolen. However, being a university, they do not have the financial resources and staff to immediately investigate the breach. They become subject to long running security investigations that disrupts the normal operation, rendering the University unable to accommodate the students.

Network Overview

The first phase of the attack was to collect information about the systems we were to attack. The initial gathering of hostnames resulted in a total of 236 unique host names along with a handful aliases and related domains, on 12 different network ranges (internal and external).

Internal network scanning resulted in the discovery of the following network equipment:

- Sagio authentication system
- ScanCom printing system
- Aerohive wireless routers
- ESX based virtualization servers
- Client machines connected to the wired and wireless networks
- IP based surveillance systems
- Dot screens display systems

Each system had a network range by themselves and thus correctly segregated from the client network. We identified usernames, network paths, OS and software versions from the metadata of 3192 files (docx, pdf, ppt and others) available on search engines like Google and Bing, with that information we identified high priority servers and started to attack them first. As each server was compromised, we tried to traverse to other networks and servers using the credentials we gathered.

[5] http://www.databreaches.net/?p=24644

[6] http://www.databreaches.net/?p=24626

[7] http://www.databreaches.net/?p=24818

[8] http://datalossdb.org/incidents/6919-453-492-email-addresses-and-passwords-dumped-on-the-internet

[9] http://datalossdb.org/incidents/6925-400-000-user-names-email-addresses-and-hashed-passwords-dumped -on-the-internet

Security Vulnerabilities

This section describes the highest priority security vulnerabilities found. They were the main reason we could traverse the network and compromise servers that we otherwise would not have access to. Each vulnerability is described and the results of the exploit are presented before we give some guidance on how the vulnerability could be remedied.

Windows NTLM Pass the Hash

On order to improve security of Windows, Microsoft has implemented the NTLM protocol that sends an encrypted NTLMv2 hash instead of the clear text password when authenticating. As a convenience feature, if the credentials on a client are identical to the one on the server, the client is automatically logged into the server, without being asked for a password.

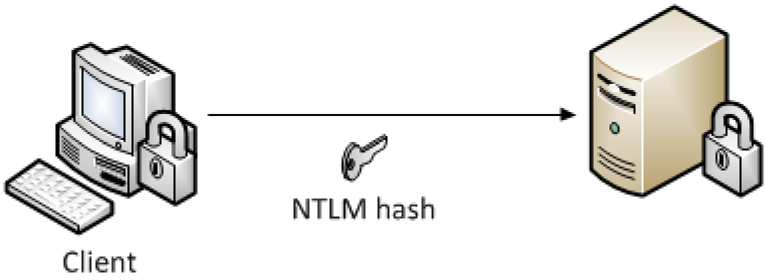


Figure 1 – NTLM authentication

It is possible to decrypt the NTLM client session and obtain the hash. It is also possible to extract the hash from an encrypted SAM[10] database on a compromised machine. Using an attack technique called "Pass the Hash", together with an obtained NTLM hash, we are able to authenticate to multiple servers on the network without ever being asked for the password.

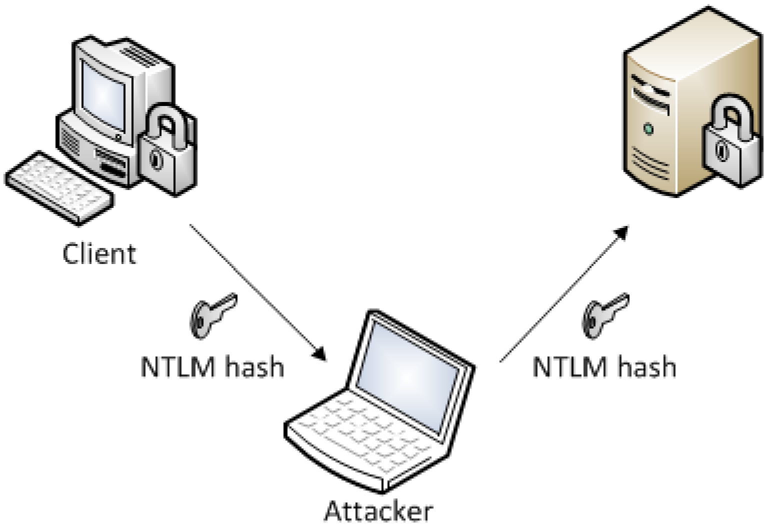


Figure 2 - MitM attack on NTLM authentication

We utilized this attack type to access hidden administrative shares on the network and gather information about users, other networks and systems. By chaining this attack, we were able to compromise most of the Microsoft Windows based servers.

Recommended Solution

NTLM is an insecure protocol[11] that is used under the following circumstances:

- The client is authenticating to a server using an IP address.
- The client is authenticating to a server that belongs to a different Active Directory forest that has a legacy NTLM trust instead of a transitive inter-forest trust
- The client is authenticating to a server that doesn't belong to a domain.
- Where a firewall would otherwise restrict the ports required by Kerberos

It is recommended to deactivate legacy LM and NTLM authentication by editing the domain group policy[12] and set it to "Send NTLMv2 response only. Refuse LM & NTLM"

To further secure the network against NTLM attacks, servers and clients should be a member of a domain to enable Kerberos authentication, and local access to the systems should be restricted or completely disabled.

[10] http://wiki.answers.com/Q/What_is_the_SAM_database

[11] http://www.ampliasecurity.com/research/OCHOA-2010-0209.txt
[12] http://msdn.microsoft.com/en-us/library/ms814176.aspx

MySQL Backwards Compatibility Pass the Hash

We obtained database credentials to a MySQL based server through misconfigured permissions on shared drives. With the credentials, we extracted 4447 users and password hashes from the database. It turned out that the hashes were in the older (version 3.23 format) that are much more insecure than newer (post 4.1 format) hashes. Up to version 4.1, they used a homemade scramble function[13] that basically scrambles the input password. The output is an 8 bytes long scrambled text and stored as a hexadecimal string.

MySQL changed the password scheme in October 2004[14] with the release of version 4.1 of their database server. They changed the hashing scheme to the more cryptographically secure 20 byte SHA-1[15] hashing algorithm. But to support existing systems, they provided an internal function named "OLD_PASSWORD()" which allowed you to hash with the old format. They also provided a configuration option called "--old-passwords"[16]. If a MySQL instance is run with this option enabled, all generated passwords are stored in the old 3.23 format.

Below is an example password hashed using the two methods, the password is "password":

- Old format: 5D2E19393CC5EF67
- New format: *2470C0C06DEE42FD1618BB99005ADCA2EC9D1E19

Besides the fact that the 3.23 hashing format is insecure, the protocol in which they are exchanged over the network is also insecure. Using this we gained access to several MySQL servers and their data.

Recommended Solution

The old hashing scheme should be disabled and all users upgrade to the newer MySQL version 4.1 format. The old scheme was marked deprecated 7 years ago and it is recommended to disable it. To maintain backwards compatibility, see the MySQL documentation[17] on the subject.

Internal database servers should be firewalled (see **Misconfigured Firewalls**) so that they are not reachable from the client network. Permissions should also be maintained in such a way that lower privileged users can extract password hashes or read databases they do not own.

[13] http://www.laszlo.nu/post/322433762/old-password
[14] http://en.wikipedia.org/wiki/MySQL#Product_history
[15] http://dev.mysql.com/doc/refman/5.0/en/password-hashing.html
[16] http://dev.mysql.com/doc/refman/5.1/en/server-options.html#option_mysqld_old-passwords
[17] http://dev.mysql.com/doc/refman/5.0/en/password-hashing.html

Sagio MIFARE Classic

We investigated the ITU's on-premise building security, and we found several security flaws. ITU uses an access-card based system where all external (and most internal) doors are protected by an electronic lock with a card reader.

The cards are MIFARE Classic[18] 1k cards with 1 KB secure internal memory allowing for various uses (such as the printing system). The internal memory is protected by the cards on-board chip which will determine if a read or write can happen based on cryptographic keys and access permissions. Built in 1994[19], they are based on quite old technology and in 2007[20] they were proven to be cryptographically broken. It has since then been advised not to use the MIFARE Classic cards, yet they are still in widespread use today[21]

We consulted Christian Panton[22], an expert on the MIFARE technology on the ITU key card system to determine the security of the system. It was concluded that even with proper use of the cards, they are too insecure to use for anything else than added security on doors already protected by keys. We found that Sagio does not use the cards in a secure manner, but have instead completely disabled the security that the MIFARE card offers.

The MIFARE Classic card contains two types of information:

- **The public ID**
  Considered unique and set by the manufacturer. It is not possible on commercial cards to write to this area. The normal assumption is that this area is safe and trustworthy. However, in recent years, cards that can change their UID have been available[23]. This invalidates the assumption that the UID is trustworthy.

- **The encrypted storage**
  Protected by keys which need to be presented when reading or writing to the card. Can be both read from and written to, and the keys can be set programmatically. It is normal for an application to keep track of the keys for each card, and then use these to read and write.

Using readily available equipment (mainly an RFID reader/writer from Touchatag, however the company is currently going out of business), we were able to crack the keys used to read and write to our MIFARE Classic access cards. In the process we also found that the same read keys were used on all cards - simplifying the process of reading cards later on.

[18] http://en.wikipedia.org/wiki/MIFARE
[19] http://en.wikipedia.org/wiki/MIFARE#History
[20] http://en.wikipedia.org/wiki/MIFARE#Security_of_MIFARE_Classic
[21] http://en.wikipedia.org/wiki/MIFARE#Places_that_use_MIFARE_technology
[22] http://christian.panton.org/
[23] http://www.xfpga.com/P_view.asp?pid=384

Using these keys, we are able to alter the data on the cards, and read their state. With this knowledge, we mapped what the cards are actually used for in an effort to see what the data means. The ITU cards are used for two things:

- Access system
- Printing system

We found that the authentication system is no more secure than writing your key on a piece of paper, and then use that as a proof of who you are. The printing system itself uses the encrypted key card data, and is therefore used correctly.

We were able to build a system that in less than a second, would record the authentication data of any card we put in front of our reader. We were then able to copy cards very quickly, which could be used together with social engineering[24] to obtain access to all areas of the building. However, it is worth noting that the pin code is still needed for the copied card, but a pin code can easily be copied by observing the target entering the pin code or access systems that does not require a pin code.

After some research, we found that the access rules are not set up correctly and security zones can overlap. Some of the zones require pin codes, others do not. Inside the overlapping zones, one could access a pin code restricted area from a restricted area that does not require a pin code.

Recommended Solution

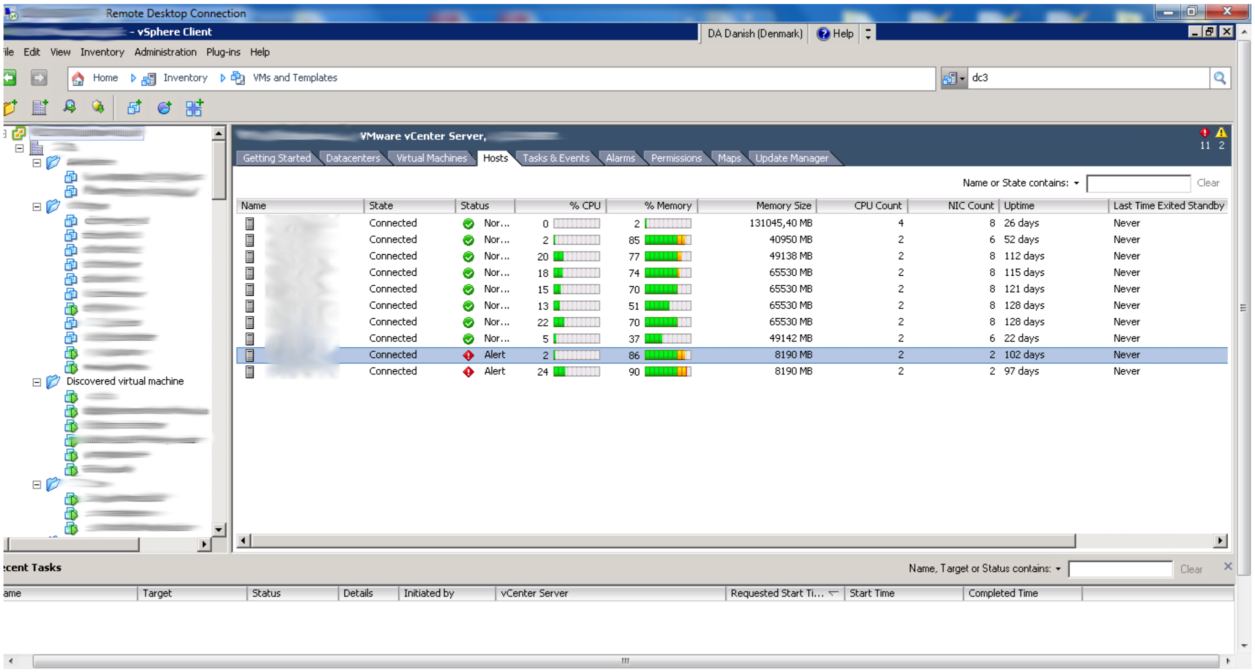
The system in its current state is being used in a very insecure way. It is recommended to contact Sagio and hear what opportunities they offer. It is also recommended to review the access zones of the entire premises, as we have observed that not all paths to an area require the same form of authentication. Some paths may require pin code, while others do not. We realize that Sagio has declared bankruptcy on the 21st of May 2012, but that they also have recovered from this around the 11th of June 2012 (21 days later)[25].

[24] http://en.wikipedia.org/wiki/Social_engineering_%28security%29
[25] http://www.securityworldhotel.com/dk/news.asp?id=59852

VMware vCenter Server Authentication

ITU uses VMware ESX as their virtualization environment, allowing them to quickly commission and decommission servers and storage. This gives them the ability to operate and maintain their network. A small organization may have a single so-called “virtualization machine” - a barebone machine with lots of computing power. These machines will then run virtualized instances, which can then be controlled remotely.

To control these servers, you use the vSphere client, which allows you to remotely manage all the virtualized instances. This tool is perfect when considering flexibility and maintainability. But by obtaining credentials to the vSphere management interface, an attacker will be able to control each and every virtualized server, as if he had physical access. This includes being able to turn servers off, delete instances and even commission new servers for various intents and purposes.



Name	State	Status	% CPU	% Memory	Memory Size	CPU Count	NIC Count	Uptime	Last Time Exited Standby
[redacted]	Connected	✓ Nor...	0	2	131045,40 MB	4	8	26 days	Never
[redacted]	Connected	✓ Nor...	2	85	40950 MB	2	6	52 days	Never
[redacted]	Connected	✓ Nor...	20	77	49138 MB	2	8	112 days	Never
[redacted]	Connected	✓ Nor...	18	74	65530 MB	2	8	115 days	Never
[redacted]	Connected	✓ Nor...	15	70	65530 MB	2	8	121 days	Never
[redacted]	Connected	✓ Nor...	13	51	65530 MB	2	8	128 days	Never
[redacted]	Connected	✓ Nor...	22	70	65530 MB	2	8	128 days	Never
[redacted]	Connected	✓ Nor...	5	37	49142 MB	2	6	22 days	Never
[redacted]	Connected	◆ Alert	2	86	8190 MB	2	2	102 days	Never
[redacted]	Connected	◆ Alert	24	90	8190 MB	2	2	97 days	Never

Figure 3 - vCenter virtualization servers inside the vSphere client

We were able to obtain access to a vSphere management interface using local Windows credentials from one of the compromised servers. This gave us full control over the network and full access to all the confidential data that reside on the servers. We found that ITU has several virtualization servers, as we can see in this screenshot above. Each of those 10 servers holds one or more virtual machines, which in turn can be controlled in the pane to the left.

Looking around we found several server consoles that were already logged in with administrator credentials. This could allow a potential hacker to access resources without being noticed. This holds true for several servers we browsed, indicating a general problem with the security policy.

Recommended Solution

Since VMware products are built on Linux with PAM enabled, you can switch the authentication modules to use an internal authentication server like an Active Directory server. This shifts the security to the authentication server instead of the local account database on the ESX server and improves security at the same time.

It should be considered a high priority task to implement a good firewall (see **Misconfigured Firewalls**) infrastructure that isolates the ESX server from the rest of the network, and also only accept authentication requests from a small amount of trusted IP addresses on the internal network.

Misconfigured Firewalls

We found that the ITU firewalls are misconfigured in such a way that the greater part of the network is available to everyone. One example of such a place is the tables in the Atrium, all of which have two different networks available: one on the inside of the network, and one outside.

It turns out that on the inside of the network (despite still being in the student's area), we were able to communicate with many of the ITU's internal servers without problems. From the outside, these servers were correctly firewalled. However, the students and staff have access to the inside of the network through a number of services, such as SSH, MySQL, RDP and VPN.

We commend that ITU protect all the internal servers in the best possible way. For example by segregating these servers into a separate network, only available to themselves, while being unreachable from the internet. At the same time, servers that are meant to be public should be available, but in a manner that won't compromise the internal servers. Generally, networks are divided into four separate zones as described below:

- **The public zone**
  This zone represents the internet, and is out of the ITU's control.

- **The private zone**
  This zone represents all faculty staff, all students and all offices at the ITU. This zone contains all clients through wired and wireless connections, making no distinction between them. The private zone can further be split into several smaller networks; one for offices, one for students and so on.

- **The DMZ zone**
  This zone contains all machines that need to accept incoming connections. This includes, but is not limited to: Web servers, remote shell servers, student databases and so on. Only public information that anyone can access is placed in this zone.

- **The administrative zone**
  This zone contains all internal servers that shouldn't be contacted from the outside world or the private network. This includes amongst others, the VMware ESX servers and backup systems.

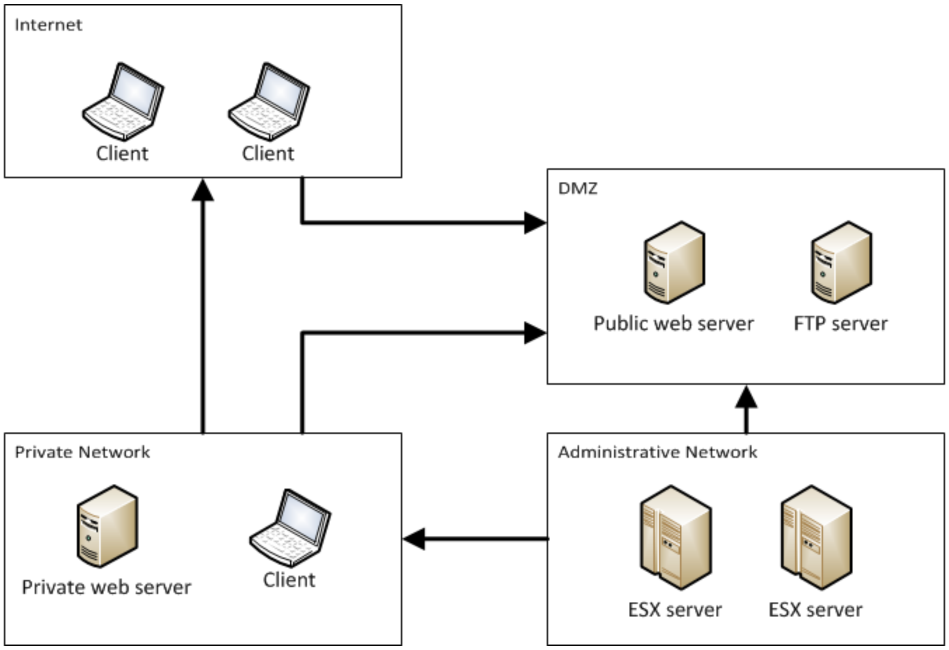


Figure 4 – Network Zone Diagram

As we can see in the picture above, there are some rules to follow. For example, no server in the private zone can receive connections from the internet. Likewise, the private zone can't connect to the ESX servers - at all.

Recommended Solution

We recommend that ITU switch to this kind of network structure (it already have the basics implemented, but it needs to be fully implemented) to improve security of the whole network.

In addition to the zones, it should be considered if some servers need extra protection via. IP restrictions. For example a database in the DMZ zone shouldn't be accessible from other machines than the web server that is accessing it. If an attacker were to take over one of the servers in the DMZ, he would have to hack the web server in order to gain access to the DB server. At the moment, it does not matter which server is hacked, as all of them have access to the database.

Data storage

The ITU IT infrastructure is composed of many servers, remote storages (SAN) and even more clients. Due to this, it is essential to have policies in place that protect the infrastructure, against human error and intruders. These policies include everything from how to handle data, to how decisions are made. One of the areas where it is essential to have clear guidelines, is where backups should be performed correctly and securely (preferably inaccessible by all other machines - see **Firewalls**).

We found such policies to be lacking during our research, as we found several servers containing residue of partial backups and old files - the backups contained confidential data such as SSL certificates, passwords, CPR numbers and more. We did, however, never encounter any encryption or secure storage - instead, everything was readily accessible.

It is also important to consider cases with running systems, where both redundancy and availability is a necessity. We didn’t, in our research, discover any redundant systems or failovers. The lack of such systems effectively renders the ITU services unavailable or otherwise unavailable if an attack or unfortunate event was to occur. These events could take place in times where the IT systems were required, for example during exams.

We found, in our research, that due to a misconfigured firewall setup (see **Misconfigured Firewalls**), we were able to access most servers in the IT infrastructure directly - effectively allowing us or a hacker to extract data much easier than if we had to go through a hardened web server. This helps protect that data, even if it resides in a DMZ zone. From a pure systems development point of view, we also found the same data on multiple servers, in different usage scenarios. This way of handling data complicates many things, such as keeping different systems in sync, while also making it harder to secure from abuse.

Recommended Solution

We recommend that the ITU take action on the problems here, as some of them should be considered critical. We've outlined different points that should be acted upon:

Encrypting confidential data

According to Danish law[26], institutions have a responsibility when it comes to protecting the personal information of any employees, and in the case of ITU, also its students. CPR numbers are considered confidential data and should[27] be stored in a secure way, to limit the risk of unauthorized users getting access to the data. Our recommendation is that the ITU store confidential data on a server that communicates over encrypted channels with a strong password policy to further improve security (see **Password Analysis**). If the content of CPR numbers are not needed for anything else than verification, it is recommended that they are stored in a secure hashed format. This also applies to other kinds of information that only requires verification, such as passwords.

Server backups

Server backups should be treated with the same confidentiality as the servers, as they contain the same information as the original server from which it came. We recommend that backups are consolidated to a single server system in a secure environment (see **Misconfigured Firewalls**). They should be stored in an encrypted format[28] with AES 128 bit or better symmetric encryption algorithm. Database, web and file servers should be revised and purged from old backup data and an up to date backup of said servers should be performed.

Purge obsolete servers

We've discovered multiple active servers which seem to have come out of service or have otherwise become obsolete. These servers pose a risk, as they may contain residue data that hasn't been removed. Instead of using time on cleaning these, they should be put out of service and the data destroyed.

[26] https://www.retsinformation.dk/forms/r0710.aspx?id=828

[27] https://www.retsinformation.dk/Forms/R0710.aspx?id=842

[28] https://www.owasp.org/index.php/Top_10_2010-A7

Consolidate common data

A big issue found was common data stored in multiple locations with varying formats and security levels. As mentioned previously, confidential data must be stored at a minimum security level, requiring extra concern. Other data might also have to be kept in sync, giving extra issues when maintaining the network. An idea might be to consolidate all this data on a single installation, simplifying many tasks such as sync and security.

Employ confidentiality levels

There are examples of data confidentiality levels in effect today, most base their policies on the CIA[29] model (confidentiality, integrity and availability). This model explains the core elements of data storage and -handling. An example of its usage can be found at the Texas State University[30].

Minimize dataset

We have found that several servers may contain the same duplicate data (see **Data Sheet**), and that this doesn't just cover CPR numbers, but also addresses and names. It is generally a good idea to employ a policy of storing the least possible amount of data. This means that only the necessary data, meaning that if a server is compromised, not everything is lost.

It is common for public institutions (schools, hospitals etc.) to identify people using a CPR number today. Instead of using that, another unique key should be used, for example at the ITU every person has a unique shorthand name.

[29] http://en.wikipedia.org/wiki/CIA_triad

[30] http://www.utexas.edu/cio/policies/pdfs/Data%20Classification%20Standard.pdf

Password Analysis

Traversing the internal network, we came across a lot of servers that stored passwords in different formats for different services. We collected the following hashes in order to analyze the general password strength:

- Microsoft Windows hashes (NT+LM): 10754
- OpenLDAP hashes (SSHA-1): 1037
- MySQL hashes (MySQL 3.23): 4523
- Apache htpasswd hashes (DES): 160
- Wordpress hashes (PHPass MD5): 3121

A short description of each hash and their usage is found below.

Microsoft Windows NTLM

Microsoft LanManager (LM) was used throughout 1990 before being replaced by the NT LanManager (NTLM) back when Windows NT was introduced. NTLM v2 was introduced with Windows NT SP4 and fully supported by Windows 2000. NTLMv2 now serves as the default authentication method when a computer is not attached to a Windows Active Directory domain. However, to keep backwards compatibility with older systems, the old LanManager (LM) authentication was kept up until the introduction of Windows Vista where it is turned off by default.

It is recommended by Microsoft that NTLM support is removed due to its usage of deprecated security protocols. Instead, companies should deploy an Active Directory domain that uses the much more secure Kerberos protocol as authentication.

In our research, we found that ITU were still using NTLM hashes with the LM hashes activated. This is highly attractive by hackers as LM hashes can be cracked in a very short period of time[31] and then used as a template for cracking the NT hash.

[31] http://en.wikipedia.org/wiki/LM_hash#Security_weaknesses

OpenLDAP SSHA-1

OpenLDAP uses the RFC2307[32] hashing scheme called SSHA that is based on a salted SHA-1[33] hashing algorithm. The famous cryptographer Bruce Schneider announced[34] back in 2005 that the SHA-1 hashing algorithm had been broken. The security of the SHA-1 algorithm was severely reduced and should never be implemented in new setups.

MySQL 3.23 Hashing Scheme

MySQL 3.23 used a home-made hashing scheme that is only recommended enabled in newer versions of MySQL[35] to have backwards compatibility with older clients. Because the hashing algorithm is very simple and not implemented using common security practices, it is very easy to crack.

Apache htpasswd DES Hashes

Apache uses htpasswd to create hashes for its Basic Authentication module. By default it generates MD5 hashes, but it is also possible to create DES hashes. The DES hashing algorithm was published back in 1977 and can have 2 character random salt to further improve security. However, due to the small key size (56 bits) it is recommended to use something that is stronger.

Wordpress PHPass Hashing Scheme

PHPass is a portable PHP hashing framework[36] created by the author of the famous John the Ripper password cracker and it is designed to create highly secure passwords. Wordpress implemented PHPass with the MD5 algorithm in version 2.5 and newer. The MD5 hash is both salted and implemented with a variable iteration count to further improve security.

Password Cracking

Using a single computer, we were able to crack 90.82% of all found hashes within 24 hours. The Wordpress hashes were excluded from the cracking as they are simply too secure to crack on current hardware. Below is a list of each hash type and how many percent were cracked:

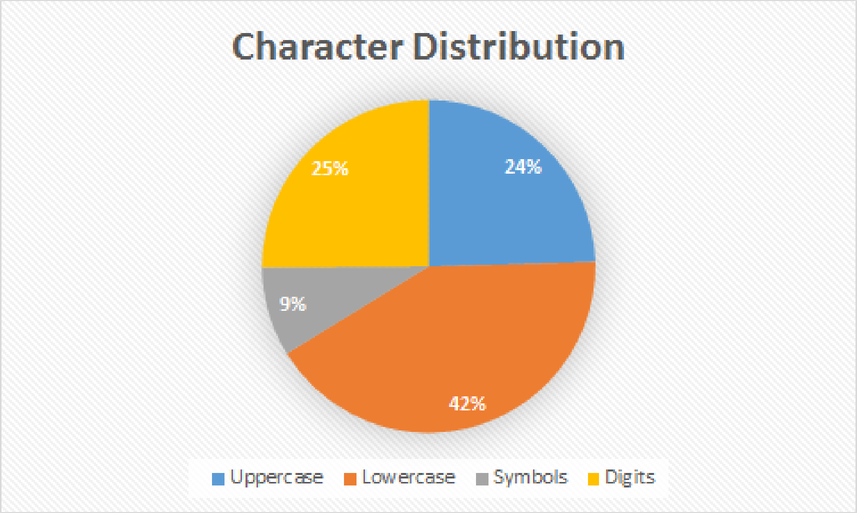
- LM Hashes: 98,6%
- NT Hashes: 98%
- MySQL Hashes: 83,8%
- htpasswd Hashes: 44%
- LDAP Hashes: 70%
- Wordpress SSHA-1 Hashes: 0%

Using a stronger machine[37], specialized FPGA[38] or Massive Cracking Array (MCA), it would have been entirely possible to crack 95% or more of all hashes within the same timeframe.

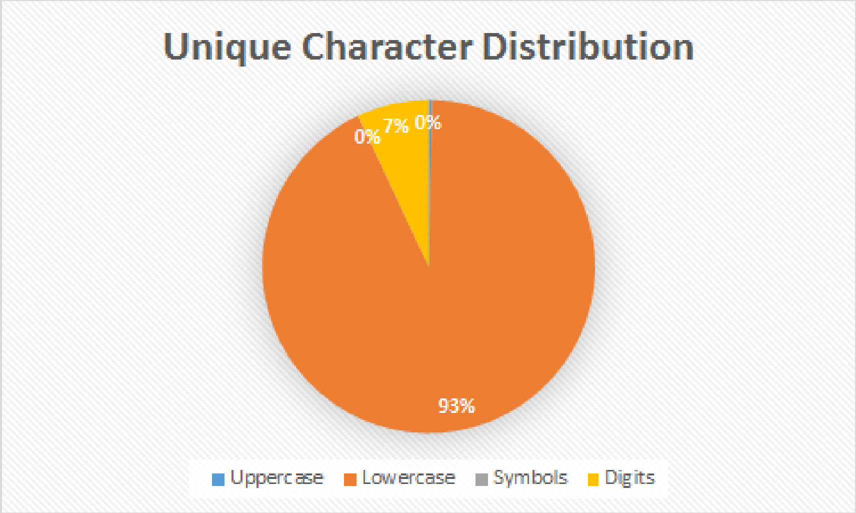
[32] http://tools.ietf.org/html/rfc2307
[33] http://en.wikipedia.org/wiki/SHA-1
[34] http://www.schneier.com/blog/archives/2005/02/sha1_broken.html
[35] http://dev.mysql.com/doc/refman/4.1/en/password-hashing.html
[36] http://www.openwall.com/phpass/
[37] http://ob-security.info/?p=546
[38] http://en.wikipedia.org/wiki/Field-programmable_gate_array

Analysis Results

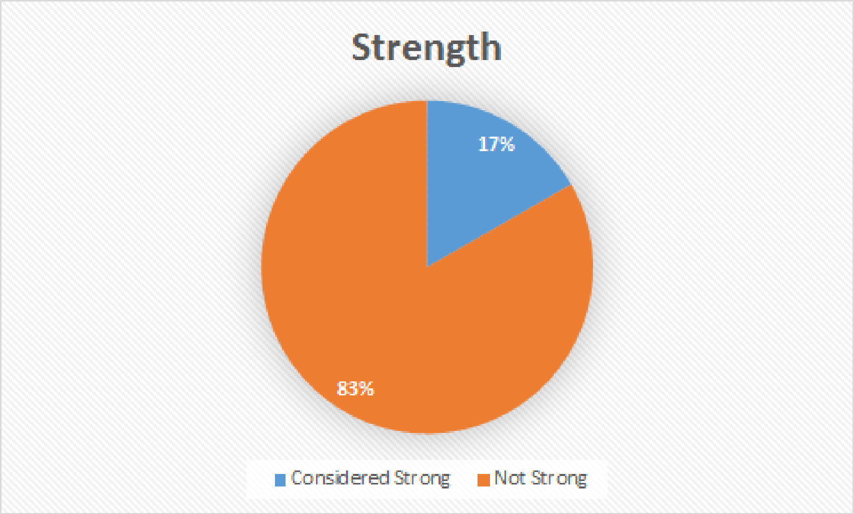
The character distribution shows what kind of character that is used in all the passwords. It is only an approximation since we were only able to crack 90.82% of the passwords; however, it is still the major representation of passwords and should be treated as such. A healthy character distribution contains equal parts of each character type as a password policy should enforce that each password should contain at least on of each kind.



The unique character distribution shows passwords that only contains the character type. As we can see, the majority (93%) of all passwords contains only lowercase characters while 7% contains digits only. Taking all the 8 character digit only passwords into an application that looks up the password as a Danish telephone number yielded that 15% of the passwords were the phone number of a student currently or recently studying at ITU. Similarly, 12% matched CPR numbers and 7% matched birthdates. A healthy unique character distribution chart would show that 0 passwords consist of one group of characters only.

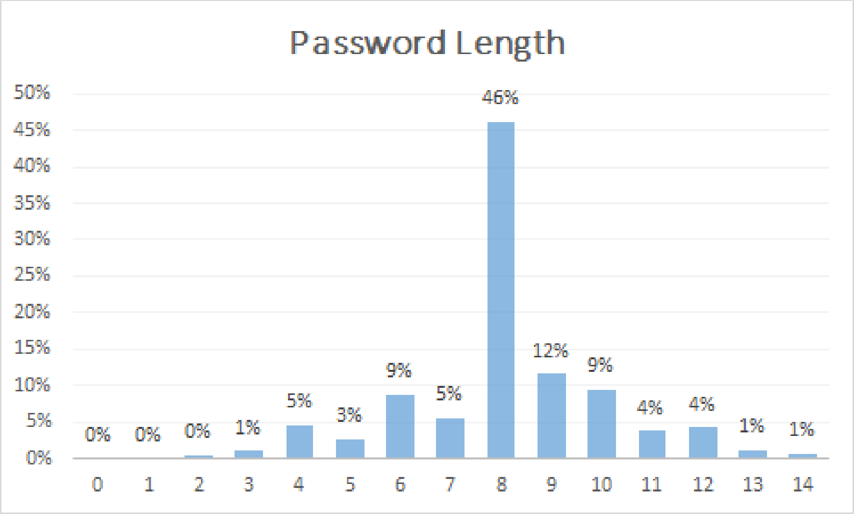


Passwords that are considered strong is 10 or more characters, contains upper and lower case, digits and at least one symbol. An example would be “cGuS43%t$bj5” that would take 1.74 centuries[39] to crack on a Massive Cracking Array.



To further illustrate the strength of the passwords, we took the most common word lists on the Internet and checked it against the passwords. 18% of the hashes were in the word lists. While this might seem like a low value, it comes from the fact that a lot of the password hashes we found comes from systems that auto generate the passwords.

The following graph shows the length (up to 14) and percentage of passwords with that length. As we can easily see, the dominant length is 8, which is normal, but also insecure. A healthy password policy would see 0 passwords below 10 characters.



[39] https://www.grc.com/haystack.htm

Password lengths

The longest and shortest passwords found are listed in the table below. Have in mind, that while the passwords are from 0 to 20 characters in length, they all took less than 10 seconds to crack using a wordlist. The passwords are low strength and follow typical weak patterns[40], making them easy to crack regardless of the length.

Password	Length
	0
1	1
x	1
p	1
nh	2
hj	2
jo	2
a3	2
db	2
bo	2

Password	Length
distributedcomputing	20
worldisnotenough	16
nielsnielsniels	15
alcatelsodavand	15
onkelscrooge183	15
Security_level1	15
Rollerblade2000	15
Krollebolle2000	15
LeedsUnited1919	15
Cecilie20062008	15

Recommended Solution

We have extracted several passwords from the database backups, file and web servers to do an analysis on the passwords. We also found thousands of password hashes in many different hashing formats and applications. Common to all of them is that they are weak in strength.

We would recommend that ITU deploys a password policy that spans the whole network. The policy should contain rules on what dictates a strong password, how passwords should be stored and recommendations on password sharing, reuse, and repeatable password patterns. This policy should be enforced throughout the entire network and passwords that do not comply with the policy should be disabled.

[40] http://hashcat.net/wiki/rule_based_attack

Data Sheet

Here we have created a table that contains numbers from the whole analysis. Have in mind that the numbers are based on observed data and not from an actual data analysis. We expect some of the numbers to have an error of about 15% as they are not designed to be an exact depiction of the infrastructure analysis, but rather an approximation of data we observed during the analysis. The confidentiality score is based purely on an estimation of how the data can be used to penetrate the network.

Name	Amount	Confidentiality Level (0-5)
Compromised Servers	157	-
Admin Passwords	7	5
Network Admin Passwords	27	5
DB Admin Passwords	20	4
Wordpress Passwords	28	1
User Passwords	14947	2
CPR numbers	37919 (11429 unique)	2
Emails	18088 (14038 unique)	1
GB of confidential data	144 GB*	5
GB of non-confidential data	3954 GB*	3

* Estimate based on the amounts of data found on compromised servers

Conclusion

We have successfully analyzed the network throughout the whole infrastructure. Multiple security vulnerabilities were found and some of them could result in a total network takeover by malicious hackers. We have made several recommendations that will increase the security where it is needed the most, but there are a lot more vulnerabilities around the network, that should be investigated.