IT infrastructure security report

Proposed changes to the main systems, processes and security architecture at Eklamot

Coco Jumbo Heurystyka International

Hubert Perliński
Michał Krzyżański
Michał Małecki
Tomasz Michalczewski

As-is analysis

Password security	Vulnerability management	Database security	Phishing and other human error prevention	Network architecture and security
Storage	Procedures	Storage	Training	Efficiency
Strength	Time efficiency	Access configuration	Prevention systems	Cost efficiency
Authentication	KPI	Backup		Stability and failover capability
Maintenance	Business continuity			Designed security methods
Creation				Actual security
				Expansion capabilities

Key:

Outstanding	Appropriate	Suboptimal	Inappropriate	Very inappropriate

Our team has thoroughly and painstakingly examined the provided information and was able to conclude that the IT system in the company is at best sub-optimal and at worst inappropriate and dangerous.

In order to improve the situation, we propose solutions that increase security, stability, efficiency and cost-efficiency in multiple aspects of the infrastructure and prepare it for many plausible scenarios.

Categorised change suggestions can be found below.

Password policy advisement

Upon reviewing the current policies, we have found **insufficient or redundant** security practices. Our recommendations for improvement are the following:

a) Password length and content

We do not consider twelve character cap as a viable policy - to ensure safety it is crucial to set the character cap at a pro forma level (for e.g. 50 characters). Moreover, to increase security and prevent high computing power brute force attacks, we advise changing the minimum number of characters in a password to 10. In this case, cracking a hypothetical password by a hacker group would take an estimated minimum of 50 years.

The use of some special characters simplifies some SQL injection attacks. To prevent that from happening, we advise disabling the following characters:
“& , | , % , = , < , > , / , \ , ; , ”” , +”
Below an example logic of checking if the password passes those requirements is depicted.

```
   if len(passwordFieldString)<10 or len(passwordFieldString)>50 :
    print("Registration failed")
    raise PasswordLengthInputError
  flag=0
  for special in ["~","!", "?", "@", "#" , "$",  "^", "&" ,"*" ,"_" ,"-","(", ")", "[" , "]" , "{ " ,
"}" , "'" , "." , ",", ":"]:
    if special in passwordFieldString:
      flag=flag+1
      break
  for uppercase in
['A','B','C','D','E','F','G','H','I','J','K','L','M','N','O','P','Q','R','S','T','U','V','W','X','Y','
Z']:
    if uppercase in passwordFieldString:
      flag=flag+1
      break
  for number in ["0","1","2","3","4","5","6","7","8","9"]:
    if number in passwordFieldString:
      flag=flag+1
      break
  if flag !=3:
    print("Registration failed")
    raise PasswordStrenghtError

  for symbol in ["&", "|", "%", "=","<",">", "/","\"","+"]:
    if symbol in passwordFieldString: ## this is just an additional measure, those characters should
be escaped in both login and password fields already
      print("Registration failed")
      raise PasswordCharacterInputError
```

Implementation of these rules in logging and registration processes can be found in “New Password System” folder.

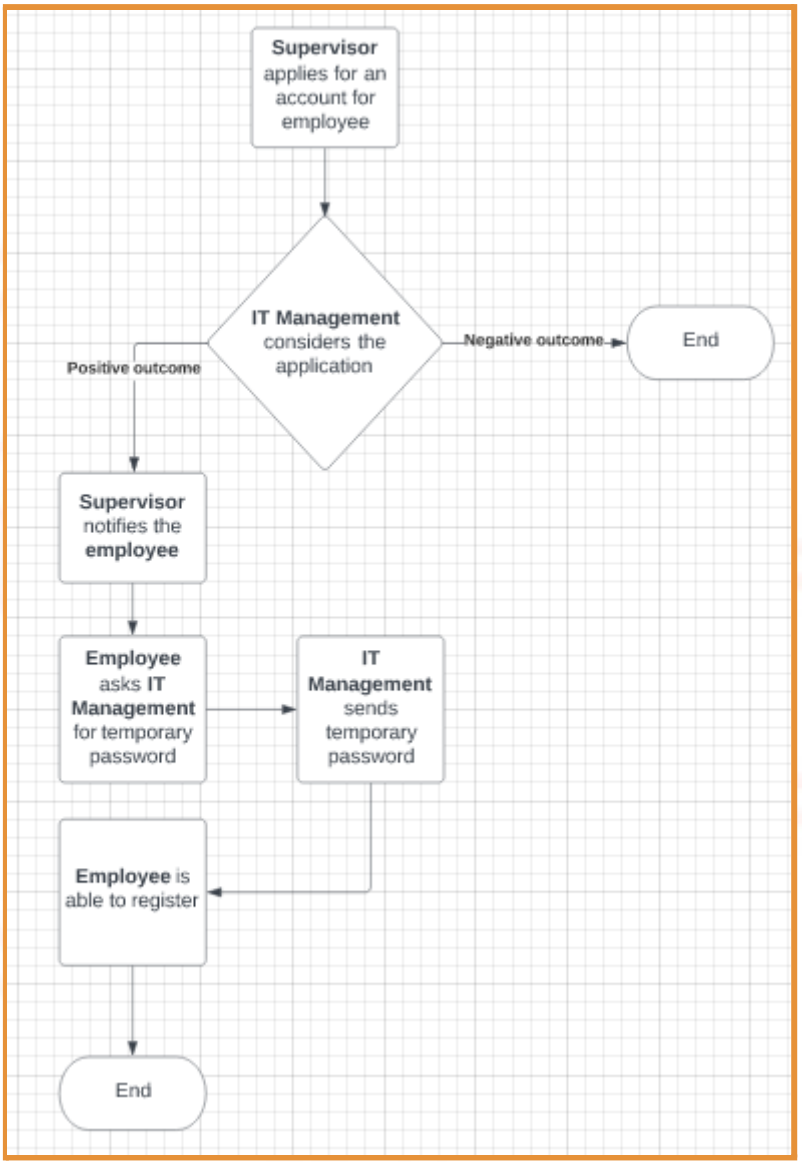
b) Periodic mandatory password changes

To minimise the risk of malicious persons taking advantage of various database leaks, we strongly advise introducing periodic mandatory password changes for all

employees. This should take place every 1-6 months (we leave frequency at your discretion).

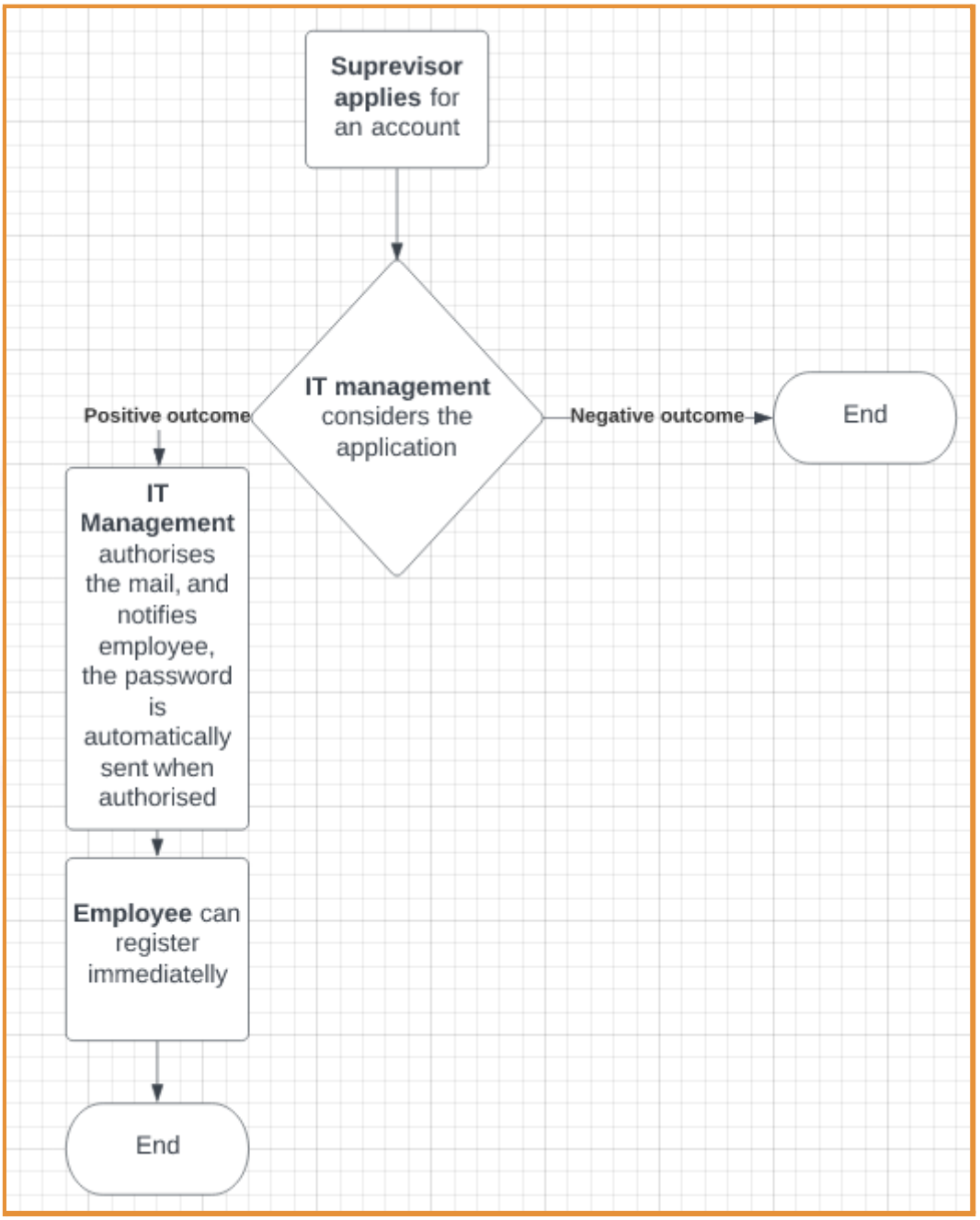
c) First login procedure

Upon reviewing the procedure, we have found it is suboptimal in terms of convenience and security. The current procedure can be described as such:



The following process has some innate flaws. It contains redundant communication between three parties: the employee, supervisor and IT Management. This creates more phishing opportunities and causes a higher risk of confidential information being leaked by insiders as happened in the last attack. Moreover, the process takes a significant amount of time.

To improve this procedure, we suggest introducing partial automatisation. The moment an account is created, a temporary password should be generated and sent automatically to the employee. This is more secure as it prevents some phishing emails posing as IT support, decreases procedure time and reduces the amount of everyday work IT personnel have to perform. This improved process is depicted below.



d) Two-Factor authentication

Implementing two-factor authentication is crucial for damage control after passwords are leaked. When implemented, intercepting a password without a 2FA token is meaningless. Only serious phishing attacks would be able to bypass both factors - should be implemented together with CJHI anti-phishing system to ensure maximum security. This topic is discussed more in the provided report on new anti-phishing system propositions.

e) Reset all passwords after the policy update

We recommend immediately conducting a company-wide password reset. A deep review of the provided database unveiled that almost none of the passwords are policy-compliant - moreover, some are on the list of most popular passwords in the world, which means that they can be cracked at any second (examples include “123” or “abcd1234”).

We have prepared a new password policy document, implementing those propositions. It can be found in the attached file “Nowa polityka haseł”.

Database architecture guidelines

Conducting analysis of database structure revealed a wide spectrum of weak spots and inadequate data storage practices.

a) Passwords storage

We are able to secure our database from SQL injection attacks by implementing appropriate password policy changes, but at the current state, the way we are keeping passwords is putting us at risk of leakage. We recommend hashing passwords before storing them in databases as a simple and effective method for damage control after potential attacks. Below is the code to the hashing algorithm.

```
def Hash(password):
    salt = os.urandom(32)
    hashed_password = hashlib.pbkdf2_hmac('sha256', password.encode(), salt, 692137)
    hex_hash = hashed_password.hex()
    return [hex_hash,salt]
```

The algorithm adds salt as an additional security measure. The salt is stored on the website, while the password is in the SQL database. Both are needed to try to brute force attack, so a database leak wouldn't be enough to effectively perform the attack. A full example of hashing algorithm and implementation can be found in “New Password System” folder.

We additionally advise using TLS/SSL certificates while sending any packets to the database, especially those containing confidential data.

b) Inactive accounts storage

Eklamot stores data of inactive users which is endangering the company with possible fines while not serving any particular business role. Inactive users should not be kept in databases and deleted immediately.

c) SQL configuration suggestions

This is not a security issue, but rather a convenience issue. The current timezone setting is UTC-8, which is inappropriate. Change timezone to Warsaw Time.

Vulnerability Management processes

Processes targeted at protecting systems from vulnerabilities were found to be not appropriate, and not conducted periodically. To repair those issues implement:

a) Scanning as part of a recurring process

We highly suggest making regular Vulnerability scans a responsibility of the security team. The current process requires SysAdmin to request a scan, which in our opinion is redundant and can lead to ignoring vulnerabilities. SysAdmin should be able to request a scan, but only when they think there is a special need.

b) Assessment of likelihood and impact

To manage vulnerabilities accordingly, it is crucial to assess the likelihood and impact of a threat early in a process - we suggest that the Security team should perform this estimation after identifying vulnerabilities and prioritise in workflow for SysAdmins the riskiest.

c) Including risk in KPIs

Although your KPIs are informative and should be kept; we suggest expanding the list by KPIs that include risk in the vulnerability process such as: “number of critical vulnerabilities per process/year”, “per cent of critical issues handled outright after detection” or “number of vulnerabilities with high impact that were derogated” or “% of detected vulnerabilities that were found critical”.
Such expansion will help assess capabilities for handling vulnerabilities that matter the most.

An exemplary proposed KPI set is as follows:

Lp.	KPI	Description
1	# of detected vulnerabilities	Number of vulnerabilities found by scanner
2	# of requests for derogation submitted	Number of requests submitted by SysAdmins
3	# of false-positive	Number of vulnerabilities flagged mistakenly
4	Average times needed to close a vulnerability in a given month	Average time needed to resolve a vulnerability (from scanning to prepared end-report)
5	# of detected critical vulnerabilities	Number of vulnerabilities found highly impactful (by safety Team)
6	% of detected vulnerabilities that were found critical	Share of critical vulnerabilities in total # of detected vulnerabilities

We prepared a new file proposition outlining the Vulnerability Management procedures for your review. It is attached to the report under the name “Vulnerability management procedures”.

Security issues in the user permissions

The file “sudoers.txt” assigns each user certain sets of permissions.

```
root    ALL=(ALL:ALL) ALL
%admin  ALL=(ALL) ALL
%sudo   ALL=(ALL:ALL) ALL
%support   ALL=(ALL:ALL)  /bin/bash, /usr/sbin/reboot, /usr/sbin/shutdown
%usermgmt  ALL=(ALL:ALL)  /usr/sbin/useradd, /usr/sbin/usermod, /usr/sbin/userdel
%grpmgmt   ALL=(ALL:ALL)  /usr/sbin/groupadd, /usr/sbin/groupdel, /usr/sbin/groupmem, /usr/sbin/groupmod
test     ALL=(ALL:ALL)    /tmp/scripts/test.sh
```

There was one security risk found. The user “test” has full access to” test.sh”. This user is probably a remnant of testing, which can become dangerous. If someone gets access to “test” they can easily access and rewrite “test.sh”.

The test.sh file may be rewritten to do any of the following malicious activities:

a) Disable firewalls
b) Disable Linux Security Modules
c) Modify access control lists
d) Change file attributes

Additionally as the file is in tmp, it can execute and disappear after the restart of the system. If an attack is performed through this file it might be difficult to establish the contents.

We advise removing the “test” user together with “test.sh” immediately.

Security issues regarding network connections

The following is the provided scan of all connections to a network.

```
PORT      STATE    SERVICE    VERSION
21/tcp    open     ftp        vsftpd 2.3.4
80/tcp    open     http       Apache httpd 2.4.46 ((Debian))
111/tcp   open     rpcbind    2-4 (RPC #100000)
443/tcp   open     ssl/ssl    Apache httpd (SSL-only mode)
3306/tcp  open     mysql      MySQL 5.5.5-10.1.26-MariaDB-1
4767/tcp  open     unknown
4769/tcp  filtered unknown
5037/tcp  open     unknown
39381/tcp open     unknown
39563/tcp open     webdav
53013/tcp open     soap       gSOAP 2.8
53014/tcp open     ssl/soap   gSOAP 2.8
53113/tcp open     soap       gSOAP 2.8
53114/tcp open     ssl/soap   gSOAP 2.8
```

The scan reveals many security issues resulting from unidentified connections and outdated software. Descriptions and solutions:

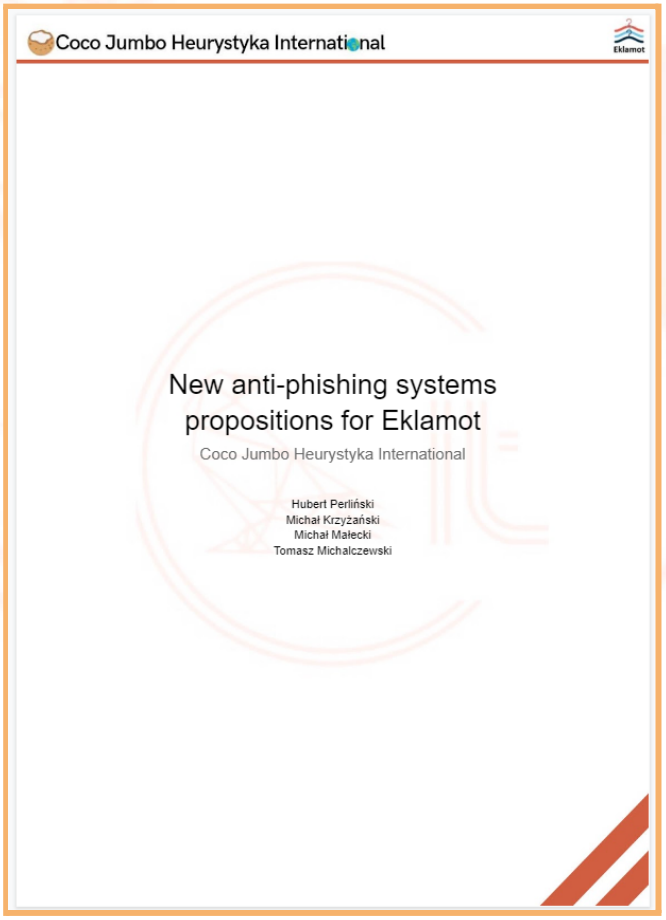
a) Immediately update vsftpd to the current version (3.0.4) as version 2.3.4 is known for its backdoor exploitability.
b) Update Apache HTTPd to the latest version (2.4.53) to maximise security and efficiency
c) 2-4 RCP #100000 is only needed for NFS file sharing. If that's the case it should be left as it is, otherwise deleted as it is just an unneeded security risk.
d) MySQL's connection should be secured with SSL, otherwise, the data that is transferred from and to the database could be intercepted
e) 4 unknown services should be disconnected from the network immediately. It should be investigated whether they are unidentified parts of IT infrastructure or an outsider connected to the network, which may be the remnant of a recent attack. Especially close attention should be brought to the one filtered connection, as filtered might mean that the probes were intercepted on purpose to hide activity.
f) Moreover port 5037 is used specifically by the android debug bridge (ADB) server - a command-line tool which facilitates actions such as installing or debugging apps, which can be used with malicious intent. Its inherent part is a daemon, which runs as a background process and hence is hard to detect. The port should be closed if not needed.
g) Soap is significantly exploitable in version 2.8.107, make sure to update it, preferably to the latest version (2.8.121) to maximise security and efficiency

Human error prevention

Human errors are inherently difficult to foreshadow and prevent. We see that the biggest risk of human errors in Eklamot is caused by phishing attempts. This is why we have created a separate **report on phishing prevention**, which can be found in the same folder.

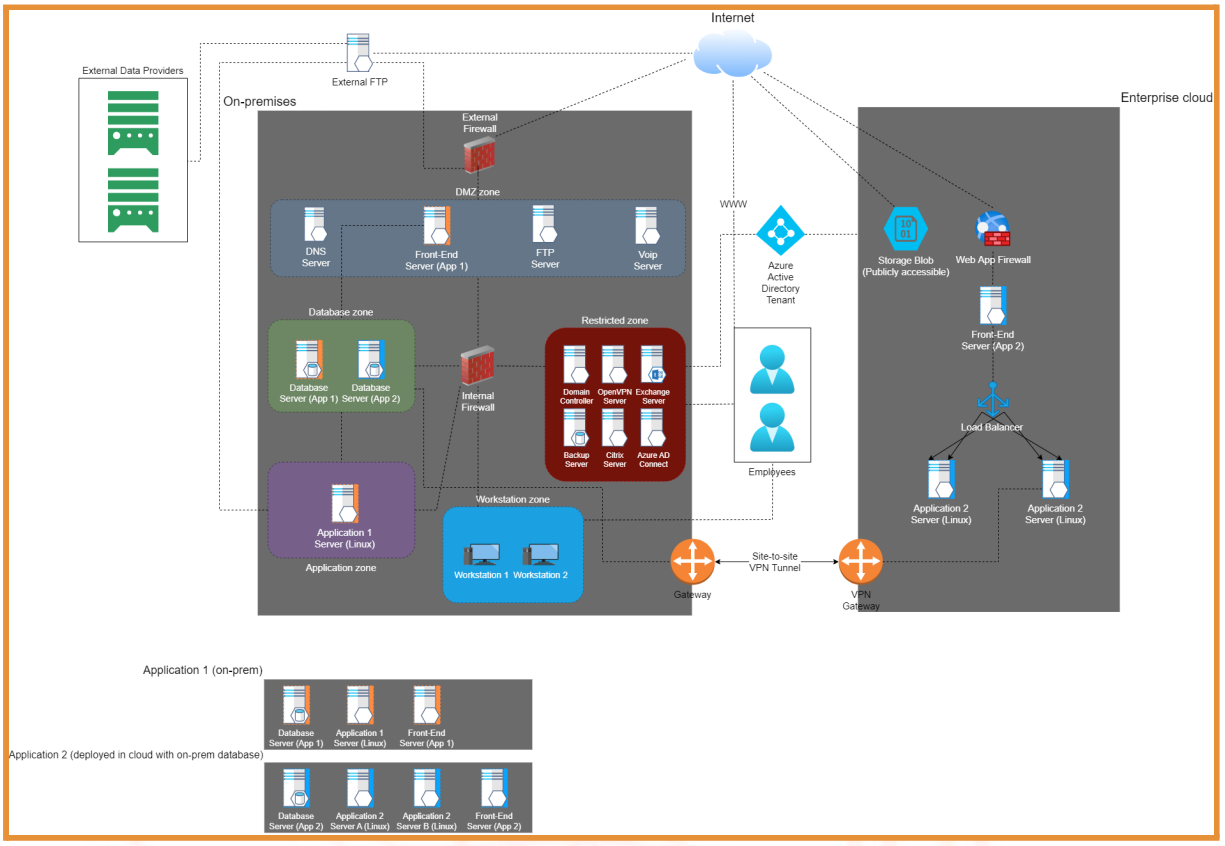
However, there are some more potential human-induced vulnerabilities worth mentioning.

a) Leaving testing code in files:
   It can lead to unauthorised access to the system. The only remedy for this is a strong work ethic, responsible office culture and double-checking.
b) Forgetting to log off: Again, an unauthorised person can gain access to the system, which may result in loss or theft of data. Can be prevented by automatic log-off systems that detect inactivity as well as a strong work ethic and responsible office culture.
c) Accidental deletion: Can happen to anybody. Can be dealt with using a backup storage system.



Network Architecture change suggestions

Our team was provided with the following diagram representing Eklamot's network:

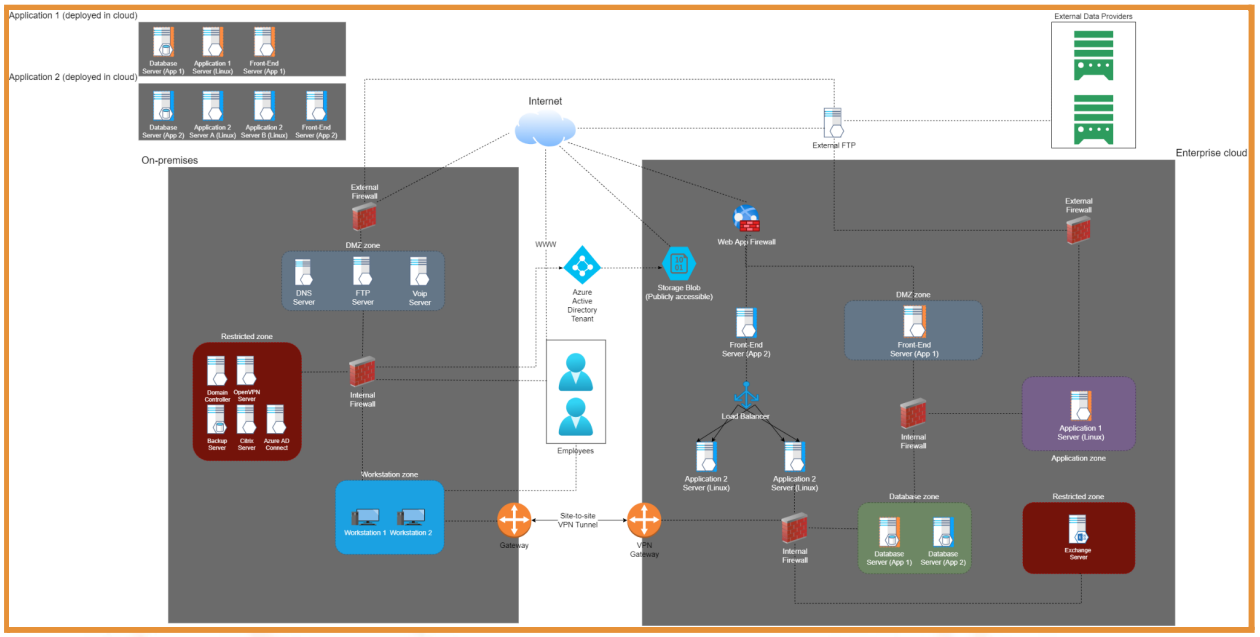


Although so far the system operated without major issues, we identified several significant **security and service continuity threats**, which can be **mitigated** by implementing **alternative network architectures**. Some suggestions, which will be described in this part of the report, can be implemented on their own and some relate to a broader set of overhauling actions. Some also require more complex operations, however, we exhort considering all of the propositions to future-proof the system and introduce improvements before they are necessitated by security breaches or failures.

Based on the change suggestions we prepared two diagram propositions for the final architecture of the network. The first one is focused on **simplifying the on-premise** network by moving the most important elements to **the cloud**, however, still relies on the in-house infrastructure for the operations. The second proposition builds on the first one and requires **introducing new servers** to the network, which may take a longer time to implement, but results in a network which can operate even when the on-premise facilities go fully offline, ensuring the **continuity of service**. The suggestions are grouped by those two scenarios, as implementing all of them ultimately leads to the networks on the new diagrams.

Scenario 1 - moving the main infrastructure to the cloud

The implementation of suggestions contained in this part should result in a network architecture similar to the one presented in the diagram below.



1. Migrating the databases, front-end and application servers to the enterprise cloud

In the current state of affairs, the databases and Application 1 infrastructure are located on-premises. In terms of security, this setup is **highly undesired**. The main reasons for this are:

- Lack of experienced on-call staff at Eklamot to manage the servers and immediately respond to threats,
- Lack of advanced automated security features, which are being constantly updated,
- High complexity and maintenance of the on-premise network,
- The danger of hardware failures with no simple failovers,
- Inferior security analysis functionalities,
- Need for more advanced offline security systems,
- A big risk of breaking the business continuity during server downtimes.

Addressing those issues on-site would come with a very significant investment of time and money, which makes little to no sense for a company like Eklamot and would result in very high threat levels during the creation and implementation of new systems. Migrating those servers to an **enterprise cloud** (like the already in use Microsoft Azure) solves the listed issues quickly and reliably. The **benefits** of such a solution include:

+ 24/7 on-location staff, highly experienced in data centre management and security,
+ Advanced automated security solutions developed by the largest and most experienced corporations, improving the measures on daily basis,

+ A less complex on-premise network, which will be easier to manage and secure,
+ No threat of server failure with no failover system,
+ Quick and easy implementation of backup servers,
+ Superior analytics tools for security management,
+ Outsourcing significant parts of offline security,
+ Reduction in risk of breaking the business operations continuity,
+ Proven solutions used by numerous companies in the sector (e.g. Zalando)
+ Little initial investment,
+ Predictable and easily scalable cost of running the servers.

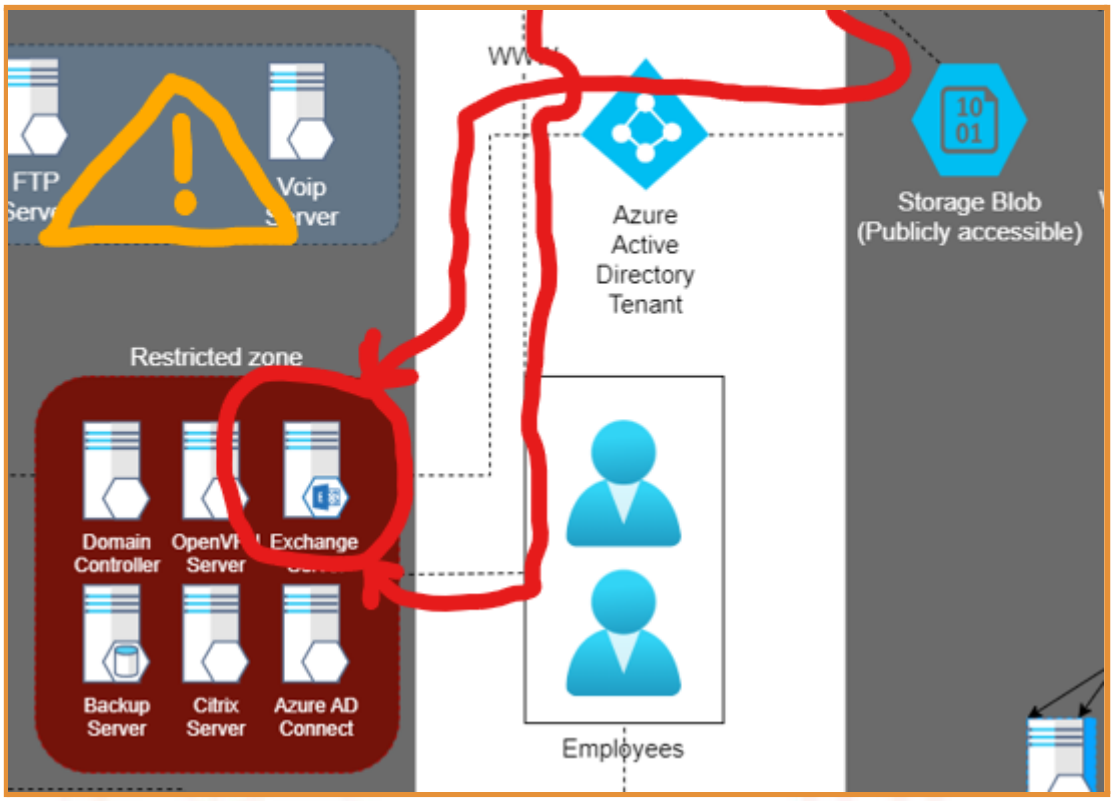
Based on those factors it is easy to see why a cloud-based server solution is suitable for Eklamot, however, it is also important to consider what would be the **potential reasons not to move the servers**.
The biggest advantage of the on-premise option over the cloud one would be the ability to create a server configuration which would be very specifically tailored to the needs of Eklamot. In the current circumstances, this option is **not necessary**, as the company does not require any uncommon services.
Another gain would be in possible savings if in the long term Eklamot will require very significant computing power. Although this is a consideration for the future, the current scale of the company's operations **does not justify such capabilities**.
If the situation changes in the future, on-premise systems could be developed and tested while the regular operations continue in the cloud, guaranteeing **proper change management practices.**

Due to those reasons, migrating the servers in question to the cloud is declared an **essential security change**, to be implemented as fast as possible.

Implementation:
To achieve this goal, a **parallel server deployment** is recommended to ensure business as usual for the other departments. First, replicate the server structure in the enterprise cloud solution, then test the system and move some of the traffic to it. If after a trial period of a few days no problems arise, take the old servers offline and hence complete the transition.

2. Migrating to a cloud Microsoft Exchange solution

Currently the mail and calendaring in the company are handled by an on-premise Microsoft Exchange solution of **questionable security**. It does not seem to use any spam and phishing filtering systems, which is elaborated upon in the separate report on anti-phishing systems. Moreover, right now there exists a direct and unfiltered line of access from the level of an employee and the Storage Blob to the restricted area containing the server, which can pose a **serious security vulnerability** when noticed by an actor with malicious intent.

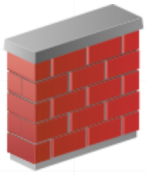


To address this issue a firewall could be introduced to filter the user requests, however, due to the high threat coming from the reported phishing attempts, we recommend a more elaborate solution - migrating from an on-premise server to the Microsoft Exchange Online services (or Microsoft 365). The **benefits** of such a move would be:

+ Access to cheap and reliable anti-spam and anti-phishing services developed by Microsoft,
+ Very high flexibility at a comparable price point,
+ Availability of service - even during an on-premise failure the employees would have access to company emails,
+ Server failover availability and complete procedures of disaster recovery,
+ Easy migration process from both a user and technical standpoint,
+ Less complex on-premise network to manage.

An **argument against** such a move would be the reduced administrative control over the infrastructure that comes with an off-site system. However, Eklamot **does not require** any extraordinary solutions in that regard and the built-in tools should be more than enough for this use case. If in the future such a need arises, a new more secure on-premise solution can be developed without significant issues.

Implementation:
For this transition, a **direct changeover** is suggested. After the cloud solution is prepared and tested, which should not take long, all employees should be informed of a system change and complete the first logins on the same day. In that way everybody will work in the same system at all times and the employee assistance in the switch can be focused on one day, preventing anyone from operating on the **legacy solution**.



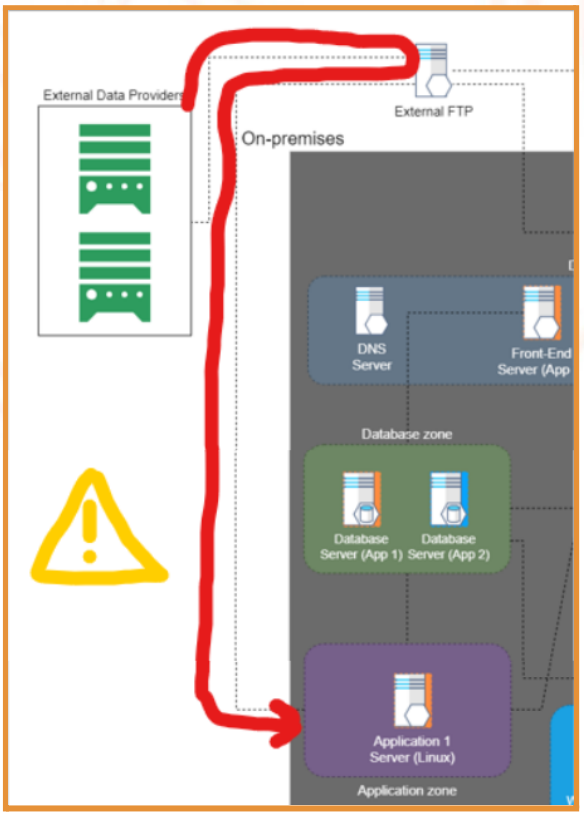
3. Introduction of additional firewalls

The contemporary architecture utilizes 3 separate firewalls, which **don't protect** some of the **most crucial elements** of the network, including the Application 1 server and the servers in the restriction zone. The latter is assumed to be an imperfection resulting from a rushed implementation of work-from-home systems during the pandemic. Also, new packet filtering needs arise from the proposed migration to cloud servers.

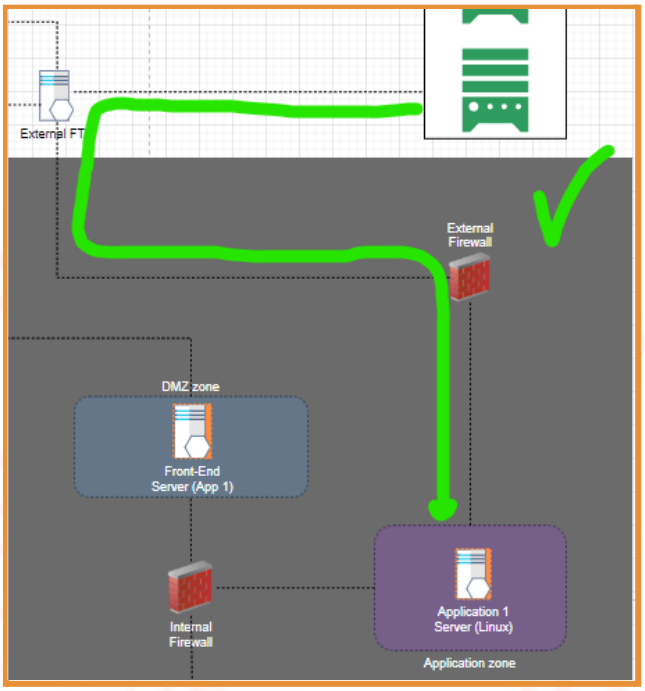
To address those vulnerabilities, at least 3 new additional firewalls are proposed and changes to the on-premises internal firewall are outlined:

1. **Filtering the connection between the External FTP server and the Application 1 server.**

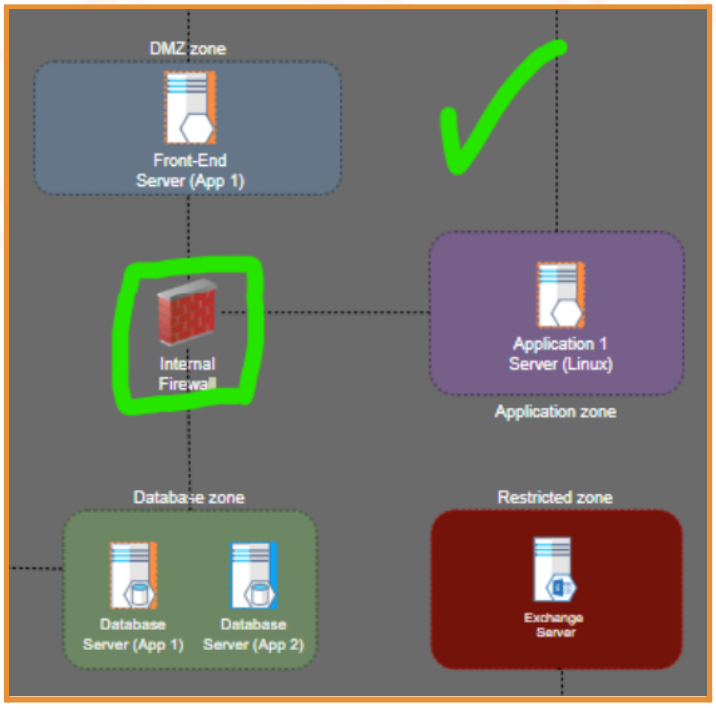
   The files on the External FTP server come from data providers which are beyond the control of Eklamot and have unfiltered access to the Application 1 server.



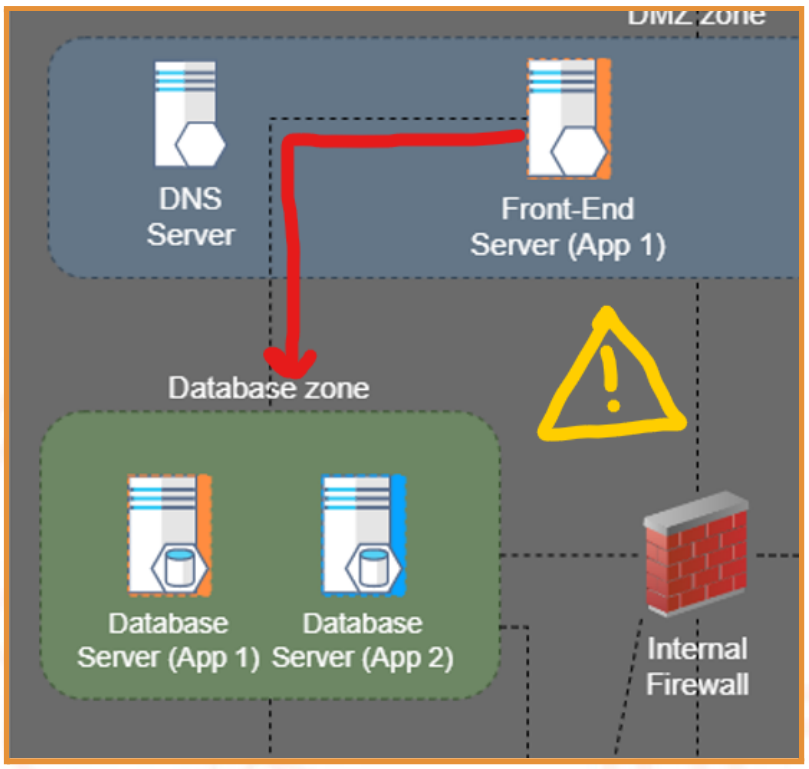
Since we have no information about their security practices, the files received from the server should **not be trusted** and under no circumstances should be allowed to reach the Application 1 server without a previous security scan. The proposed fix is outlined below:



2. **Controlling the traffic of the new Application 1 infrastructure:**

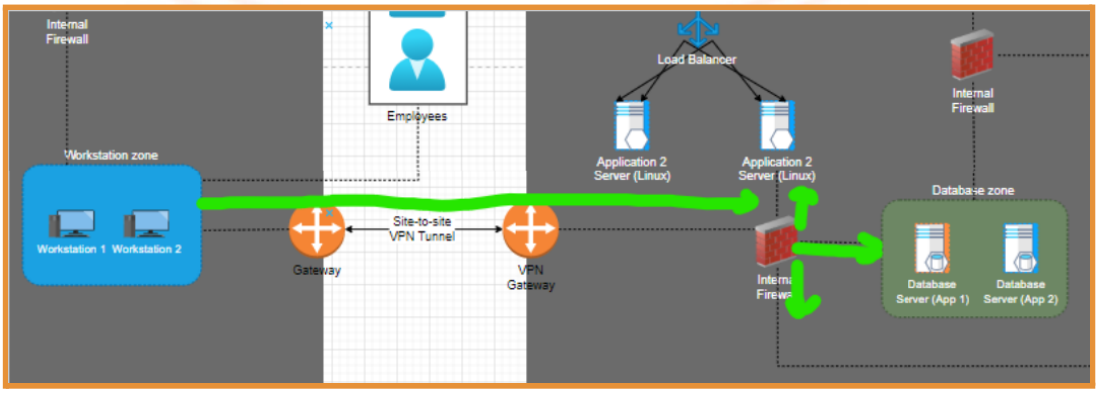


This firewall is necessitated by the move of Application 1 to the enterprise cloud. Moreover, in the current set-up, the Front-end of Application 1 has a **poorly secured** line of access to the Database zone, which should not be allowed.



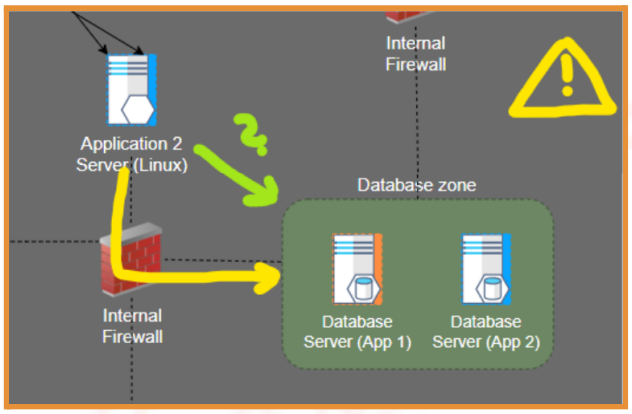
It is possible that a skilled attacker could **abuse the connection** (marked by the red arrow) and attack or gain access to the database (e.g. an SQL injection). This significant vulnerability is removed in the proposed cloud deployment of Application 1. To ensure a smooth user experience in spite of the added firewall, resource-efficient commercial firewall software can be used, guaranteeing fast scans.

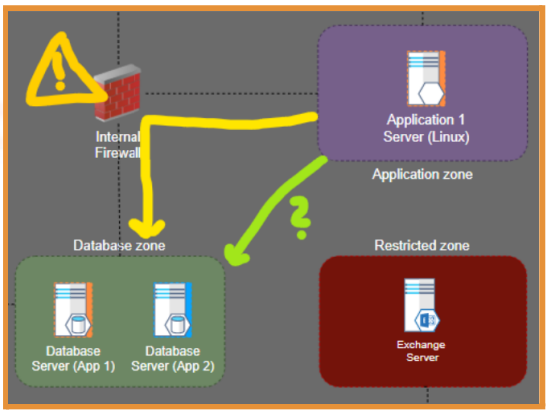
3. **Monitoring the new connection between the workstations and the cloud system.**



The migrated databases and Exchange server need to be secured from potential insider interference, both intentional and accidental. This is **standard practice protection**.

It could be argued that routing the Application server traffic to the databases through a firewall is **sub-optimal** and this infrastructure should be connected with a direct line.

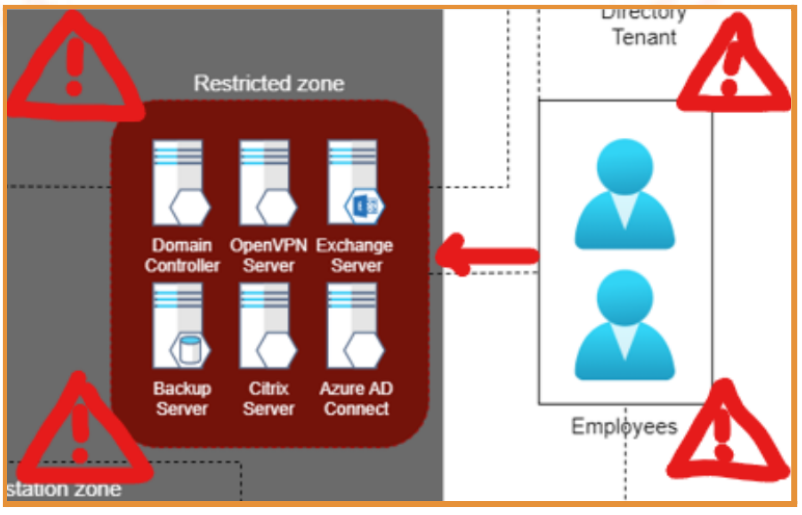




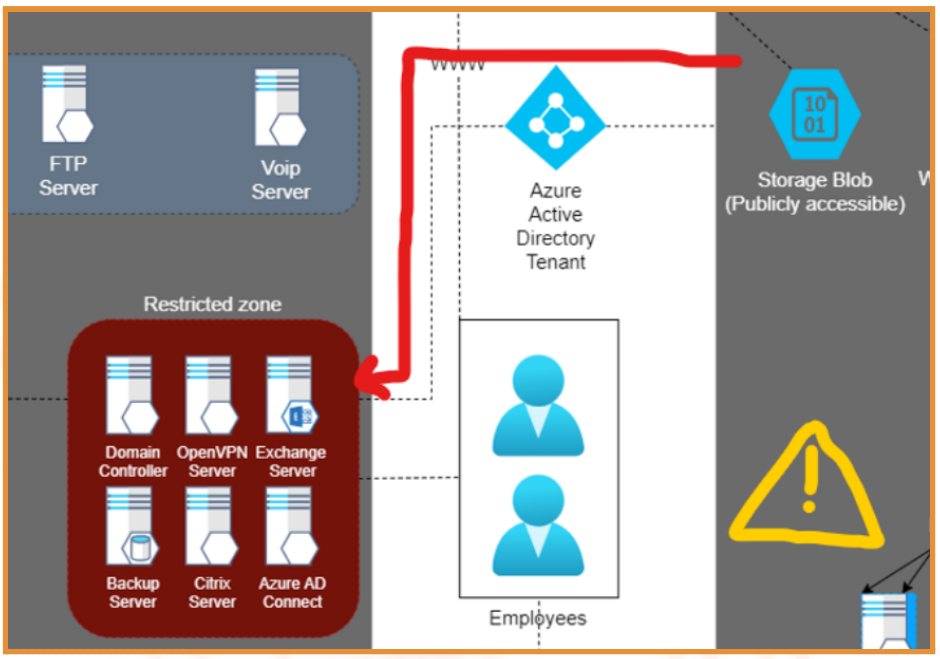
However, remembering the recent cyberattacks at Eklamot we **strongly recommend** utilizing this **extra layer of security**. If during the first weeks of working on the new architecture the firewall does not detect any issues arising on this line of access, the filtering settings can be turned down to allow a faster and smoother connection.

4. **Controlling the connection between the employees, storage blob and the restricted zone.**

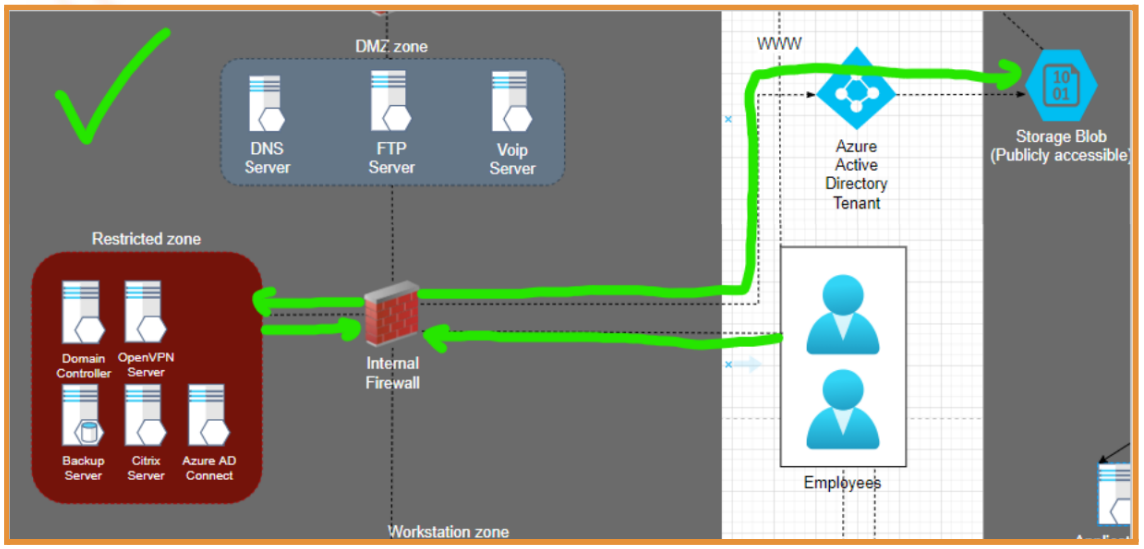
In the system in operation, the employees have unmonitored access to the Restricted zone, containing Eklamot's most crucial infrastructure, including the Backup Server and the Domain Controller. This is a **critical threat to system safety** and should be addressed **immediately.**



This state of affairs makes reaching the pivotal servers for a potential attacker significantly simpler and the breach could remain **unnoticed for years**. The pandemic forced such unsafe solutions on many companies, however this late into the "home-office era" this solution is **unacceptable**.
Another vulnerability in the same system area concerns the publicly accessible Storage Blob. Although the AADT provides some level of security in terms of user authentication, the traffic between the Blob is also **not monitored**, which makes the active threat identification unnecessarily more challenging.

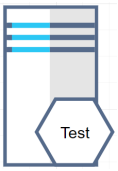


A simple solution to those important problems is an introduction of a firewall between the nodes in question and the Restricted Zone.



With this, the actions the employees can undertake are better controlled and monitored, while retaining the ability to work from home. The files transferred from the Storage Blob are also better registered, making the active threat detection process more streamlined.

Implementation:
The firewalls described in points 1 and 4 should be implemented **immediately** and be given **top priority.** The other two are to be configured together during the cloud service set-up process. To make the process faster, we recommend using the same firewall software as is used for the already established ones in the architecture. A separate **security audit** of the servers from the Restriction zone is recommended, to ensure that there are no dormant threats already in the network.



4. Repurposing old hardware as test servers

As a result of the server migration to the cloud, there will be some on-premise hardware no longer used for the main operations. This, however, does not mean that it has to be put in storage or sold to the second-hand market, many alternative uses for it can be identified. We believe that the most beneficial use for Eklamot would be turning the hardware into a cheap local test environment.

This would greatly enhance Eklamot's **Change Management** capabilities, as the change requests could be tested in an environment separate from the production environment. The testing data could also be easily anonymised. The final result would be increasing the **continuity of service**, by decreasing the chances of system-breaking errors appearing in the production architecture.

Currently, to the best of our knowledge, Eklamot does not operate any test systems and the benefits of doing so are very far-reaching. There are, however also other, more elaborate possible uses for the hardware the company already has, which are not necessarily preclusive with the proposed testing environment. They are described in more detail in the second scenario of architecture overhaul.

Implementation:
This is a secondary project proposition and should be implemented when the other security concerns are handled.

5. Implementation of good maintenance and system recovery practices

Although the network architecture pictured in the Scenario 1 diagram can be described as **vastly superior** to the current solution, it still has some **compromises**, which were left in the system to ensure the **implementability** of the proposed solution. Scenario 2 tackles those trade-offs, however, requires a bigger resource investment to configure. The issues that could theoretically arise in case the on-premise systems go partially or fully offline include:

- Inaccessibility of Eklamot's subdomains,
- Inaccessibility of Citrix dependent applications,

- System log-in issues,
- VPN connection issues.

If those risks are to be accepted by Eklamot, there are some good maintenance and system recovery practices, which will help **prevent downtime** and reduce the **Recovery Time Objective** (RTO) and **Recovery Point Objective** (RPO).

Implementation:
Since Eklamot is an internet shop, any downtime is a potential loss in revenue, thus the RTO should be no longer than a few hours. To reach this goal we suggest delegating the task of preparing and updating system recovery procedures to an experienced employee of the IT department. to be available 24/7 for and trained in system recovery.

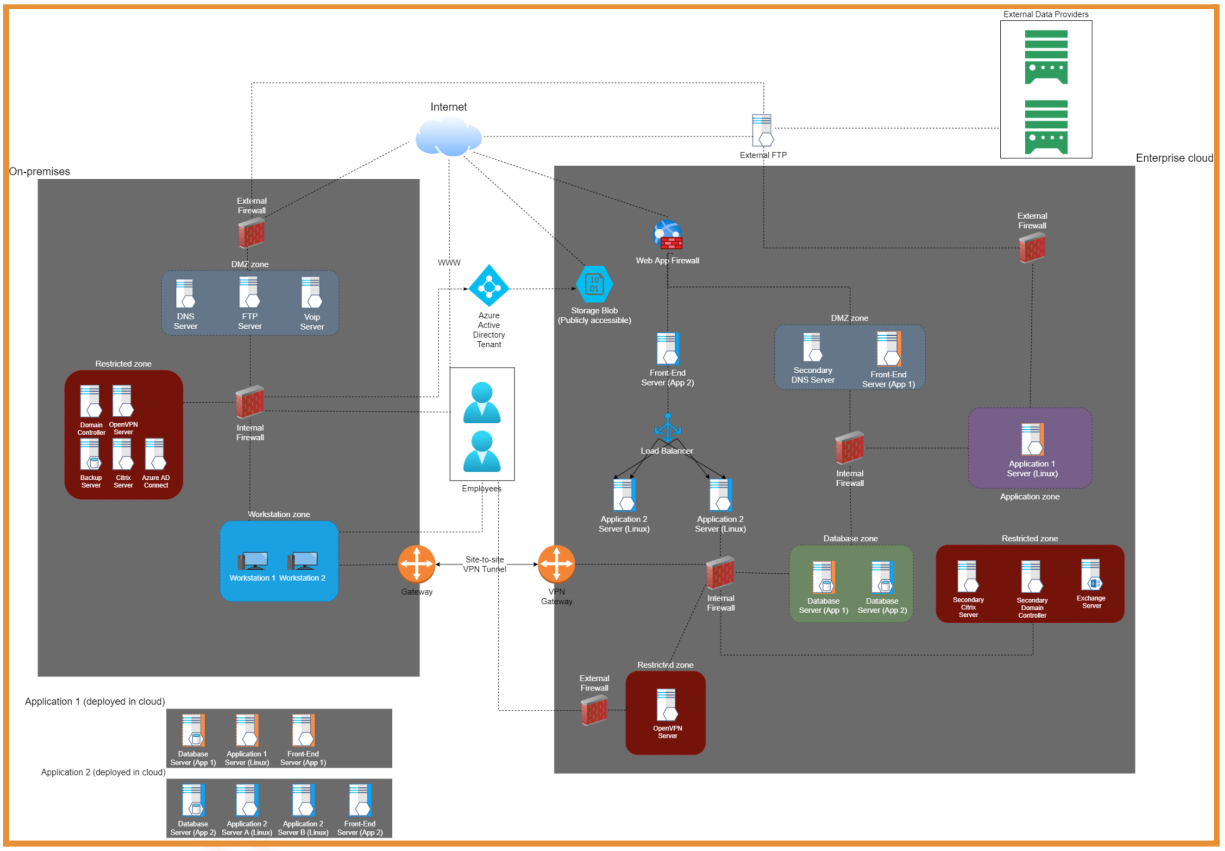
Eklamot should aim at a few hours long RPO, although operations other than ordering can continue during downtime. Backups should be done regularly and stored on the on-premises backup server.

In case of a power shortage, it is important to have an emergency power supply connected, so that temporarily stored data is not lost and operations can be continued for a period of time.

Other specific procedures should be identified and regularly implemented by the designated employee.

Scenario 2 - introducing independence from on-premise solutions

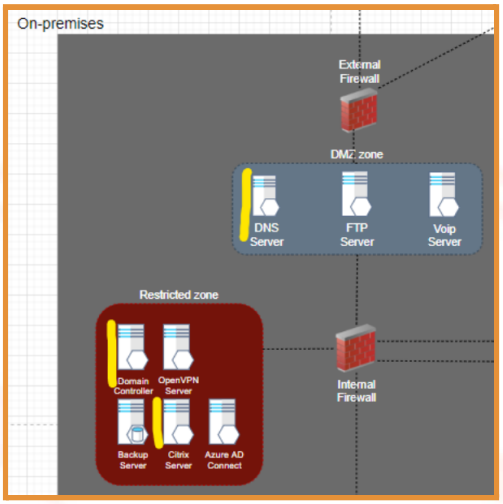
As written in point 5 of Scenario 1 change suggestions, the previously described system includes some **compromises** to increase the implementability of the suggestions and reduce the resource investment. However, if Eklamot wants to achieve a truly **future-proof** solution and **full systems redundancy**, we recommend extending the changes made to include the ones described in this sub-chapter. Provided all the suggestions are accepted, the final network architecture could be represented by the following diagram:



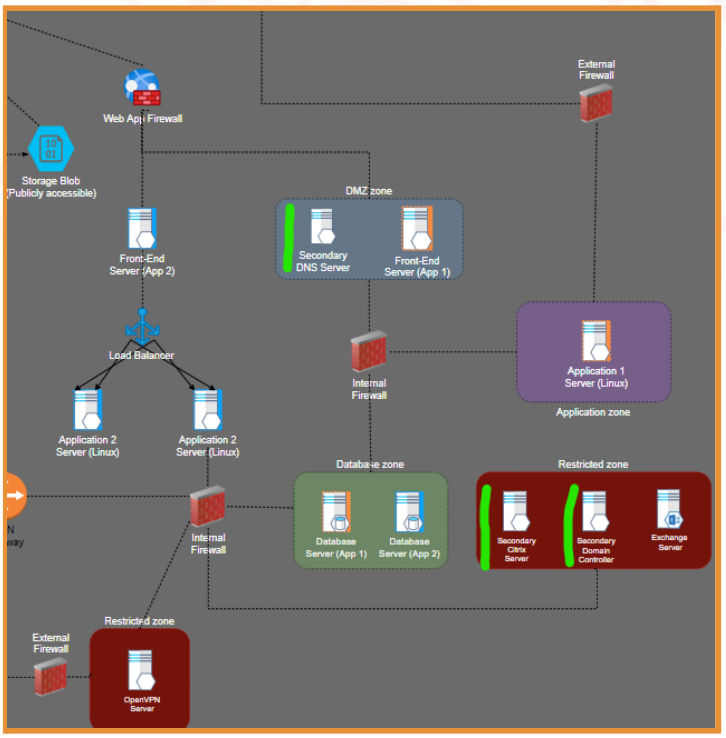
2

1. Creating secondary Citrix and DNS servers and the secondary Domain Controller

In Scenario 1, the Citrix and DNS servers, together with the Domain Controller are located on-premise.



If extra resources are available, this cost-saving solution can be changed to one providing full redundancy, which comes with important benefits. It would be creating new secondary servers in the enterprise cloud services, which were marked with a green line in the snippet of the architecture diagram below.



The advantages of using this set-op rather than the fully on-premises one are:

+ Ability to flexibly increase or decrease the operational capacity of the store in preparation for usage spike events (sales, new product launches, etc.)
+ Systems redundancy - if the on-premise operations go fully or partially offline, the continuity of business operations is ensured. The secondary servers take on the role of the primary servers and keep the store operations online. The capacity can be initially limited, however, the continuity prevails.
+ Maintenance of the on-premise servers is easier to plan and complete, as the secondary servers can temporarily take over.

The drawbacks of this solution come from the required initial configuration time and increased continuous spending on cloud services. The capabilities offered are however worth the effort.

Why not completely move those systems to the cloud?
It may be tempting to fully get rid of the on-premise solutions and rely fully on the cloud services, as failover capabilities and flexibility would still be maintained. We decided not to recommend this solution due to three major reasons.

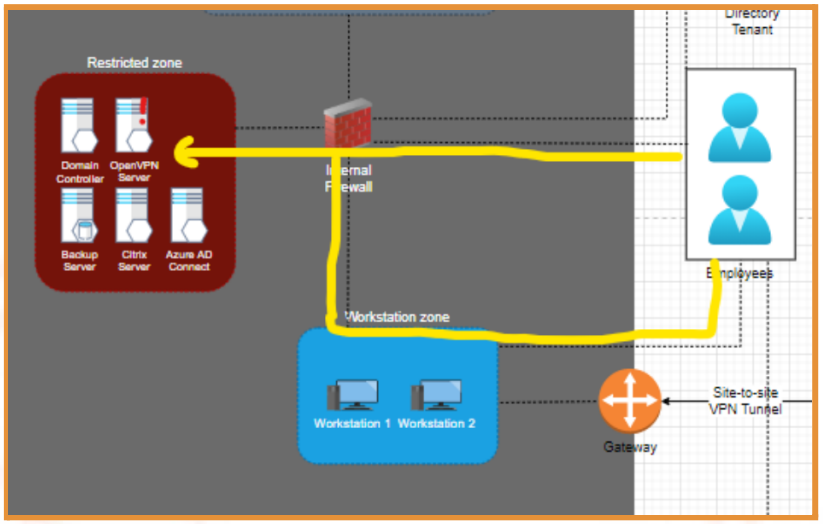
1. The on-premise infrastructure is already created, making its costs lower than relying fully on cloud solutions.
2. This setup allows for greater control and cross-company redundancy.
3. Since the on-premise network is to become much simpler, proper security of those servers is easier to maintain.

Why isn’t the same reasoning applied to the main application servers and the databases?
The application servers are more advanced infrastructure and are much more costly in maintenance than e.g. the Citrix server. Their security is also highly dependent on the Domain Controller. Keeping them in the cloud is an anti wasted capital decision, as upgrading the on-premise systems would come with huge unnecessary spending.
In the highly unlikely scenario of the full cloud provider system failure, the application infrastructure can be restored with relative ease using the data backups.

Implementation:
This system upgrade does not cause any impact on the ongoing operations, so the servers can be created in the background of the regular work. As this is an extra security feature no particular order of server creation is recommended.

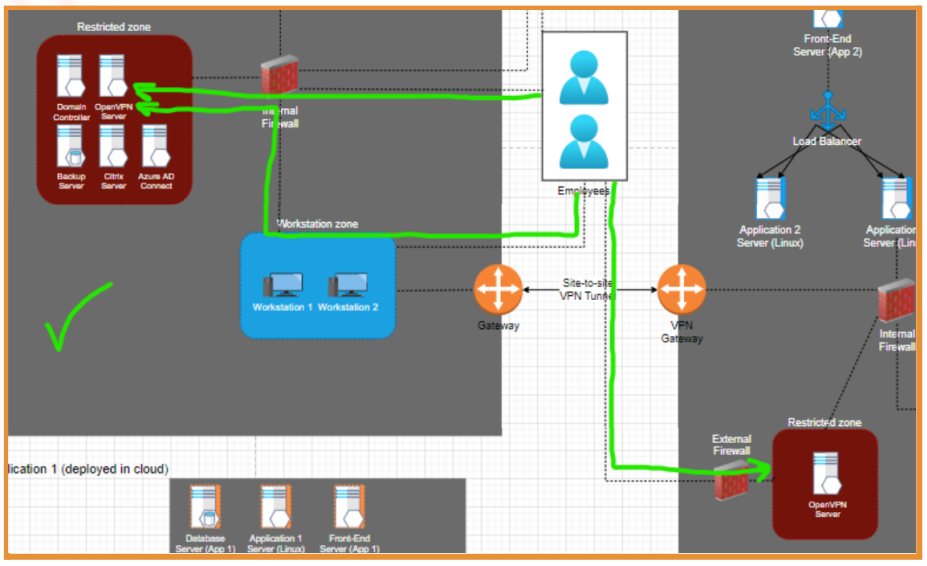
2. Creating secured access for the employees outside of their workstations.

Now if there are issues with connecting to the on-premise VPN server, the employees can lose access to the system, even if they are working from home.



Open VPN servers rarely fail and even if they do it is not complicated to bring them back online, so usually, this threat can be disregarded. However, for an e-commerce company, even temporary downtimes can lead to important revenue losses, which is why it is worth considering this risk.

Providing the employees with an alternative secure connection, which can be also used from home, will help ensure that no matter what happens, access to the system will be retained. The green lines in the diagram below indicate the possible ways an employee could reach a VPN server with the proposed solution implemented.



The alternative server would be deployed in the cloud in order to be fully independent of the on-prem one. Of course, it would also need a proper firewall configuration to monitor the work-from-home activities.

Implementation:
A simple OpenVPN server setup deployed together with the expansion of the cloud service utilization.

3. Using the old hardware to create a honeypot.

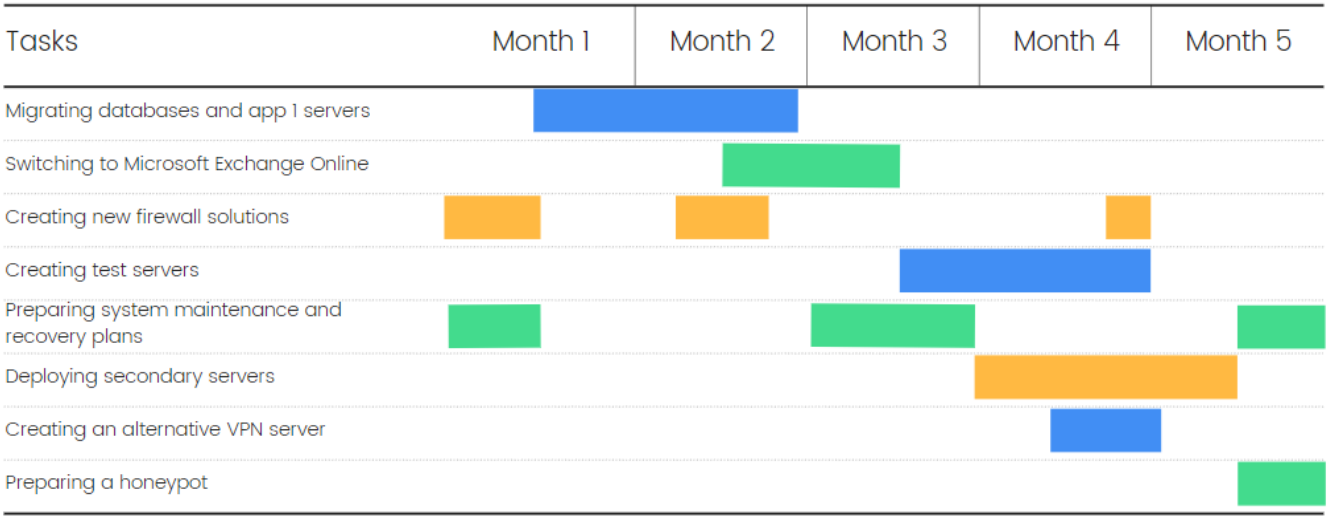
Honeypots are separate servers or networks that pose as bait for hackers. They are created and used to improve detection and protection from cyberattacks. While not being a priority security mechanism, it may trick amateur hackers and provide information on potential tools used to exploit vulnerabilities and vulnerabilities themselves. Old servers are a perfect candidate to implement such a system as they will not generate many additional costs.

Costs estimation

It is difficult to estimate exact costs as we were not given information on the total amount of storage and processing power required. We leave the calculation at your discretion using the software provided by Azure, which can be found here.

Network architecture changes roadmap

The following Gantt chart describes the change implementation process with regard to the importance of introducing those enhancements.



It should be noted that the exact timeframe is not necessarily accurate - the time of implementation depends on the workforce available. The most important division is the order of introducing the changes.

To-be security model

Password security	Vulnerability management	Database security	Phishing and other human error prevention	Network architecture and security
Storage	Procedures	Storage	Training	Efficiency
Strength	Time efficiency	Access configuration	Prevention systems	Cost efficiency
Authentication	KPI	Backup		Stability and failover capability
Maintenance	Business continuity			Designed security methods
Creation				Actual security
				Expansion capabilities

Key:

Outstanding	Appropriate	Suboptimal	Inappropriate	Very inappropriate

Summary

In this report we have provided guidelines to improve the IT infrastructure of the company. Updated password security can be ensured through stronger password requirements, hashing, improving registration procedures and enforced periodic password changes. Vulnerability management processes can improve security and time efficiency which ensures business continuity in the long run. Storing and accessing data may be made safer through hashing, encryption and regular backups. Phishing attacks require training and modern preventive software to eliminate all potential threats associated with them. Finally, the network architecture has a lot of room for improvement in both security, stability, failover capability, efficiency and further expansion capabilities.

Implementation of all of the suggested changes will significantly improve Eklamot's IT security, which is visualized in the "To-be security model". The IT systems are the backbone of every company, which is why it is very important not to ignore dangers and work toward the "To-be" system.