University of Amsterdam

Remote Acquisition Boot Environment (RABE): Visualizing the server side

Dennis Cortjens
dennis.cortjens@os3.nl

REPORT

3rd of January, 2015

Abstract

In the field of digital forensics the acquisition of multiple computers in large IT infrastructures have always been a complex and time consuming task. Especially when one doesn't know which computer to investigate and therefore needs to acquire them all. Triage software has increased the efficiency in cases like this. The software gives an indication which computers to acquire, but one still needs to disassemble and acquire storage devices of the specific computers on the crime scene.
Previous research created a proof of concept for automating the remote acquisition of multiple computers and tested the performance of this concept. However, it focused mainly on the client side of the concept. This (follow-up) research focused on the server side. It enhanced and visualized the concept's server side with a dashboard from which the administrative, legal and technical tasks of the remote acquisition process can be performed.

Contents

1 Introduction

Over the last 10 years our world has been more and more digitized. We have access to computers at school, our work and in public places. This has led to an increase of large IT infrastructures with multiple computers (clients and servers). Within these infrastructures the system administration is automated or can be done remotely. Unfortunately, this is not always the case in all IT fields.

1.1 Problem

In the field of digital forensics the acquisition of multiple computers in large IT infrastructures have always been a complex and time consuming task. Especially when one doesn't know which computer to investigate and therefore needs to acquire them all. Triage software has increased the efficiency in cases like this. The software gives an indication which computers to acquire, but one still needs to disassemble and acquire storage devices of the specific computers on the crime scene. At companies, data centres and universities this is quite an issue.

Previous research into the remote acquisition of computers created a proof of concept and tested the performance of this concept. However, it focused mainly on the client side of the concept. This (follow-up) research can enhance the server side by visualizing the administrative, legal and technical tasks of the remote acquisition process into a dashboard which brings the RABE proof of concept a step closer to be used in the field.

1.2 Position

A study on the remote acquisition of computers resulted in the following related material:

In 2013 Martin B. Koopmans and Joshua I. James wrote a paper on automated network triage. They described a working client-server network triage environment based on PXE booting and created a tool for investigating client computers in large IT environments, called the Automated Network Triage (ANT). They focused on the triage part of a forensic investigation. [1]

In 2014 I wrote a report on the remote acquisition of multiple computers with a bootable Linux CD / PXE [2] as will be mentioned in section 2.1.

In 2014 Eric van den Haak wrote a report on the remote data acquisition on block devices in large environments. He described copy-on-read and copy-on-write methods and tested these methods. He focused on the server side of the concept. [3]

1.3 Scope

The main question for this research is:

How can the server side of the RABE proof of concept be visualized into a dashboard from which the administrative, legal and technical steps of the remote acquisition process can be performed?

This question is researched by the following sub questions:

1. What are the administrative tasks for the remote acquisition process?
2. What are the legal tasks for the remote acquisition process?
3. What are the technical tasks for the remote acquisition process?
4. Which of these tasks can be visualized into a dashboard?
5. What is needed for the technical implementation of the dashboard?
6. What is the basic configuration of the server?
7. How can these be combined into the server side of the RABE proof of concept?

This research focuses on the server side of the remote acquisition.

It creates a server configuration with the following requirements:

- Generates a form with administrative and legal details for the public prosecutor or examining judge and hosting provider
- Creates a full forensic image of a storage device
- Provides in the re-export of iSCSI devices for connecting to an investigator's computer trough the server

2 Background

2.1 Previous research

Previous research created a proof of concept for the remote acquisition of computers, called the Remote Acquisition Boot Environment (RABE). The concept consists of a default RABE live image for the client, an authoring tool for configuring live images and a basic configuration for the server. The live image is not yet forensically sound and therefore cannot be used in the field. It was a proof of concept used to measure the speed and time of a remote acquisition. The test results led to the adoption of iSCSI for the concept. The live image boots a computer with a PXE or an optical disc and loads the Ubuntu environment. Within this environment it starts an OpenVPN connection to communicate securely with the server and distributes the storage devices to the server. The acquisition of the storage devices can be started from the server at any time. A schematic overview of this process is shown in figure 1. This concept focused mainly on the client side implementation and provided a basic server configuration. [2]

This research is a follow-up. It uses the client side implementation of the previous research. An additional flow chart diagram of the RABE client process is shown in figure 2. In this flow chart diagram the blue boxes represent user interaction and the orange an automatic process.

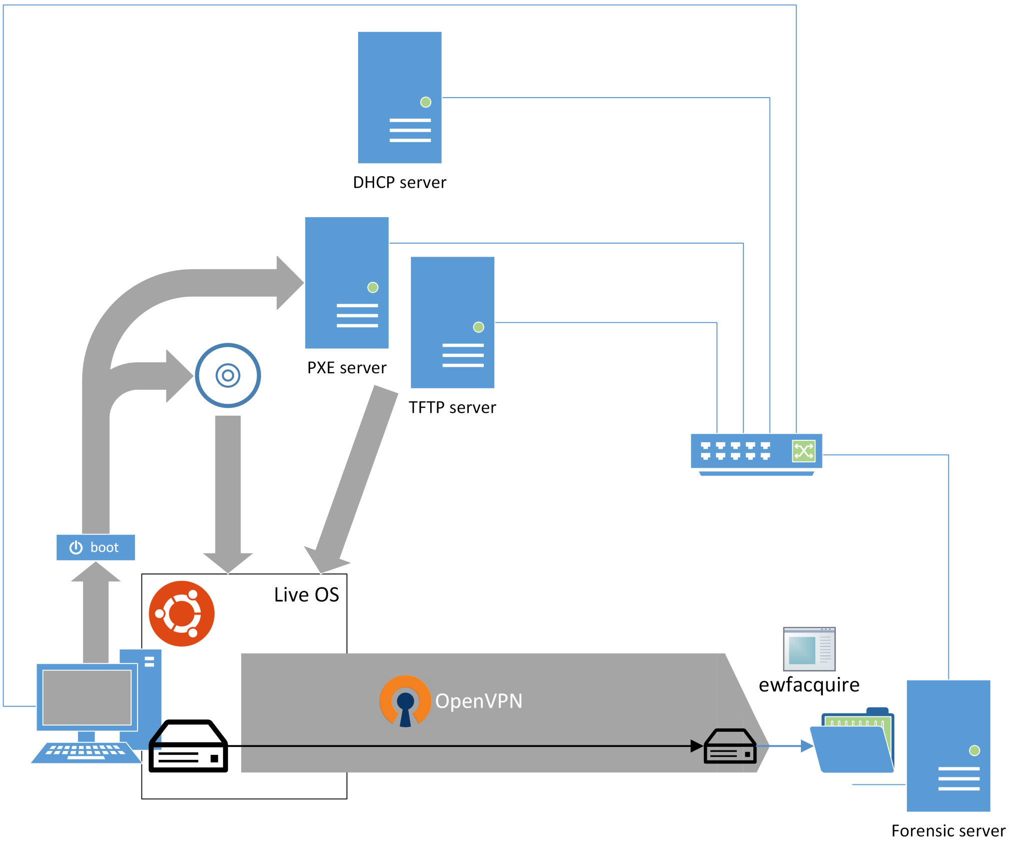


Figure 1: RABE client (schematic)

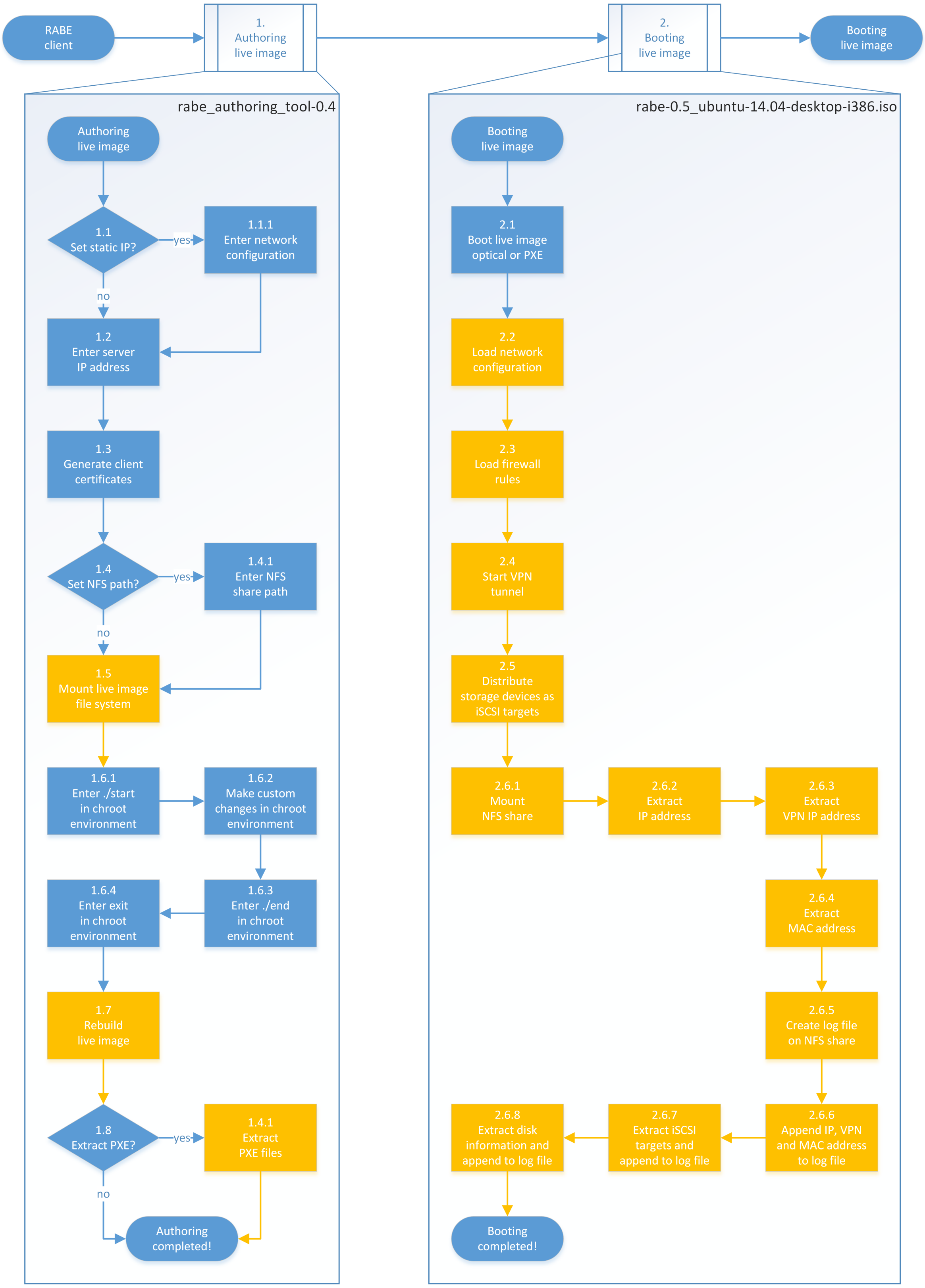

RABE client
1. Authoring live image
2. Booting live image
Booting live image
rabe_authoring_tool-0.4
Authoring live image
1.1 Set static IP?
yes
1.1.1 Enter network configuration
no
1.2 Enter server IP address
1.3 Generate client certificates
1.4 Set NFS path?
yes
1.4.1 Enter NFS share path
no
1.5 Mount live image file system
1.6.1 Enter ./start in chroot environment
1.6.2 Make custom changes in chroot environment
1.6.3 Enter ./end in chroot environment
1.6.4 Enter exit in chroot environment
1.7 Rebuild live image
1.8 Extract PXE?
yes
1.4.1 Extract PXE files
no
Authoring completed!
rabe-0.5_ubuntu-14.04-desktop-i386.iso
Booting live image
2.1 Boot live image optical or PXE
2.2 Load network configuration
2.3 Load firewall rules
2.4 Start VPN tunnel
2.5 Distribute storage devices as iSCSI targets
2.6.1 Mount NFS share
2.6.2 Extract IP address
2.6.3 Extract VPN IP address
2.6.4 Extract MAC address
2.6.5 Create log file on NFS share
2.6.6 Append IP, VPN and MAC address to log file
2.6.7 Extract iSCSI targets and append to log file
2.6.8 Extract disk information and append to log file
Booting completed!

Figure 2: RABE client flow chart diagram

2.2 Dutch Criminal System

The main sources of the Dutch criminal system are two criminal law books; one for criminal proceedings (Criminal Procedure Code) and one for criminal acts (Penal Code). The most important principle of Dutch criminal law is the principle of legality. This implies that no one will be found guilty of a crime without a prior penal law. The laws are enforced by the government; the police, public prosecutor's office and judiciary. The police has district departments and a central agency. Just like the public prosecutor's office that has district departments and a central office. The judiciary has district courts, courts of appeal and a supreme court.

A criminal case starts at the police who investigates the offence or felony. The investigation is conducted under legal supervision of the public prosecutor. The public prosecutor decides whether or not the suspect will be prosecuted. In the case of a serious and complex crime an examining judge is also part of the investigation. The examining judge, part of the judiciary, decides on the use of more severe investigative proceedings and/or means of coercion. This judge issues warrants, decides on the provisional detention, questions witnesses, interrogates suspects and appoints expert witnesses. Their role is not to prosecute the accused, but to gather facts and look for any evidence, incriminating or exculpatory. The examining judge does not sit on the trial court which makes a decision in the case.

In the Netherlands there is no jury system. The judges are the ones who decide whether or not the accused is guilty as charged and which penalty should follow. The general procedure is as follows: most cases are handled at a district court after which an appeal can follow before a court of appeal. The third and final instance to which one may appeal is the supreme court.

3 Research

3.1 Approach

To determine how the server side of the RABE proof of concept can be visualized into a dashboard, research is done on the process of a remote acquisition and its participants. The process includes the administrative, legal and technical tasks of a remote acquisition and are charted by the sub research questions. The legal tasks need to fit the Dutch criminal system. The server side configuration is taken from previous research and extended with the resources needed for the implementation of the dashboard. Eventually this is combined into a fully working proof of concept.

3.2 Process

The remote acquisition process consists of many tasks. It does not only contain the technical steps like authoring the live image, booting the image and acquiring the storage devices of the client to the server. There are also administrative and legal requirement to keep in mind. These are closely related in the process. Administratively the process requires a case and target, but also the cooperation of the hosting provider. Legally the process is bound to the Dutch criminal system. Depending on the target's location the public prosecutor or examining judge needs to issue a warrant. Despite the Dutch criminal system not yet being ready for such a process, this research will continue within the boundaries of the current system with recommendations for the future. This will be mentioned in section 3.3.

The during this research discovered tasks led to a flow chart diagram for a remote acquisition. The flow chart diagram is shown in figure 3 and each step is described in table 1. The steps are coloured according to their type. Administrative steps are blue, legal steps orange and technical steps green. A case will have different status labels during the process and these are included in the flow chart diagram and steps.

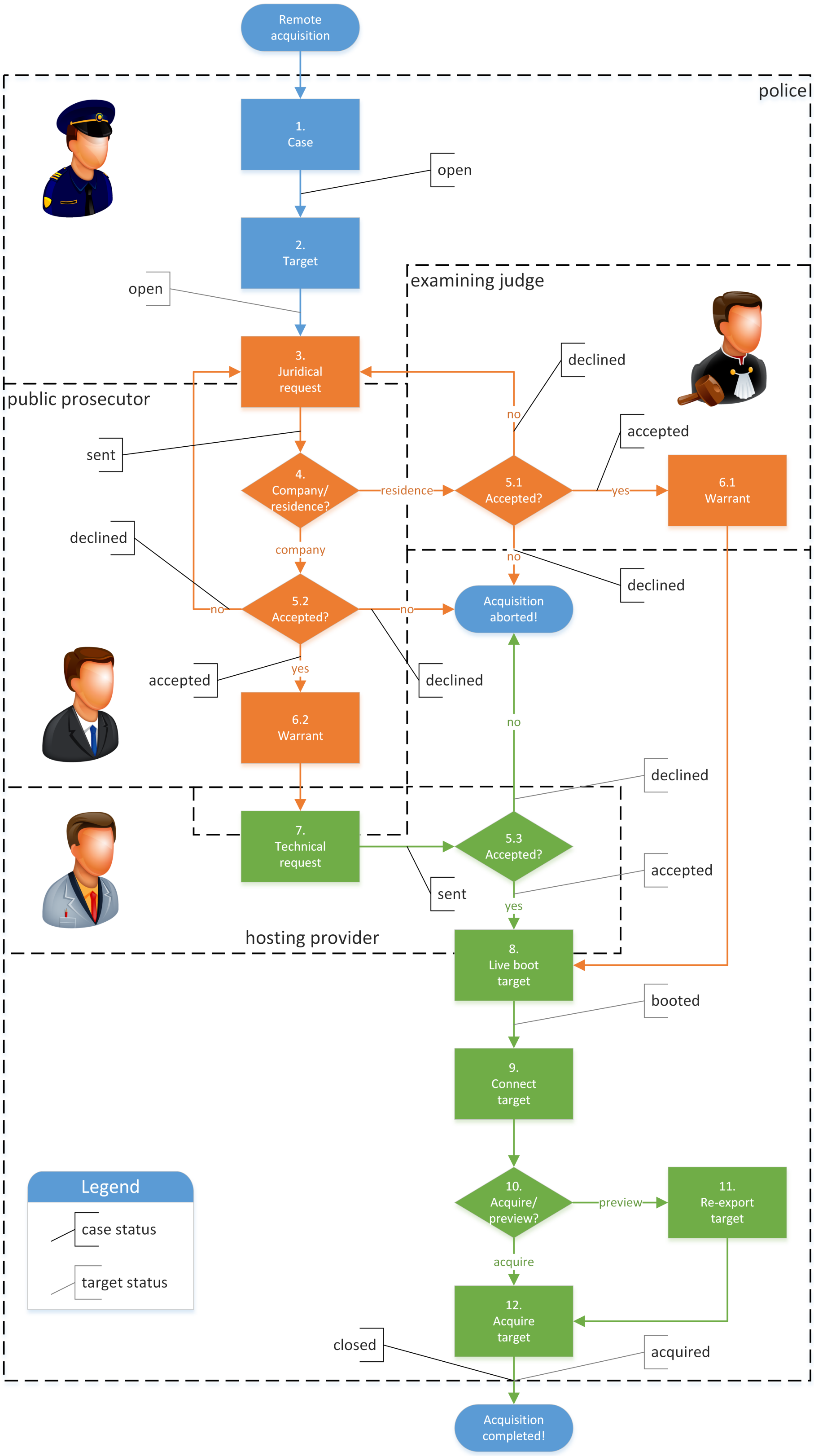


Figure 3: Remote acquisition flow chart diagram

Step	Name	Description
1	Case	A criminal case in which a remote acquisition is needed. After creation the status of the case is 'open'.
2	Target	The remote target that needs to be acquired. After creation the status of the target is 'open'.
3	Legal request	Create a legal request for the acquisition of the target. After sending, the status of the case is 'sent'.
4	Company/residence?	Determine whether the target resides within a company or residence and according to this send it to the public prosecutor (company) or examining judge (residence).
5.1	Accepted?	The request is accepted or rejected by the examining judge. After, the status of the case is 'accepted' or 'rejected'.
5.2	Accepted?	The request is accepted or rejected by the public prosecutor. After, the status of the case is 'accepted' or 'rejected'.
5.3	Accepted?	The request is accepted or rejected by the hosting provider. After, the status of the target is 'accepted' or 'rejected'.
6.1	Warrant	The examining judge issued a warrant.
6.2	Warrant	The public prosecutor issued a warrant.
7	Technical request	Create a technical request for the acquisition of the target. After, the status of the case is 'sent'.
8	Live boot target	Live boot the target with a RABE live image. After, the status of the target is 'booted'.
9	Connect target	Connect to the distributed storage device of the target.
10	Acquire/preview?	Determine whether to acquire or preview (live investigate) the distributed storage device of the target.
11	Re-export target	Re-export the distributed storage device of the target to preview it on an investigator's computer.
12	Acquire target	Acquire the distributed storage device of the target. After, the status of the target is 'acquired' and if all targets in the case are, the status of the case is 'closed'.

Table 1: Remote acquisition flow chart steps

3.2.1 Administrative tasks

The administrative tasks are mostly tasks related to the management of the case. These blend in with the legal and technical tasks on various subjects. The main objectives of these administrative tasks are ensuring a forensically sound process and giving a clear view of all actions which have been performed.

case management For the process to start, it requires a criminal case that needs a remote acquisition. This is possible in all kinds of cases. The process needs the registration of the case with all its characteristics like: case number and name, treating investigator, location information and others. This information is also needed for the legal request to the public prosecutor or examining judge and is therefore closely related to the legal tasks.

target management For the process to continue, it requires a target. This target needs to be identified by an IP address or hostname. It does not matter that the target resides within a company or residence, because the process provides a workflow for both. In the case of a residence it would be a local search in which a remote acquisition is performed on the scene. In the case of a company it is based on a single computer (usually server) that is hosted by a provider and needs to be acquired remotely. It needs the registration of the target with all its characteristics like: IP and MAC address, storage devices and others. This information is also needed for the technical request to the hosting provider and for other technical proceedings. Therefore it is closely related to the technical tasks.

request management The sending and receiving of requests is an administrative task, although it may have a legal character when obtaining a warrant from the public prosecutor or examining judge or have a technical character when handing over the warrant to the hosting provider. A request needs the legal and technical information needed for acceptance. Especially the legal aspects are very important, because without a warrant there will be no remote acquisition at all. This information needs to recorded.

file management The process requires the storage of files. This includes the storage of the digital version of the warrant, the files of the acquisition and the logs of the (technical) tasks. This is again closely related to the legal and technical tasks.

logging All actions (administrative, legal and technical) need to be logged. This defines a forensically sound process and gives a clear view of all the actions that have been performed.

status labelling All objects have various states during the process. These need to be recorded to keep track of the progress. This contributes to the speed and forensic soundness of the process.

3.2.2 Legal tasks

warrant request The Dutch criminal system requires the police to request a warrant according to certain rules and requirements before it is issued by the relevant authority. A request needs to contain a case number and an extensive case description in which the need for the warrant is clearly defined. Besides that the case has to meet some legal requirements as well. These will be mentioned in section 3.3.

request acceptance/rejection The relevant authority has to accept or reject the request. The relevant authority depends on the type of location in which the target resides. A company is the jurisdiction of the public prosecutor. A residence is the jurisdiction of the examining judge, because this is a profound invasion of the privacy as stated in European law and jurisprudence. This will be mentioned in section 3.3.

digital warrant If the request is accepted, the relevant authority will issue a warrant. This document is printed and signed by the authority before it is legal. The warrant is often scanned and send by email. In fact one can speak of a digital warrant.

warrant hand over The Dutch criminal system requires the relevant authority to hand over a copy of the warrant to the rightful claimant of the target's location. At this moment this needs to be a hard copy of the warrant.

3.2.3 Technical tasks

company cooperation The process requires the cooperation of the company. In most cases this will be a hosting provider. Therefore the request and warrant need to be send to the hosting provider as the legal basis for the acquisition. The hosting provider is obligated by law to cooperate. If the acquisition is remote, some actions need to be performed before being able to remotely connect to the target without being on the scene as an investigator.

chain of evidence The chain of evidence is the most important aspect of a forensically sound process. It is part of both the legal and technical domain. From a legal perspective evidence can only be used when it is unaltered during the investigation. This requires the technical part to be clear and well-documented. Normally, the chain of evidence is ensured by the forensic investigator, but with a third party (hosting provider) involved this cannot be done completely. This requires for clear guidelines on tasks and reporting, bound by legislation.

live boot target The target needs to be booted with the RABE live image. The image first needs to be created, but that part of the process is beyond the scope of this research. The image boots an Ubuntu environment from which it starts an OpenVPN connection to communicate securely with the server and distributes the storage devices to the server.

connect target From the server, the distributed devices of the target need to be connected in order to carry out sequel actions.

acquire target The server needs to create a full forensic image of the storage devices of the target as mentioned in section 1.3.

re-export target The server needs to provide in the re-export of the storage devices of the target for an investigator's computer to connect and perform a preview (live investigation) as mentioned in section 1.3.

3.2.4 Participants

The process involves four important participants depending on the workflow. Each participant needs to perform certain actions from the dashboard. These are illustrated with use case diagrams.

police The police has the most tasks in the process and dashboard, because they are the investigative authority in the Dutch criminal system. Therefore most of the administrative tasks reside with them. First the police needs to be able to login to the dashboard. Due to case and target management they need to perform related tasks. There is a high dependency between the case and target object. A case consists of one or more targets which makes a target an extension of case. A legal task to be performed by the police is sending the legal request to the relevant authority. The request contains the case information. The same applies to the technical request to the hosting provider. It cannot be send without a warrant and so depends on the legal request. From a target, its distributed storage devices can be connected, followed by an acquisition or re-export of the devices. The merge action is for merging the temporary information, posted by the target (client), with the actual target in the case. This information is needed to be able to connect to the target. This is shown in figure 4.

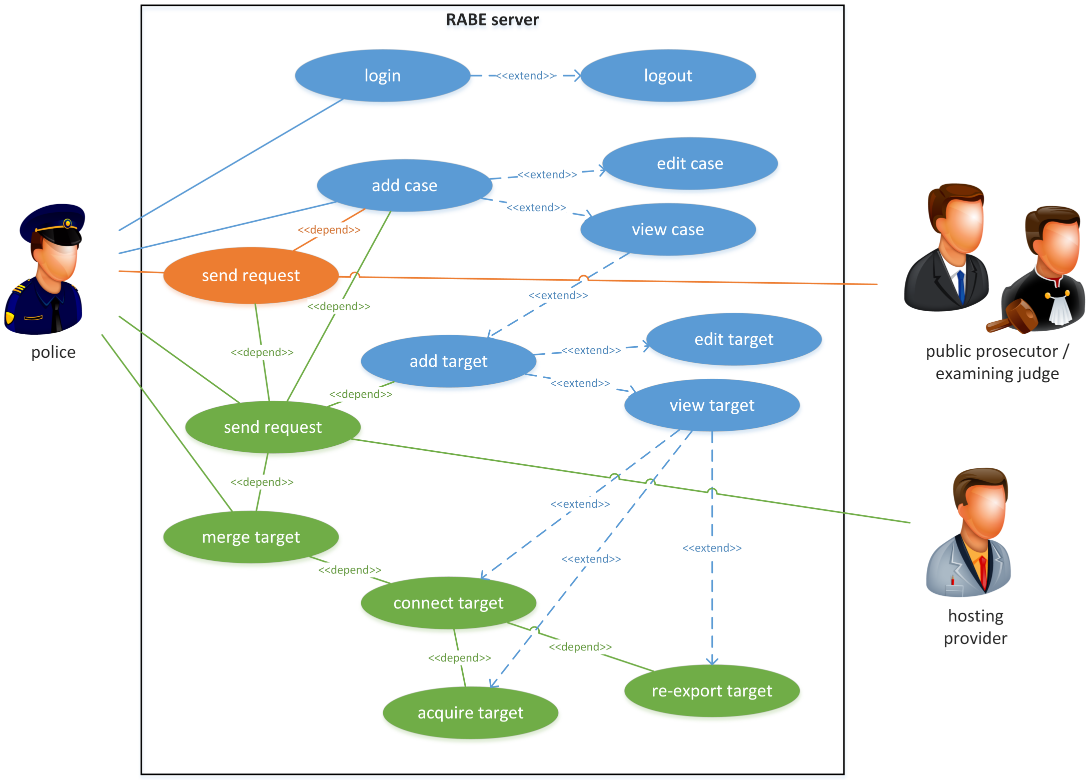


Figure 4: Use case diagram police

public prosecutor The public prosecutor also needs to login and has an overview of all requests. A request needs to be viewed and will show the case number and description. The location on which the target resides should also be showed in order to determine who will issue the warrant. If the location is a company the public prosecutor needs to be able to accept or reject the request. On acceptance the public prosecutor need to issue and upload the warrant. If the location is a residence the public prosecutor needs to forward the request to the examining judge. This is shown in figure 5.

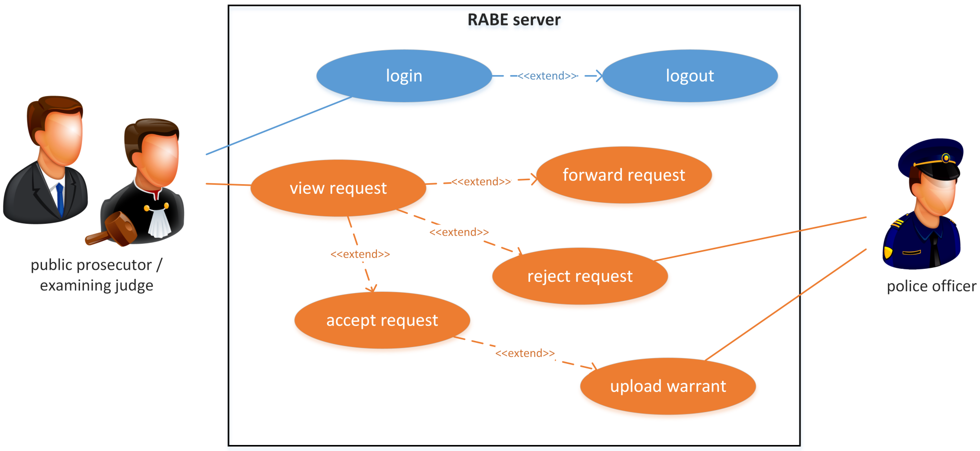


Figure 5: Use case diagram public prosecutor / examining judge

examining judge The examining judge requires the same tasks as the public prosecutor. The only difference is that one needs to have an overview of all requests related to residences. For this the same use case diagram applies as shown in figure 5. The dashboard offers a strict separation of concerns between authorities and companies which is covered by a login system with separated pages and multiple checks to prevent access to one another's part of the dashboard.

hosting provider The hosting provider again needs to login to the dashboard and also needs an overview of all requests, but in this case only the technical requests. It needs to be able to view and download the warrant and for each target accept or reject. If accepted the hosting provider should be able to create a RABE live image to boot the specific target or download a general RABE live image. However, the creation of the image within the dashboard is beyond the scope of this research. After the target is booted with the live image it needs to be set as booted, so the police knows the target is ready.

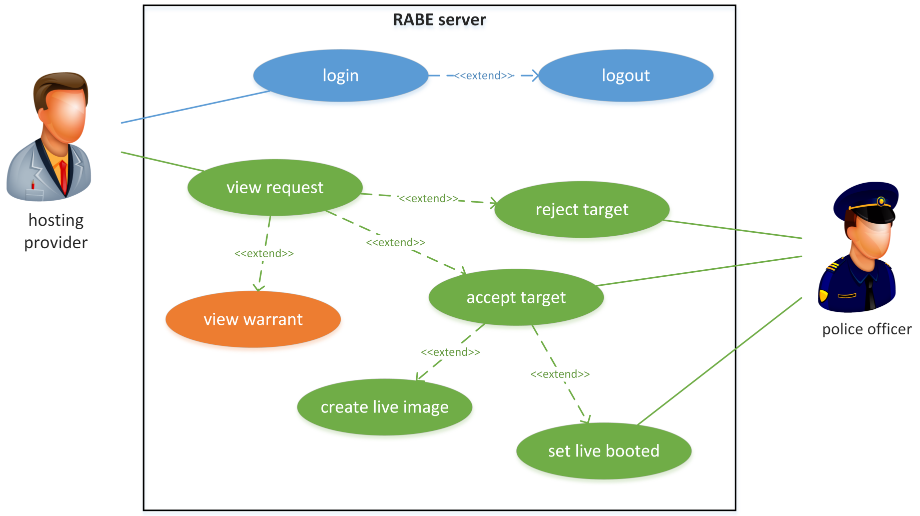


Figure 6: Use case diagram hosting provider

3.3 Legal

The legal basis for a remote acquisition should be found in the Dutch criminal system. However, the law book for criminal proceedings (Criminal Procedure Code) does not provide an exact framework for a remote acquisition. Despite this, there are proceedings that do provide some aspects of the legal framework to perform such an acquisition.

In the current criminal system, the acquisition of a computer at a hosting provider is performed based on article 125i of the Criminal Procedure Code. The article states that the examining judge, public prosecutor, assistant public prosecutor and police officer are authorized to perform a search with the purpose of acquiring data stored or recorded on a storage device on location.

Article 125i is an extension of the authority to search places, but does not provide all mentioned people to conduct a search on their own. The right to conduct a search in a place depends on its location and circumstances. Article 96c gives the public prosecutor the authority to conduct a search in every place, except a home without the consent of the occupant or an office of a person with certain legal privileges. In case of a pressing necessity this authority is delegated to the assistant public prosecutor. In the Dutch criminal system the assistant public prosecutor is a high ranked police officer. Article 110 gives the full authority to conduct a search in any place to the examining judge. In case of a pressing necessity this authority is delegated to the public prosecutor by article 97.

Article 96c Criminal Procedure Code[*]
1.
In case of a criminal offence caught in the act or in case of suspicion of a crime as defined in article 67, paragraph 1, the public prosecutor is authorized to search any place, except a home without the consent of the occupant or an office of a person as referred to in article 218.
2.
In case of pressing necessity and if action of the public prosecutor cannot be postponed, the assistant public prosecutor may exercise this jurisdiction. He will need permission from the public prosecutor. If urgently required or the permission of the public prosecutor cannot be asked for in time, the permission can be granted by the public prosecutor within three days after the search. If the public prosecutor denies permission, he makes sure the effects of the search will be undone.
3.
Searching places in accordance with the provisions of paragraph 1 shall be carried out under the direction of the public prosecutor or, in case of application of the second paragraph, under the direction of the assistant public prosecutor.
...

Article 110 Criminal Procedure Code[*]
1.
The examining judge can search any place on request by the public prosecutor or by his own motion, if he pursues investigative measures according to articles 181 to 183. He can appoint the persons who will join him. The request will mention the criminal act and the name of the suspect or, if the name is unknown, an as accurate as possible description of the suspect, as well as the facts and circumstances which make clear that the legal conditions for exercising the competence for the search are fulfilled.
2.
The search of places in accordance with the provisions of paragraph 1 shall be directed by the examining judge and in the presence of the public prosecutor or, in case of his absence, an assistant public prosecutor.
...

Article 125i Criminal Procedure Code[*]
The examining judge, the public prosecutor, the assistant public prosecutor and the police officer have under the same conditions as provided in articles 96b, 96c, first, second and third paragraph, 97, first through fourth paragraph, and 110, first and second paragraph, an entitlement to search a place in order to acquire data stored or recorded on a storage device at this place. In the interest of the investigation they can capture this data. Articles 96, 98, 99 and 99a shall apply accordingly.

These articles provide the legal framework for an acquisition on a crime scene. A combination of articles 96c and 125i give the public prosecutor and digital forensic investigator, who is a police officer, the authority to search a company and perform an acquisition. In this case the article 125i warrant is issued by the public prosecutor. With the combination of articles 110 and 125i the examining judge has the authority to search a residence and perform an acquisition. In practice the digital forensic investigator will perform

[*] The Dutch Criminal Procedure Code has no official English translation. These are translations from the original Dutch articles. The Dutch texts are mentioned in appendix A.

the actual acquisition under the authority of the examining judge. In this case the article 125i warrant is issued by the examining judge, because it involves a search in a residence. This explains the legal steps 3 to 6 in the flow chart diagram and steps as previously shown in figure 3 and described in table 1 of section 3.2.

Despite article 125i not mentioning how the acquisition of the data should be performed, it does require a physical presence on location. So article 125i only limitedly provides a legal framework for this proof of concept. It covers the aspect of *acquiring data stored or recorded on a storage device*.

Another article that provides aspects of the legal framework for this proof of concept is article 125k. The article states that in extension to article 125i a person who has knowledge of the security of a computer system is obligated to provide access to that system, including decryption. This does not provide a full legal framework for a remote acquisition, but does address the aspect of *cooperation of someone who has access to the system*.

The chain of evidence of a remote acquisition can be ensured in two manors; by appointing an expert witness or ordering an article 125k warrant. The Dutch criminal system has a list of third party expert witnesses who can be appointed by the public prosecutor in order to use their knowledge in criminal cases. If a third party expert is not on the list, an examining judge can appoint that expert to be an expert witness. This addresses the use of a third party within the chain of evidence. For a remote acquisition the examining judge could appoint an employee of the hosting provider to be a temporary expert witness for that specific case. Another manor can be article 125k which bounds the tasks of a third party to what is ordered by the examining judge in the *interest of the investigation*. In this matter the forensic investigator should provide the examining judge the technical tasks of the order. Again, both options do not provide a full legal framework for ensuring the chain of evidence in the current system, but addresses a possible framework for the future.

> Article 125k Criminal Procedure Code[*]
> 1.
> In the interest of the investigation and in extension to article 125i or
> 125j, an order can be addressed to the one who has presumably knowledge
> of the security of a computer system, to provide access to that system or
> its parts. The one to whom the order is addressed has to comply by
> providing knowledge about the security of the system.
> 2.
> If encrypted data is found within the computer system the first paragraph
> shall apply accordingly. The order is addressed to the one who has
> presumably knowledge of the encryption of the data.
> 3.
> The order referred to in the first paragraph, is not given to the
> suspect. Article 96a paragraph 3 shall apply accordingly.

This concludes that at this moment there is no complete legal framework to cover a remote acquisition, but can be made possible in the near future with jurisprudence to these article or new legislation. There could be a complete legal framework for such an acquisition, if jurisprudence to article 125i would conclude that the search does not have to be performed physically on the scene. And if jurisprudence to article 125k would conclude that the obligation to provide access to the system also applies to the hosting provider booting the target with a live image.

One really needs to understand that this proof of concept is not a form of hacking which is still illegal within the Dutch criminal system. In the case of hacking one gets access to a system by breaking its security without the knowledge of any of the participants that provide the system, including owner and hosting provider. This proof of concept is based on the idea of an acquisition that could be performed under article 125i on the scene, but because of capacity and time management could be performed remotely. Despite not having a complete legal framework, this research continues on this basis.

[*] The Dutch Criminal Procedure Code has no official English translation. These are translations from the original Dutch articles. The Dutch texts are mentioned in appendix A.

4 Functional design

This section describes the functional design of the concept which required the following features:

1. providing case administration and management
2. providing target administration and management
3. generating request forms for the examining judge, public prosecutor and hosting provider
4. connecting to targets' distributed storage devices
5. acquiring a target's storage device
6. providing the re-export of a target's storage device

Each step of the process describes actions and objects. An action is marked *italic* and an object is marked **bold**.

4.1 Case (1)

The process starts at the police with a criminal case which needs the remote acquisition of a server.

The police *logs* into the Remote Acquisition Boot Environment (RABE) dashboard and *creates* a new **case**. The case requires a case number and case description (mandatory information). The number relates to the case in other computer systems (reference). The description is a summary of the case for participants further down the process. Additionally, a case name can be added (optional information). The name is the original name of the case used by the police and public prosecutor. When a case is created, the case ID, case status, case path, case creation date and case creation user will be set (automatic information). The ID is an incremented number within the RABE dashboard. The status is one of the statuses defined for the object within the dashboard (open). The path is the default case directory on the system that runs the dashboard. The creation date and user are the date on which the case is created and the user that created the case. The case information can be *edited* and *viewed*.

This addresses case management and file management (case path) as mentioned in section 3.2.1.

Action	:	case (create, edit and view)
From participant	:	police
For participant(s)	:	police, public prosecutor, examining judge and hosting provider
Description	:	A case object with information related to the case which can be created, edited and viewed.
Mandatory information	:	case number, case description
Optional information	:	case name
Automatic information	:	case ID, case status, case path, case creation date, case creation user
Case status	:	open

4.2 Target (2)

The remote acquisition requires a target; the server.

The police *creates* a new **target**. The target is related to a **case** and requires a target number and target IP address (mandatory information). The number relates to the target in other computer systems (reference). The IP address is the (public) IP address of the target. When a target is created, the target ID, related case ID, target status, target creation date and target creation user will be set (automatic information). The target ID is an incremented number within the RABE dashboard. The case ID is the ID of the case it is related to. The status is one of the statuses defined for the object within the dashboard (open). The creation date and user are the date on which the target is created and the user that created the target. The target information can be *edited* and *viewed*.

This addresses target management as mentioned in section 3.2.1.

Action	:	target (create, edit and view)
From participant	:	police
For participant(s)	:	police, public prosecutor, examining judge and hosting provider
Description	:	A target object with information related to the target which can be created, edited and viewed.
Mandatory information	:	target number, target IP address
Optional information	:	none
Automatic information	:	target ID, related case ID, target status, target creation date, target creation user
Case status	:	open
Target status	:	open

4.3 Legal request (3)

The remote acquisition requires a warrant from the public prosecutor or examining judge. The police requests the warrant, but cannot contact an examining judge directly. All requests are send to the public prosecutor. This is bound by law.

The police *creates* a new legal **request** for the public prosecutor. The request is related to a **case** and through a case to at least one **target**. It requires additional case information; location type, location name (optional), location address and location city. The type is a company or residence which defines the jurisdiction for the warrant. The name is the name of the company. The address and city are the address and city of the location on which the related targets are hosted.

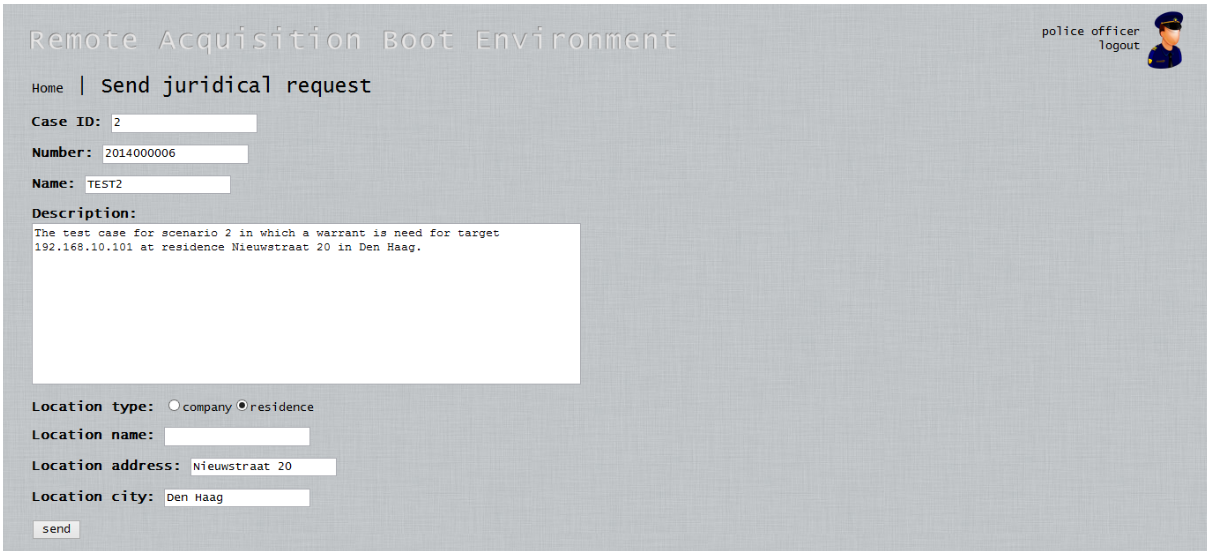
When a request is created, the case status, request ID, related case ID, request status, request creation date and request creation user will be set (automatic information). The case status is one of the statuses defined for the object within the dashboard (sent). The request ID is an incremented number within the RABE dashboard. The case ID is the ID of the case it is related to. The request status is one of the statuses defined for the object within the dashboard (open). The creation date and user are the date on which the request is created and the user that created the request.

After a request is created, the case it relates to can no longer be edited. This ensures the integrity of the case information in the dashboard for the review process by the public prosecutor and/or examining judge.

This addresses request management as mentioned in section 3.2.1 and warrant request as mentioned in section 3.2.2.

A preview of the legal request is shown in figure 7.

Action	:	request (create)
From participant	:	police
For participant(s)	:	public prosecutor
Description	:	A request object with information related to the case which can be created.
Mandatory information	:	(case) location type, location address, location city
Optional information	:	(case) location name
Automatic information	:	case status request ID, related case ID, request status, request creation date, request creation user
Case status	:	sent
Target status	:	open
Request status	:	open



Send juridical request

Case ID: 2

Number: 2014000006

Name: TEST2

Description:

The test case for scenario 2 in which a warrant is need for target 192.168.10.101 at residence Nieuwstraat 20 in Den Haag.

Location type: ○ company ● residence

Location name:

Location address: Nieuwstraat 20

Location city: Den Haag

send

Figure 7: RABE dashboard legal request (police)*

4.4 Company/residence? (4)

The public prosecutor receives the request and handles it based on the jurisdiction of its location type. If the request is for a company it is within the jurisdiction of the public prosecutor. If the request is for a residence it is within the jurisdiction of the examining judge.

company:
No action, continue at step 5.2 (accepted?).

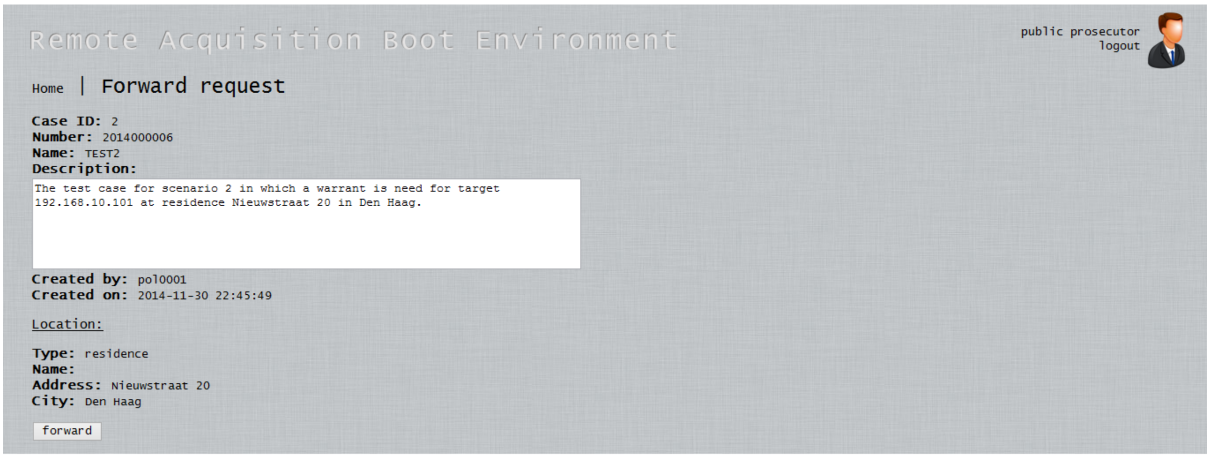
residence:
The public prosecutor *logs* into the RABE dashboard and *forwards* the **request** to the examining judge. This does not require any information input or setting.

This addresses request management as mentioned in section 3.2.1.

A preview of forwarding the request is shown in figure 8.

Action	:	request (view and forward)
From participant	:	public prosecutor
For participant(s)	:	examining judge
Description	:	A request object with information related to the case which can be viewed and forwarded.
Mandatory information	:	none
Optional information	:	none
Automatic information	:	none
Case status	:	sent
Target status	:	open
Request status	:	open

* During development, juridical is changed to legal. This screenshot is taken before this change.



Forward request

Case ID: 2
Number: 2014000006
Name: TEST2
Description:

The test case for scenario 2 in which a warrant is need for target 192.168.10.101 at residence Nieuwstraat 20 in Den Haag.

Created by: pol0001
Created on: 2014-11-30 22:45:49

Location:

Type: residence
Name:
Address: Nieuwstraat 20
City: Den Haag

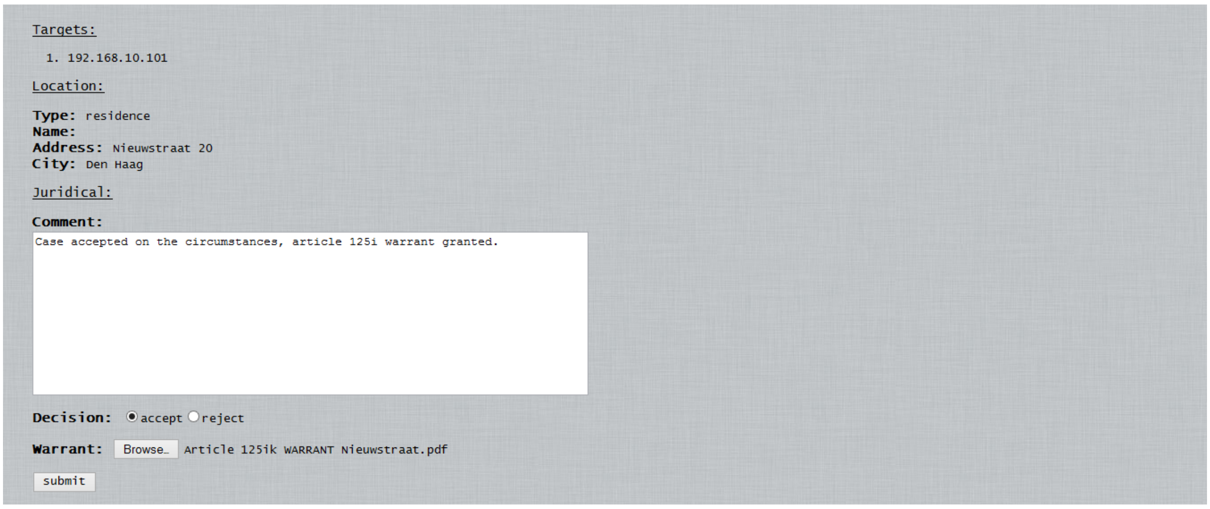
Figure 8: RABE dashboard forward request (public prosecutor)

4.5 Accepted? (5.1)

The examining judge views the request and accepts or rejects it.

accept:
The examining judge *logs* into the RABE dashboard and *views* the **request**. The examining judge *reviews* the case information and *accepts* the request. It requires additional case information; legal comment, legal decision and warrant (mandatory information). The comment is a summary of the examining judge's decision to accept the request. The decision is: accept. For the warrant, continue at step 6.1 (warrant).

A preview of accepting the request is shown in figure 9.



Targets:

1. 192.168.10.101

Location:

Type: residence
Name:
Address: Nieuwstraat 20
City: Den Haag

Juridical:

Comment:

Case accepted on the circumstances, article 125i warrant granted.

Decision: ● accept ○ reject

Warrant: Article 125ik WARRANT Nieuwstraat.pdf

Figure 9: RABE dashboard legal request*

* During development, juridical is changed to legal. This screenshot is taken before this change.

reject:
The examining judge *logs* into the RABE dashboard and *views* the **request**. The examining judge *reviews* the case information and *rejects* the request. It requires additional case information; legal comment and legal decision (mandatory information). The comment is a summary of the examining judge's decision to reject the request. The decision is: reject.

When a request is rejected, the case status, legal decision date, legal decision user and request status will be set (automatic information). The case status is one of the statuses defined for the object within the dashboard (rejected). The legal decision date and user are the date on which the decision is made and the examining judge that made the decision. The request status is one of the statuses defined for the object within the dashboard (closed).

After a request is rejected, return to step 3 (legal request) if additional case information should be provided. Otherwise, the acquisition process is aborted!

Action	:	request (view and reject)
From participant	:	examining judge
For participant(s)	:	police
Description	:	A request object with information related to the case which can be viewed and rejected.
Mandatory information	:	(case) legal comment, legal decision
Optional information	:	none
Automatic information	:	case status, legal decision date, legal decision user request status
Case status	:	rejected
Target status	:	open
Request status	:	closed

This addresses request management as mentioned in section 3.2.1 and request acceptance/rejection as mentioned in section 3.2.2.

4.6 Warrant (6.1)

The examining judge issues the warrant.

The examining judge *uploads* a digital version of the warrant and *submits* the decision. When a request is accepted, the case status, legal decision date, legal decision user and request status will be set (automatic information). The case status is one of the statuses defined for the object within the dashboard (accepted). The legal decision date and user are the date on which the decision is made and the examining judge that made the decision. The request status is one of the statuses defined for the object within the dashboard (closed).

This addresses request management and file management (file upload only) as mentioned in section 3.2.1. It also addresses request acceptance/rejection and digital warrant as mentioned in section 3.2.2.

A preview of accepting the request is previously shown in figure 9 of section 4.5.

Action	:	file upload (warrant)
From participant	:	examining judge
For participant(s)	:	police and hosting provider
Description	:	The file upload of the warrant to the case directory which can be viewed.
Mandatory information	:	warrant file
Optional information	:	none
Automatic information	:	none
Case status	:	sent
Target status	:	open
Request status	:	open

Action	:	request (view and accept)
From participant	:	examining judge
For participant(s)	:	police and hosting provider
Description	:	A request object with information related to the case which can be viewed and accepted.
Mandatory information	:	(case) legal comment, legal decision, warrant
Optional information	:	none
Automatic information	:	case status, legal decision date, legal decision user request status
Case status	:	accepted
Target status	:	open
Request status	:	closed

4.7 Accepted? (5.2)

The public prosecutor views the request and accepts or rejects it.

accept:
The public prosecutor *views* the **request**. The public prosecutor *reviews* the case information and *accepts* the request. It requires additional case information; legal comment, legal decision and warrant (mandatory information). The comment is a summary of the public prosecutor's decision to accept the request. The decision is: accept. For the warrant, continue at step 6.2 (warrant).

A preview of accepting the request is previously shown in figure 9 of section 4.5.

reject:
The public prosecutor *views* the **request**. The public prosecutor *reviews* the case information and *rejects* the request. It requires additional case information; legal comment and legal decision (mandatory information). The comment is a summary of the public prosecutor's decision to reject the request. The decision is: reject.

When a request is rejected, the case status, legal decision date, legal decision user and request status will be set (automatic information). The case status is one of the statuses defined for the object within the dashboard (rejected). The legal decision date and user are the date on which the decision is made and the public prosecutor that made the decision. The request status is one of the statuses defined for the object within the dashboard (closed).

After a request is rejected, return to step 3 (legal request) if additional case information should be provided. Otherwise, the acquisition process is aborted!

Action	:	request (view and reject)
From participant	:	public prosecutor
For participant(s)	:	police
Description	:	A request object with information related to the case which can be viewed and rejected.
Mandatory information	:	(case) legal comment, legal decision
Optional information	:	none
Automatic information	:	case status, legal decision date, legal decision user request status
Case status	:	rejected
Target status	:	open
Request status	:	closed

Again, this addresses request management as mentioned in section 3.2.1 and request acceptance/rejection as mentioned in section 3.2.2.

4.8 Warrant (6.2)

The public prosecutor issues the warrant.

The public prosecutor *uploads* a digital version of the warrant and *submits* the decision. When a request is accepted, the case status, legal decision date, legal decision user and request status will be set (automatic information). The case status is one of the statuses defined for the object within the dashboard (accepted). The legal decision date and user are the date on which the decision is made and the public prosecutor that made the decision. The request status is one of the statuses defined for the object within the dashboard (closed).

Again, this addresses request management and file management (file upload only) as mentioned in section 3.2.1. It also addresses request acceptance/rejection and digital warrant as mentioned in section 3.2.2.

A preview of accepting the request is previously shown in figure 9 of section 4.7.

Action	:	file upload (warrant)
From participant	:	public prosecutor
For participant(s)	:	police and hosting provider
Description	:	The file upload of the warrant to the case directory which can be viewed.
Mandatory information	:	warrant file
Optional information	:	none
Automatic information	:	none
Case status	:	sent
Target status	:	open
Request status	:	open

Action	:	request (view and accept)
From participant	:	public prosecutor
For participant(s)	:	police and hosting provider
Description	:	A request object with information related to the case which can be viewed and accepted.
Mandatory information	:	(case) legal comment, legal decision, warrant
Optional information	:	none
Automatic information	:	case status, legal decision date, legal decision user request status
Case status	:	accepted
Target status	:	open
Request status	:	closed

4.9 Technical request (7)

The remote acquisition requires the cooperation of the company; the hosting provider. The police requests this cooperation with the warrant from the public prosecutor or examining judge.

The police *creates* a new technical **request** for the hosting provider. The request is related to a **case** and through a case to at least one **target**. This does not require any information input. When a request is created, the case status, request ID, related case ID, request status, request creation date and request creation user will be set (automatic information). The case status is one of the statuses defined for the object within the dashboard (sent). The request ID is an incremented number within the RABE dashboard. The case ID is the ID of the case it is related to. The request status is one of the statuses defined for the object within the dashboard (open). The creation date and user are the date on which the request is created and the user that created the request.

This addresses request management as mentioned in section 3.2.1.

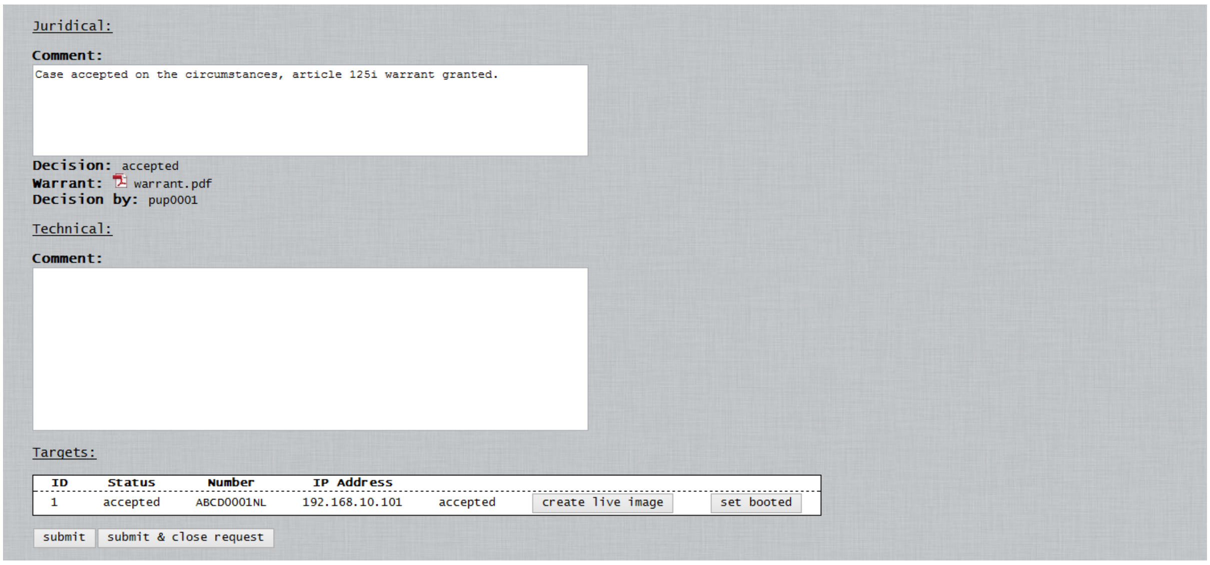
Action	:	request (create)
From participant	:	police
For participant(s)	:	hosting provider
Description	:	A request object with information related to the case which can be created.
Mandatory information	:	none
Optional information	:	none
Automatic information	:	case status request ID, related case ID, request status, request creation date, request creation user
Case status	:	sent
Target status	:	open
Request status	:	open

4.10 Accepted? (5.3)

The hosting provider views the request, including the warrant, and accepts or rejects the related targets.

accept:
The hosting provider *views* the **request**. The hosting provider *reviews* the warrant and *accepts* a **target** from the list. When a target is accepted, the target status will be set (automatic information). The target status is one of the statuses defined for the object within the dashboard (accepted).

A preview of an accepted target is shown in figure 10.



Juridical:

Comment:

Case accepted on the circumstances, article 125i warrant granted.

Decision: accepted
Warrant: warrant.pdf
Decision by: pup0001

Technical:

Comment:

Targets:

ID	Status	Number	IP Address			
1	accepted	ABCD0001NL	192.168.10.101	accepted	create live image	set booted

Figure 10: RABE dashboard technical request (hosting provider)*

Action	:	target (accept)
From participant	:	hosting provider
For participant(s)	:	police
Description	:	A target object with information related to the target which can be accepted from a list of related targets.
Mandatory information	:	none
Optional information	:	none
Automatic information	:	target status
Case status	:	accepted
Target status	:	accepted
Request status	:	open

reject:
The hosting provider *views* the **request**. The hosting provider *reviews* the warrant and *rejects* a **target** from the list. When a target is rejected, the target status will be set (automatic information). The target status is one of the statuses defined for the object within the dashboard (rejected).

Action	:	target (reject)
From participant	:	hosting provider
For participant(s)	:	police
Description	:	A target object with information related to the target which can be rejected from a list of related targets.
Mandatory information	:	none
Optional information	:	none
Automatic information	:	target status
Case status	:	accepted
Target status	:	rejected
Request status	:	open

* During development, juridical is changed to legal. This screenshot is taken before this change.

In both cases, the hosting provider can *update* the request with additional case information; the technical comment (optional information). The comment is a summary of the hosting provider's actions or an explanation for the rejected target(s). The technical comment date, technical comment user will be set (automatic information). The technical comment date and user are the date on which the comment is made and the hosting provider employee that made the comment. This action can be performed multiple times.

Action	:	request (update)
From participant	:	hosting provider
For participant(s)	:	police
Description	:	A request object with information related to the case which can be updated.
Mandatory information	:	none
Optional information	:	(case) technical comment
Automatic information	:	(case) technical comment date, technical comment user
Case status	:	accepted
Target status	:	open, accepted, rejected or booted
Request status	:	open

This addresses request management as mentioned in section 3.2.1 and warrant hand over as mentioned in section 3.2.2.

4.11 Live boot target (8)

The hosting provider live boots the related targets, using a live boot image. If the search is within a residence under direction of the examining judge, this action is taken by the police.

The hosting provider *generates* a live boot image for a **target** from the list (beyond the scope of this research). The hosting provider uses this image to boot the target and limits the actions to this task. This ensures the chain of evidence for the process. When a target is booted, the target status will be set (automatic information). The target status is one of the statuses defined for the object within the dashboard (booted).

Action	:	target (boot)
From participant	:	hosting provider (/ police)
For participant(s)	:	police (/ -)
Description	:	A target object with information related to the target which can be set to booted from a list of related targets.
Mandatory information	:	none
Optional information	:	none
Automatic information	:	target status
Case status	:	accepted
Target status	:	booted
Request status	:	open

When all targets in a request are either rejected or booted, the hosting provider *closes* the request. This requires additional case information; the technical comment (mandatory information). The comment is a summary of the hosting provider's actions or an explanation for the rejected target(s). The technical comment date, technical comment user and request status will be set (automatic information). The technical comment date and user are the date on which the comment is made and the hosting provider employee that made the comment. The request status is one of the statuses defined for the object within the dashboard (closed).

Action	:	request (update and close)
From participant	:	hosting provider
For participant(s)	:	police
Description	:	A request object with information related to the case which can be updated and closed.
Mandatory information	:	(case) technical comment
Optional information	:	none
Automatic information	:	(case) technical comment date, technical comment user request status
Case status	:	accepted
Target status	:	rejected or booted
Request status	:	closed

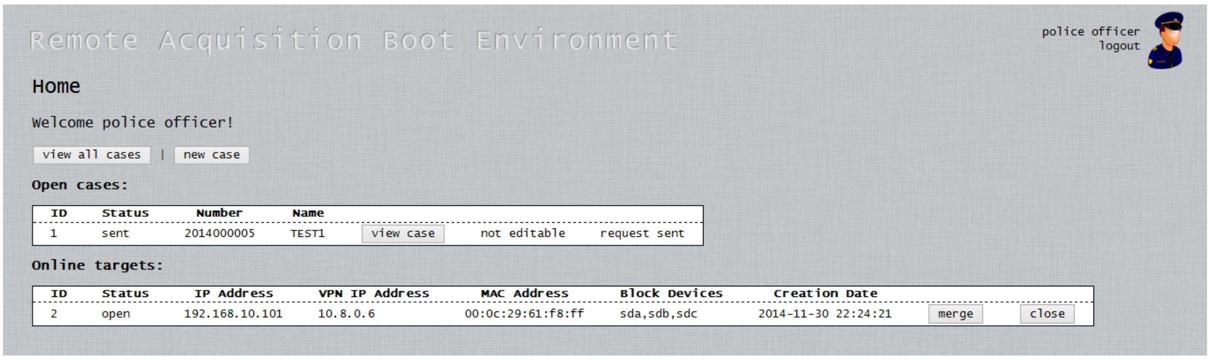
This addresses target and request management as mentioned in section 3.2.1. It also addresses company cooperation, chain of evidence and live boot target as mentioned in section 3.2.3.

4.12 Connect target (9)

A script within the live boot image posts information of the booted target to the dashboard. The target information appears in an online target list for the police. The police **merges** the information with the corresponding **target** in the dashboard. This does not require any information input. It adds additional target information; VPN IP address, MAC address, block devices, disk information and temporary target information (automatic information). The VPN IP and MAC address are the identifying addresses of the target. The block devices are a list of the block devices within the target. The disk information is a summary of the meta information of the storage devices within the target. This information is required to be able to connect to the target, but the responsibility for this information lies with the live boot image. The temporary target status is one of the statuses defined for the object within the dashboard (closed), so it is removed from the online target list.

A preview of the online target list is shown in figure 11 and a preview of merging the target information is shown in figure 12.

Action	:	target (merge and view)
From participant	:	police
For participant(s)	:	-
Description	:	A target object with information related to the target which can be merged and viewed.
Mandatory information	:	none
Optional information	:	none
Automatic information	:	(target) VPN IP address, MAC address, block devices, disk information temporary target status
Case status	:	accepted
Target status	:	booted
Request status	:	open or closed
Temporary target status	:	closed



Remote Acquisition Boot Environment

police officer
logout

Home

Welcome police officer!

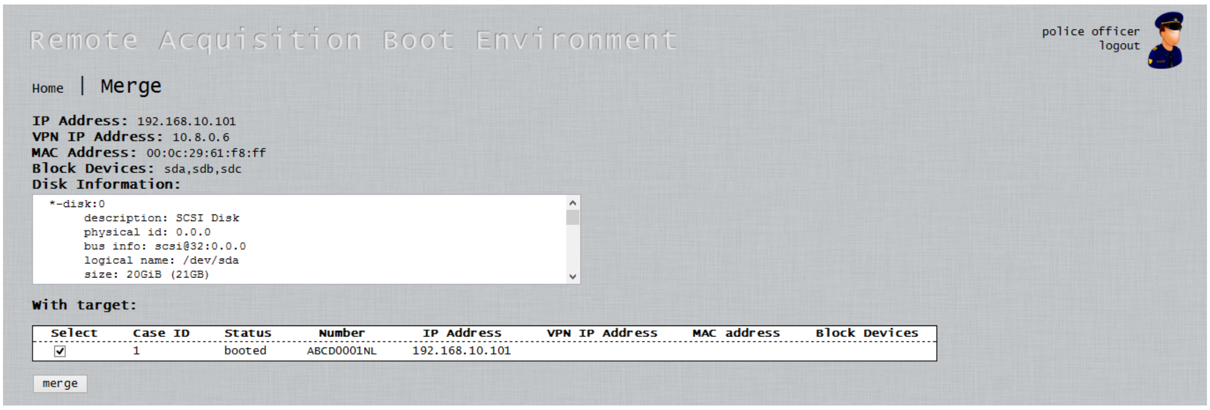
Open cases:

ID	Status	Number	Name			
1	sent	2014000005	TEST1	view case	not editable	request sent

Online targets:

ID	Status	IP Address	VPN IP Address	MAC Address	Block Devices	Creation Date		
2	open	192.168.10.101	10.8.0.6	00:0c:29:61:f8:ff	sda,sdb,sdc	2014-11-30 22:24:21	merge	close

Figure 11: RABE dashboard online targets list (police)



Remote Acquisition Boot Environment

police officer
logout

Home | Merge

IP Address: 192.168.10.101
VPN IP Address: 10.8.0.6
MAC Address: 00:0c:29:61:f8:ff
Block Devices: sda,sdb,sdc
Disk Information:

```
  *-disk:0
       description: SCSI Disk
       physical id: 0.0.0
       bus info: scsi@32:0.0.0
       logical name: /dev/sda
       size: 20GiB (21GB)
```

With target:

Select	Case ID	Status	Number	IP Address	VPN IP Address	MAC address	Block Devices
☑	1	booted	ABCD0001NL	192.168.10.101			

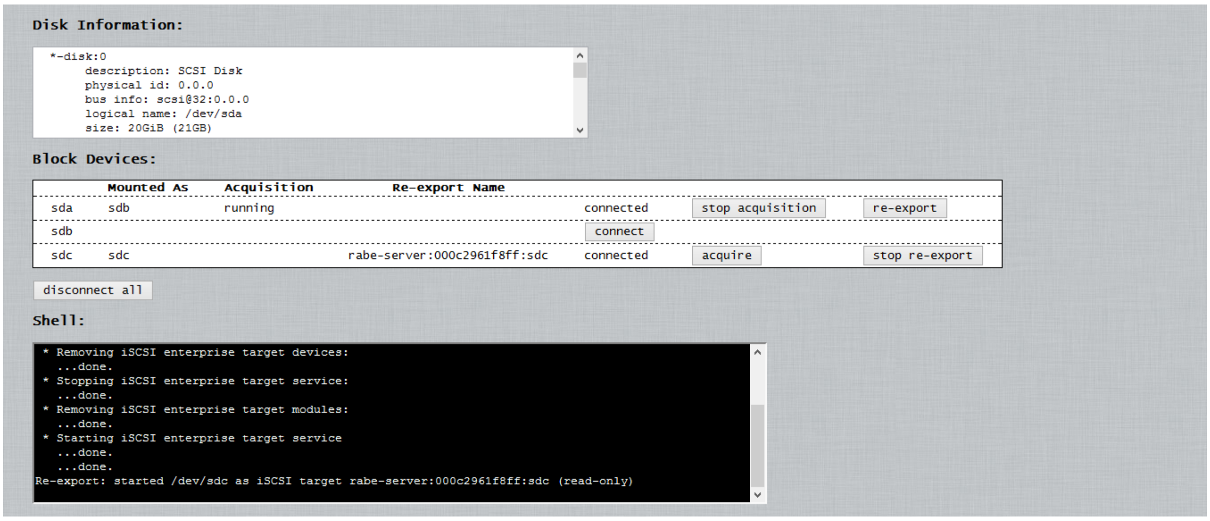
Figure 12: RABE dashboard merge target

The dashboard lists the block devices of a **target**. The police *connects* to a block device through the dashboard. This does not require any information input. It stores the local mount block device for further actions (temporary information). The mount block device is the block device on the system that runs the dashboard on which the target's device is mounted. The shell output of all the target actions is also shown.

This addresses target management as mentioned in section 3.2.1 and connect target as mentioned in section 3.2.3.

A preview of the block device list is shown in figure 13.

Action	:	target (connect)
From participant	:	police
For participant(s)	:	-
Description	:	Connecting to the block device of a physical target.
Mandatory information	:	none
Optional information	:	none
Automatic information	:	none
Temporary information	:	local mount block device
Case status	:	accepted
Target status	:	booted
Request status	:	open or closed



Disk Information:

```
 *-disk:0
      description: SCSI Disk
      physical id: 0.0.0
      bus info: scsi@32:0.0.0
      logical name: /dev/sda
      size: 20GiB (21GB)
```

Block Devices:

	Mounted As	Acquisition	Re-export Name			
sda	sdb	running		connected	stop acquisition	re-export
sdb				connect		
sdc	sdc		rabe-server:000c2961f8ff:sdc	connected	acquire	stop re-export

Shell:

```
 * Removing iSCSI enterprise target devices:
   ...done.
 * Stopping iSCSI enterprise target service:
   ...done.
 * Removing iSCSI enterprise target modules:
   ...done.
 * Starting iSCSI enterprise target service
   ...done.
   ...done.
Re-export: started /dev/sdc as iSCSI target rabe-server:000c2961f8ff:sdc (read-only)
```

Figure 13: RABE dashboard block device list

4.13 Acquire/preview? (10)

The police *decides* whether to perform an acquisition or a live preview, depending on the circumstances of the case. An acquisition provides a full forensic image of a block device and a live preview provides the browsing of a block device.

4.14 Re-export target (11)

The police *re-exports* a block device through the dashboard. This does not require any information input. It stores and shows the export name for further actions (temporary information). The export name is the name used on the system that runs the dashboard on which the target's device is exported.

The re-export can also be stopped. This requires the export name to stop the specific device.

This addresses target management as mentioned in section 3.2.1 and re-export target as mentioned in section 3.2.3.

A preview of re-exporting a block device is previously shown in figure 13 of section 4.12.

Action	:	target (re-export)
From participant	:	police
For participant(s)	:	-
Description	:	Re-exporting the block device of a physical target.
Mandatory information	:	none
Optional information	:	none
Automatic information	:	none
Temporary information	:	export name
Case status	:	accepted
Target status	:	booted
Request status	:	open or closed

Action	:	target (stop re-export)
From participant	:	police
For participant(s)	:	-
Description	:	Stop re-exporting the block device of a physical target.
Mandatory information	:	export name
Optional information	:	none
Automatic information	:	none
Temporary information	:	none
Case status	:	accepted
Target status	:	booted
Request status	:	open or closed

4.15 Acquire target (12)

The police *acquires* a block device through the dashboard. This requires the case and target path and local mount block device (mandatory information). The path is the case and target directory on the system that runs the dashboard for the forensic image and log files. The mount block device is the block device on the local computer on which the target's device is mounted. It stores and shows the acquisition status (temporary information). The status is temporary (running). The full log of the acquisition can be viewed through the dashboard.

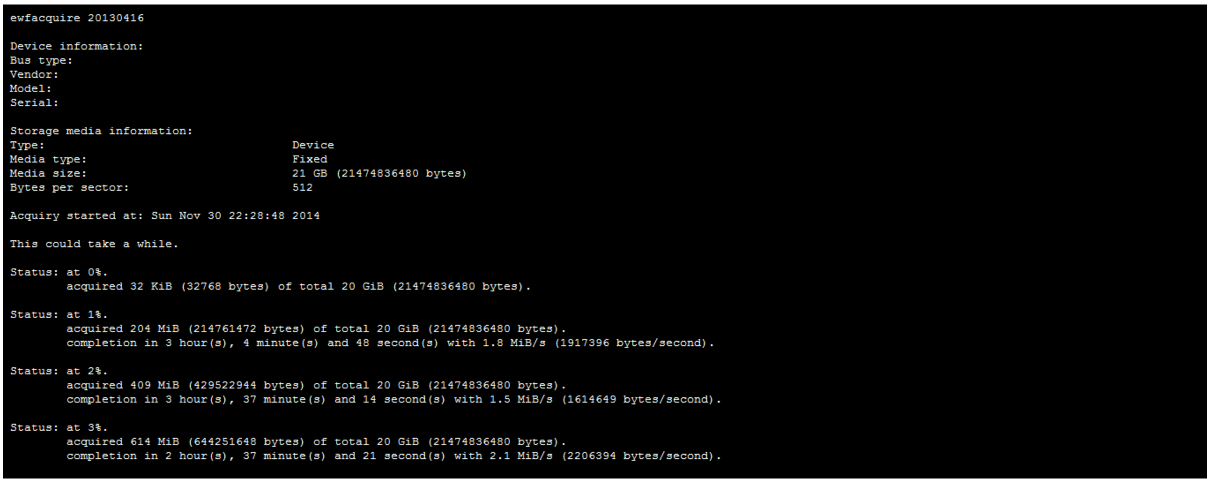
The acquisition can also be stopped. Again, this requires the case and target path and local mount block to stop the specific process. It also stores and shows the acquisition status (temporary information). The status is temporary (stopped).

This addresses target and file management as mentioned in section 3.2.1. It also addresses acquire target as mentioned in section 3.2.3.

A preview of the acquisition log is shown in figure 14.

Action	:	target (acquire)
From participant	:	police
For participant(s)	:	-
Description	:	Acquire the block device of a physical target.
Mandatory information	:	case and target path local mount block device
Optional information	:	none
Automatic information	:	none
Temporary information	:	acquisition status
Case status	:	accepted
Target status	:	booted
Request status	:	open or closed
Temporary status	:	running

Action	:	target (stop acquisition)
From participant	:	police
For participant(s)	:	-
Description	:	Stop acquiring the block device of a physical target.
Mandatory information	:	case path local mount block device
Optional information	:	none
Automatic information	:	none
Temporary information	:	acquisition status
Case status	:	accepted
Target status	:	booted
Request status	:	open or closed
Temporary status	:	stopped



```
ewfacquire 20130416

Device information:
Bus type:
Vendor:
Model:
Serial:

Storage media information:
Type:                                   Device
Media type:                             Fixed
Media size:                             21 GB (21474836480 bytes)
Bytes per sector:                       512

Acquiry started at: Sun Nov 30 22:28:48 2014

This could take a while.

Status: at 0%.
        acquired 32 KiB (32768 bytes) of total 20 GiB (21474836480 bytes).

Status: at 1%.
        acquired 204 MiB (214761472 bytes) of total 20 GiB (21474836480 bytes).
        completion in 3 hour(s), 4 minute(s) and 48 second(s) with 1.8 MiB/s (1917396 bytes/second).

Status: at 2%.
        acquired 409 MiB (429522944 bytes) of total 20 GiB (21474836480 bytes).
        completion in 3 hour(s), 37 minute(s) and 14 second(s) with 1.5 MiB/s (1614649 bytes/second).

Status: at 3%.
        acquired 614 MiB (644251648 bytes) of total 20 GiB (21474836480 bytes).
        completion in 2 hour(s), 37 minute(s) and 21 second(s) with 2.1 MiB/s (2206394 bytes/second).
```

Figure 14: RABE dashboard acquisition log

When all actions are completed, the block devices of the target can be disconnected. This does not require any information input or setting. It disconnects all devices of the target.

Action	:	target (disconnect all)
From participant	:	police
For participant(s)	:	-
Description	:	Disconnecting all connected block devices of a physical target.
Mandatory information	:	none
Optional information	:	none
Automatic information	:	none
Temporary information	:	none
Case status	:	accepted
Target status	:	booted
Request status	:	open or closed

5 Technical design

This section describes the technical design of the concept. The actual implementation of the design is mentioned in appendix B and the testing of the design is mentioned in appendix C.

5.1 Server

The technical design of the server extended the basic server configuration of the previous research. This configuration was altered by deleting NFS functionality and adding web service and database functionality. This resulted in a server side implementation as shown in figure 15.

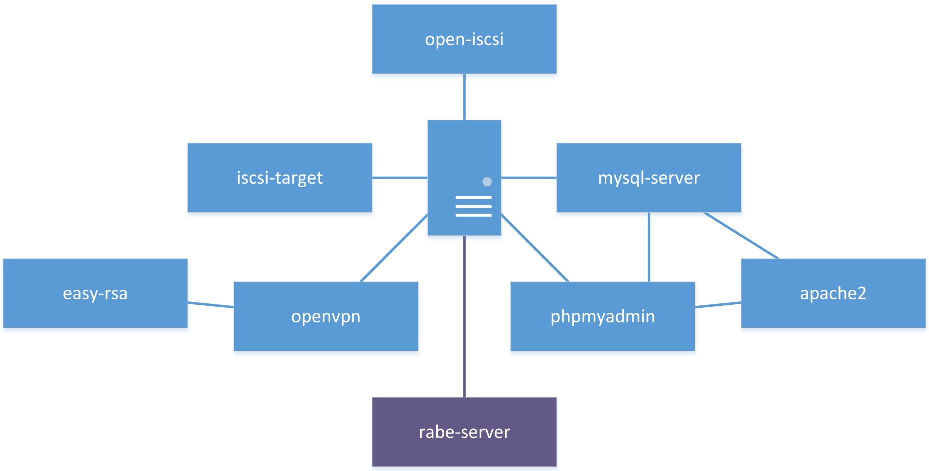


Figure 15: Extended server side implementation

5.1.1 Packages

open-iscsi The `open-iscsi` package is an open source iSCSI initiator program. It includes the `iscsiadm` program for discovering and connecting iSCSI targets. [4] This package is required by the dashboard for connecting to the targets' distributed storage devices.

iscsitarget The `iscsitarget` package is the iSCSI Enterprise Target (IET) program. It is an open source iSCSI target program for distributing storage devices in an enterprise environment. [5] This package is required by the dashboard for re-exporting a target's storage device as an iSCSI target for connecting trough the server to an investigator's computer.

openvpn The `openvpn` package is the OpenVPN program. It is an open source SSL VPN program for creating a routed IP tunnel (TUN) or ethernet tunnel (TAP). The package contains the programs for client and server implementations. [6]. This package is used for starting the server side of the routed VPN tunnel (TUN) to communicate securely with the target.

easy-rsa The `easy-rsa` package is a small RSA key management command line utility to build and manage public key infrastructure certificates. It is part of the OpenVPN project [6], but needs to be installed separately. This package is required by the server configuration for creating server and client certificates.

mysql-server The `mysql-server` package is the MySQL database management system. It is an open source database system for using databases in various environments. MySQL has a wide integration into various programming languages and is one of the most popular database systems around. [7] This package is required by the dashboard for storing structured data like cases and targets.

phpmyadmin The phpmyadmin package is a web-based administration tool to manage MySQL-based databases. It supports a wide range of operations on MySQL, MariaDB and Drizzle. [8] This package is used for managing the MySQL database of the dashboard.

apache2 The apache2 package is the Apache HTTP Server program. It is an open-source web service for offering web pages. The Apache HTTP Server is the most popular web server around. [9] This package is required by phpMyAdmin and the dashboard for the web service.

5.1.2 rabe-server

The rabe-server provides the actual functionality of the dashboard as mentioned in section 4. It consists of two components; a database and web service. The design is schematically shown in figure 16.

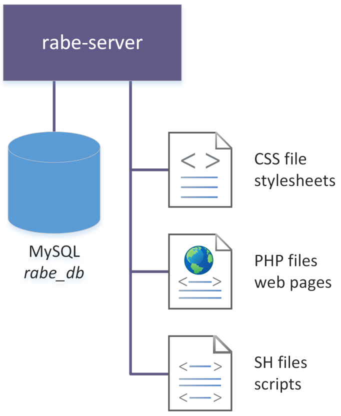


Figure 16: rabe-server (schematic)

The separation of concerns between the different participants is ensured by a login system and separated user group pages.

The dashboard will be an installable Debian package or a full server ISO image named rabe-server. This Debian package and ISO image need to be created with a make file; make_rabe-server_deb.sh and make_rabe-server_iso.sh. The sources and their make files can be found on Remote Acquisition Boot Environment: rabe-framework project on GitHub [10].

5.2 Client

The new server design required the integration of the client from the previous research. The new design provided a simpler and more efficient solution for posting client side information to the server. In the initial client concept this was achieved by the send_client_information service, posting a file with the client information to a NFS share on the server. The steps for this NFS solution are illustrated by steps 1.4, 1.4.1 and 2.6.1 to 2.6.8 as previously shown in figure 2 of section 2.1. This method is replaced by a database solution. The service now posts the client information to the web server which stores it in a temporary database table to be merged with the correct target later on.

Another improvement to the client is the iSCSI export of storage devices in read-only mode. The dashboard required this functionality to make sure no data would be written to the re-exported storage device. The initial client concept was based on the principle of mounting the actual block devices in read-only mode, so this would not be required for the export of the storage devices. However, this principle was not in the scope of the previous research and was not implemented. To ensure a more forensic sound process this option was also added to the client side implementation.

This resulted in an improved RABE client process as shown in figure 17.

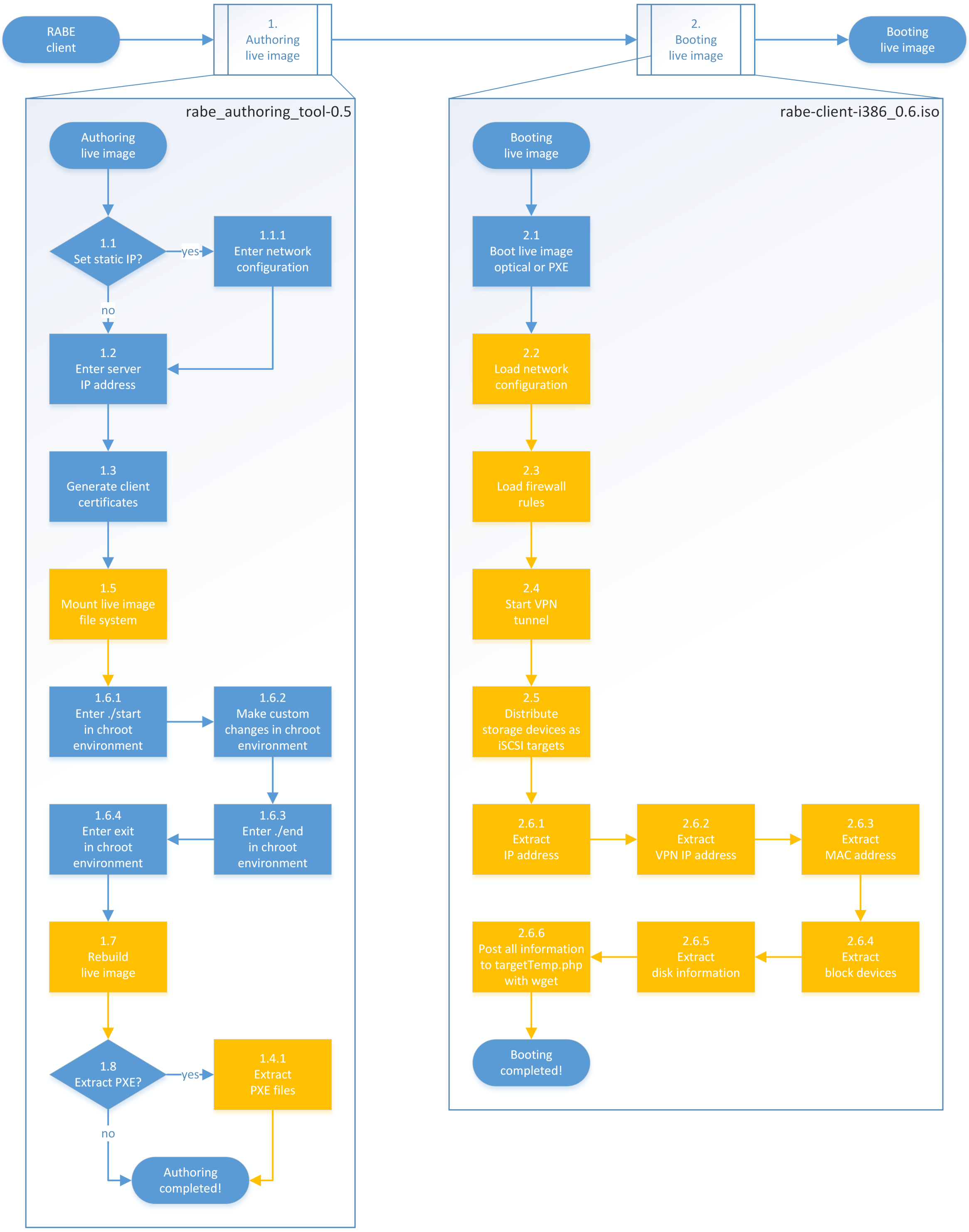

RABE client
1. Authoring live image
2. Booting live image
Booting live image
rabe_authoring_tool-0.5
Authoring live image
1.1 Set static IP?
yes
1.1.1 Enter network configuration
no
1.2 Enter server IP address
1.3 Generate client certificates
1.5 Mount live image file system
1.6.1 Enter ./start in chroot environment
1.6.2 Make custom changes in chroot environment
1.6.3 Enter ./end in chroot environment
1.6.4 Enter exit in chroot environment
1.7 Rebuild live image
1.8 Extract PXE?
yes
1.4.1 Extract PXE files
no
Authoring completed!
rabe-client-i386_0.6.iso
Booting live image
2.1 Boot live image optical or PXE
2.2 Load network configuration
2.3 Load firewall rules
2.4 Start VPN tunnel
2.5 Distribute storage devices as iSCSI targets
2.6.1 Extract IP address
2.6.2 Extract VPN IP address
2.6.3 Extract MAC address
2.6.4 Extract block devices
2.6.5 Extract disk information
2.6.6 Post all information to targetTemp.php with wget
Booting completed!

Figure 17: Improved RABE client flow chart diagram

6 Conclusion

6.1 General

The main research question was:

How can the server side of the RABE proof of concept be visualized into a dashboard from which the administrative, legal and technical steps of the remote acquisition process can be performed?

This research distinguished the administrative, legal and technical steps of a remote acquisition by looking into the participants of the process and discovering what these participants needed to do within the process. It led to a dashboard with separated sections for each of the participants. It visualizes the workflow with a user interface providing all participants the needed information and tasks, including the actual acquisition and re-export of targets. It also offers a strict separation of concerns, because all participants need a shielded environment to ensure a fair trail. The dashboard is build with the technologies: HTML, CSS, Javascript, MySQL and PHP. It is a web-based application that could be used across the internet, but still needs some enhancements, research and testing before being used on the open internet. The server side implementation of the Remote Acquisition Boot Environment (RABE) extends the proof of concept to a fully working concept from client to server with a workflow from an administrative, legal and technical perspective.

However, the Dutch criminal system offers no complete legal framework for a remote acquisition at this moment. This is due the fact of slightly outdated legislation. Two articles provide some aspects of a remote acquisition. Article 125i covers the aspect of *acquiring data stored or recorded on a storage device*. Article 125k addresses the aspect of *obligated cooperation of someone who has access to the system*. There could be a complete legal framework for such an acquisition, if jurisprudence to article 125i would conclude that the search does not have to be performed physically on the scene. And if jurisprudence to article 125k would conclude that the obligation to provide access to the system also applies to the hosting provider booting the target with a live image.

Another thing one really needs to understand is that this proof of concept is not a form of hacking which still is illegal within the Dutch criminal system. In the case of hacking one gets access to a system by breaking its security without the knowledge of any of the participants that provide the system, including owner and hosting provider. This proof of concept is based on the idea of an acquisition that could be performed under article 125i on the scene, but because of capacity and time management could be performed remotely.

In the future this proof of concept could be used to support the tasks of a remote acquisition, but it does need more development and thorough penetration testing. There needs to be more collaboration between the participants that are part of the criminal system (police, public prosecutor and examining judge) to meet requirements on all sides of the process. This research was a first step. The process can be extended with specific requirements for the the public prosecutor or examining judge. The implementation can be extended with more business logic for all sides. Some examples are mentioned in section 6.3. This all could happen rather soon, if used strictly as a supportive system next to the main system of each participant and not as a replacement. It is a system that combines related tasks and shares information in a modern way as needed in this digital era. However, first a legal framework for a remote acquisition is needed.

This concept is an example for the participants and developers of future IT systems for these participants. It shows that the collaboration and interoperability of systems will help in making processes, like a remote acquisition, manpower and time efficient. In this way parts of the concept might be implemented in the IT systems of each participant, to be tied together eventually.

6.2 Achievements

extended the proof of concept This research extended the proof of concept for the remote acquisition of multiple computers by adding a full configuration for the server and including a dashboard in which all steps of a remote acquisition can be performed. The server configuration is installable by a Debian package or a full server ISO image.

provided a visualized workflow This research provided all participants (police, public prosecutor, examining judge and hosting provider) a user interface with the needed information and tasks. It separates the concerns of each participant to ensure a fair trail.

provided a conclusion on the applicability of a remote acquisition in the Dutch criminal system This research provided the conclusion that at this moment the Dutch criminal system does not provide a complete legal framework for a remote acquisition. It has some articles that provide some aspect of it, but unfortunately they are bound to a physical search on the scene and not a virtual one.

improved the client live image This research improved the client live image by replacing the NFS solution of posting client information to a NFS share with a database solution in which the information is posted to a temporary target table in the database. This temporary information is then merged with the actual target by the user.

extended the open framework for future research This research extended the open platform for future research. With the server side implementation and the process' workflow included, it provides the key aspects to perform a real acquisition and several starting points for future research. It will be published on GitHub within the 'rabe-framework' project [10].

6.3 Future research

This report extended the open framework for future research as mentioned in section 6.2. During this research a couple of aspects were discovered that could help to improve the concept on various subjects. The client implementation improvement are omitted and can be found in the report of the previous research [2].

6.3.1 Dashboard

The RABE dashboard in this research is a working concept providing the needed information, but can be improved by the following subjects:

online status The online status of the target computer can not be seen in the dashboard, besides the posting of the client information to the temporary table in the database. This could be implemented by continuously checking the status of the VPN tunnel.

email support The dashboard offers an overview of open requests for the public prosecutor, examining judge and hosting provider. It can not be expected of them that they continuously check the system for updates. An email system that provides status updates to the participants on the status of request would be a useful asset.

business logic The dashboard offers the business logic for the workflow, but could be provided with more business logic that automated status labelling or other tasks.

file integrity check The best practice used for executing `sudo` commands by the dashboard could be a potential security risk if the shell scripts are altered. A file integrity check at login could reduce such a risk. Especially when the actual file checksums are centrally managed.

6.3.2 Penetration testing

The server uses several best practices for ensuring security and the separation of concerns. Although these are implemented to the best efforts, the system needs be tested and possibly improved for the use across the open internet.

6.3.3 Partial acquisition

With a fully working proof of concept for the complete acquisition of a distributed storage device of a target (client), the sparse acquisition with copy-on-read by Eric van den Haak [3] can added.

References

[1] Martin B. Koopmans and Joshua I. James,
Automated network triage,
http://www.sciencedirect.com/science/article/pii/S1742287613000273,
University College Dublin (UCD),
2013,
paper.

[2] Dennis Cortjens,
Bootable Linux CD / PXE for the remote acquisition of multiple computers,
http://www.delaat.net/rp/2013-2014/p70/report.pdf,
University of Amsterdam (UvA),
2014,
report (master thesis).

[3] Eric van den Haak,
Remote data acquisition on block devices in large environments,
http://www.delaat.net/rp/2013-2014/p71/report.pdf,
University of Amsterdam (UvA),
2014,
report (master thesis).

[4] Open-iSCSI,
Open-iSCSI Project,
http://www.open-iscsi.org/,
2005,
website.

[5] SourceForge,
iSCSI Enterprise Target,
http://iscsitarget.sourceforge.net/,
2010,
website.

[6] OpenVPN Technlogies,
OpenVPN Community Software,
https://openvpn.net/index.php/open-source/overview.html,
2014,
website.

[7] MySQL,
MySQL Community Server,
http://dev.mysql.com/downloads/mysql/,
2014,
website.

[8] phpMyAdmin,
phpMyAdmin,
http://www.phpmyadmin.net/home_page/index.php,
2014,
website.

[9] Apache Software Foundation,
HTTP Server Project,
http://httpd.apache.org/,
2014,
website.

[10] GitHub,
Remote Acquisition Boot Environment: rabe-framework,
`https://github.com/rabe-framework`,
2014,
website.

List of Figures

List of Tables

Appendices

A Wetboek van Strafvordering (Dutch Criminal Procedure Code)

Artikel 96c Wetboek van Strafvordering

1.

In geval van ontdekking op heterdaad van een strafbaar feit of in geval van verdenking van een misdrijf als omschreven in artikel 67, eerste lid, kan de officier van justitie ter inbeslagneming elke plaats, met uitzondering van een woning zonder toestemming van de bewoner en een kantoor van een persoon met bevoegdheid tot verschoning als bedoeld in artikel 218, doorzoeken.

2.

Bij dringende noodzakelijkheid en indien het optreden van de officier van justitie niet kan worden afgewacht, kan een hulpofficier deze bevoegdheid uitoefenen. Hij behoeft daartoe de machtiging van de officier van justitie. Indien vanwege de vereiste spoed of de onbereikbaarheid van de officier van justitie de machtiging niet tijdig kan worden gevraagd, kan de machtiging binnen drie dagen na de doorzoeking door de officier van justitie worden verleend. Weigert de officier van justitie de machtiging, dan draagt hij zorg dat de gevolgen van de doorzoeking zoveel mogelijk ongedaan worden gemaakt.

3.

Het doorzoeken van plaatsen overeenkomstig het bepaalde in het eerste lid geschiedt onder leiding van de officier van justitie of, in geval van toepassing van het tweede lid, onder leiding van de hulpofficier.

...

Artikel 110 Wetboek van Strafvordering

1.

De rechter-commissaris kan, op vordering van de officier van justitie en indien hij uit hoofde van de artikelen 181 tot en met 183 onderzoekshandelingen verricht tevens ambtshalve, ter inbeslagneming elke plaats doorzoeken. Hij kan zich daarbij doen vergezellen van bepaalde door hem aangewezen personen. De vordering vermeldt het strafbare feit en indien bekend de naam of anders een zo nauwkeurig mogelijke omschrijving van de verdachte, alsmede de feiten of omstandigheden waaruit blijkt dat de wettelijke voorwaarden voor uitoefening van de bevoegdheid zijn vervuld.

2.

Het doorzoeken van plaatsen overeenkomstig het bepaalde in het eerste lid geschiedt onder leiding van de rechter-commissaris in tegenwoordigheid van de officier van justitie of, in geval van diens verhindering, van een hulpofficier van justitie.

...

Artikel 125i Wetboek van Strafvordering

Aan de rechter-commissaris, de officier van justitie, de hulpofficier van justitie en de opsporingsambtenaar komt onder dezelfde voorwaarden als bedoeld in de artikelen 96b, 96c, eerste, tweede en derde lid, 97, eerste tot en met vierde lid, en 110, eerste en tweede lid, de bevoegdheid toe tot het doorzoeken van een plaats ter vastlegging van gegevens die op deze plaats op een gegevensdrager zijn opgeslagen of vastgelegd. In het belang van het onderzoek kunnen zij deze gegevens vastleggen. De artikelen 96, tweede lid, 98, 99 en 99a zijn van overeenkomstige toepassing.

Artikel 125k Wetboek van Strafvordering

1.

Voor zover het belang van het onderzoek dit bepaaldelijk vordert, kan indien toepassing is gegeven aan artikel 125i of artikel 125j tot degeen van wie redelijkerwijs kan worden vermoed dat hij kennis draagt van de wijze van beveiliging van een geautomatiseerd werk, het bevel worden gericht toegang te verschaffen tot de aanwezige geautomatiseerde werken of delen daarvan. Degeen tot wie het bevel is gericht, dient desgevraagd hieraan gevolg te geven door de kennis omtrent de beveiliging ter beschikking te stellen.

2.

Het eerste lid is van overeenkomstige toepassing indien in een geautomatiseerd werk versleutelde gegevens worden aangetroffen. Het bevel richt zich tot degeen van wie redelijkerwijs kan worden vermoed dat hij kennis draagt van de wijze van versleuteling van deze gegevens.

3.

Het bevel, bedoeld in het eerste lid, wordt niet gegeven aan de verdachte. Artikel 96a, derde lid, is van overeenkomstige toepassing.

B Implementation

B.1 Server

B.1.1 Packages

The following packages required additional configuration settings for the dashboard to function correctly.

iscsitarget The service needs to be enabled by setting `ISCSITARGET_ENABLE=true` in `/etc/default/iscsitarget` to start at boot. The iSCSI targets need to be set in the configuration file located at `/etc/iet/ietd.conf` (this is performed by the dashboard automatically as mentioned in section B.1.3).

openvpn The VPN connection needs to be enabled by setting `IAUTOSTART="<name-of-vpn-connection>"` in `/etc/default/openvpn` to start at boot.

apache2 The service is started automatically and serves web pages from the default location: `/var/www/html/`. For the dashboard to be used on the open internet it requires the use of HTTPS, but this is beyond the scope of this research.

B.1.2 Database

The database for the dashboard is a MySQL database named `rabe_db`. It is based on the database diagram as shown in figure 18.

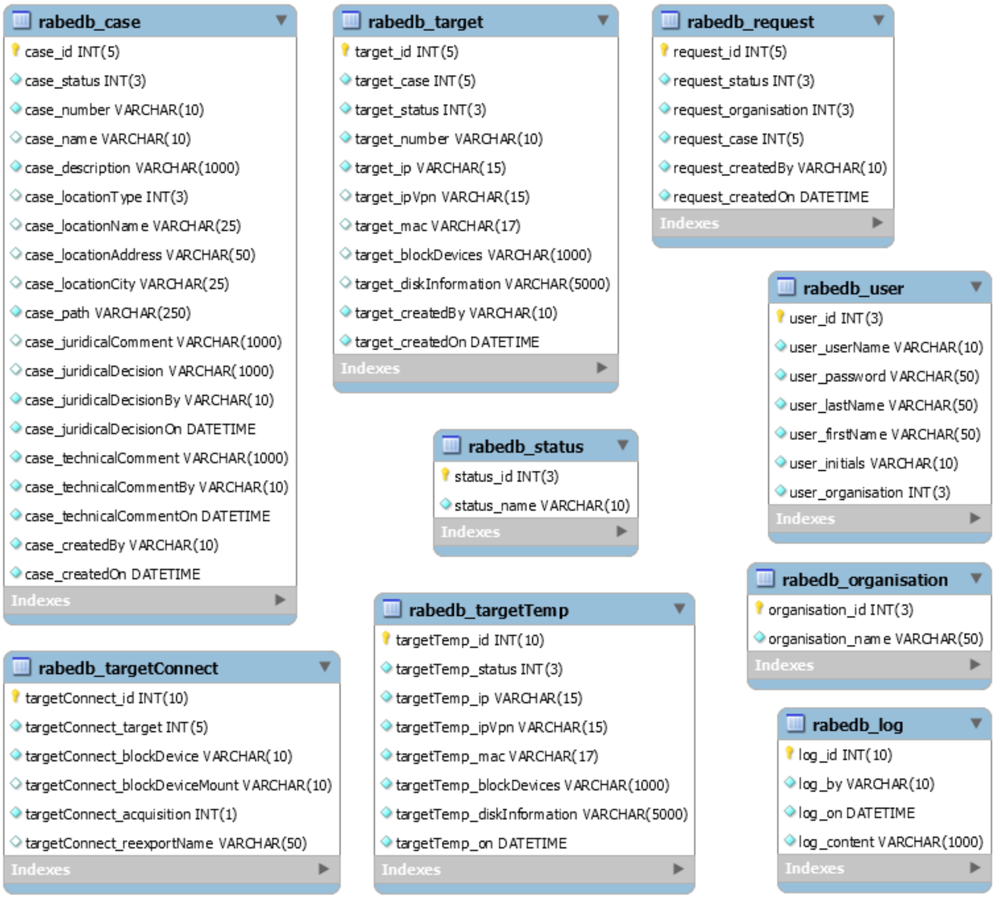

rabedb_case
case_id INT(5)
case_status INT(3)
case_number VARCHAR(10)
case_name VARCHAR(10)
case_description VARCHAR(1000)
case_locationType INT(3)
case_locationName VARCHAR(25)
case_locationAddress VARCHAR(50)
case_locationCity VARCHAR(25)
case_path VARCHAR(250)
case_juridicalComment VARCHAR(1000)
case_juridicalDecision VARCHAR(1000)
case_juridicalDecisionBy VARCHAR(10)
case_juridicalDecisionOn DATETIME
case_technicalComment VARCHAR(1000)
case_technicalCommentBy VARCHAR(10)
case_technicalCommentOn DATETIME
case_createdBy VARCHAR(10)
case_createdOn DATETIME
Indexes
rabedb_target
target_id INT(5)
target_case INT(5)
target_status INT(3)
target_number VARCHAR(10)
target_ip VARCHAR(15)
target_ipVpn VARCHAR(15)
target_mac VARCHAR(17)
target_blockDevices VARCHAR(1000)
target_diskInformation VARCHAR(5000)
target_createdBy VARCHAR(10)
target_createdOn DATETIME
Indexes
rabedb_request
request_id INT(5)
request_status INT(3)
request_organisation INT(3)
request_case INT(5)
request_createdBy VARCHAR(10)
request_createdOn DATETIME
Indexes
rabedb_user
user_id INT(3)
user_userName VARCHAR(10)
user_password VARCHAR(50)
user_lastName VARCHAR(50)
user_firstName VARCHAR(50)
user_initials VARCHAR(10)
user_organisation INT(3)
Indexes
rabedb_status
status_id INT(3)
status_name VARCHAR(10)
Indexes
rabedb_organisation
organisation_id INT(3)
organisation_name VARCHAR(50)
Indexes
rabedb_targetConnect
targetConnect_id INT(10)
targetConnect_target INT(5)
targetConnect_blockDevice VARCHAR(10)
targetConnect_blockDeviceMount VARCHAR(10)
targetConnect_acquisition INT(1)
targetConnect_reexportName VARCHAR(50)
Indexes
rabedb_targetTemp
targetTemp_id INT(10)
targetTemp_status INT(3)
targetTemp_ip VARCHAR(15)
targetTemp_ipVpn VARCHAR(15)
targetTemp_mac VARCHAR(17)
targetTemp_blockDevices VARCHAR(1000)
targetTemp_diskInformation VARCHAR(5000)
targetTemp_on DATETIME
Indexes
rabedb_log
log_id INT(10)
log_by VARCHAR(10)
log_on DATETIME
log_content VARCHAR(1000)
Indexes

Figure 18: Database diagram

rabedb_case The rabedb_case table holds the following information of a case:

- ID
- Status
- Number
- Name
- Description
- Location type
- Location name
- Location address
- Location city
- Directory path
- Legal comment
- Legal decision
- User that made the legal decision
- Date the legal decision was made
- Technical comment
- User that made the technical comment
- Date the last technical comment was made
- User that created the case
- Date the case was created

This table provides the storage of this information for case management as mentioned and required in section 3.2.1.

rabedb_log The rabedb_log table holds all INSERT and UPDATE SQL queries that are executed within the dashboard. It also stores all login SELECT queries to look into all login attempts for security purposes. In this way it is possible to look into all important actions within the dashboard.

rabedb_organisation This is a relational table for translating the organisation ID to the full organisation name in the case of a JOIN SQL statement.

rabedb_request The rabedb_request table holds the information of the request. It includes legal and technical requests. This table provides the storage of this information for request management as mentioned and required in section 3.2.1.

rabedb_status This is a relational table for translating the status ID to the full status name in the case of a JOIN SQL statement. It holds the following statuses:

1. open
2. closed
3. sent
4. accepted
5. rejected
6. booted
7. acquired

These are the same as in the flow chart diagram and steps as previously shown in figure 3 and described in table 1 of section 3.2. This table provides in the status labelling as mentioned and required in section 3.2.1.

rabedb_target The `rabedb_target` table holds the following information of a target:

- ID
- Reference to the case
- Status
- Number
- IP address
- VPN IP address
- MAC address
- Block devices
- Disk information
- User that created the target
- Date the target was created

This table provides the storage of this information for target management as mentioned and required in section 3.2.1.

rabedb_targetConnect This is a temporary table for the actions that can performed with the target's distributed storage devices. It holds the following information:

- ID
- Reference to the target
- Block device
- Mount block device on the server
- Acquisition status
- Re-export name

rabedb_targetTemp This is a temporary table for all targets that are booted. At boot a target will post the following information about the target to the dashboard:

- ID
- Status
- IP address
- VPN IP address
- MAC address
- Block device
- Disk information
- Date the information was submitted

This information is manually merged with the actual target registered with the case.

rabedb_user The `rabedb_user` table holds all users of the dashboard. It includes personal and authorization information like first and last name, username, password and organisation. These last are important for the separation of concerns between the users of the dashboard.

B.1.3 Web server

The web server is the Apache HTTP Server. It consists of stylesheets, pages and scripts served from the default directory as shown in table 2.

File	Description
access.php	The redirection page for unauthorized access.
acquire_log.php	The page for showing the shell output of the acquisition.
acquire.php	The page for acquiring a target's distributed storage device.
all_cases.php	The page for viewing all cases.
all_requests_2.php	The page for viewing all requests for the public prosecutor user group.
all_requests_3.php	The page for viewing all requests for the judiciary user group.
all_requests_4.php	The page for viewing all requests for the hosting provider user group.
case.php	The page for viewing a case.
connect.php	The page for connecting to a target's storage device.
disconnect.php	The page for disconnecting all target's storage devices.
edit_case.php	The page for editing a case.
edit_target.php	The page for editing a target.
file_upload.php	The page for uploading files (warrant).
home_1.php	The home page for the police user group.
home_2.php	The home page for the public prosecutor user group.
home_3.php	The home page for the judiciary user group.
home_4.php	The home page for the hosting provider user group.
index.php	The first and redirection to the login page.
live_image_download.php	The page for downloading the default live image.
login.php	The login page.
logout.php	The logout page.
merge.php	The page for merging temporary target information with an actual target.
mkdir_case.php	The page for creating the case directory structure.
mkdir_target.php	The page for creating the target directory structure.
new_case.php	The page for creating a new case.
new_target.php	The page for creating a new target.
pictures	The folder with all the pictures for the pages.
rabe.css	The stylesheet file for the RABE GUI.
re-export.php	The page for re-exporting a target's storage device.
request_2.php	The page for viewing a request for the public prosecutor user group.
request_3.php	The page for viewing a request for the judiciary user group.
request_4.php	The page for viewing a request for the hosting provider user group.
request_forward.php	The page for forwarding a request.
request_legal.php	The page for sending a legal request.
request_technical.php	The page for sending a technical request.
set_accepted.php	The page for setting the 'accepted' status to a target.
set_booted.php	The page for setting the 'booted' status to a target.
set_closed.php	The page for setting the 'closed' status to a request.
set_rejected.php	The page for setting the 'rejected' status to a target.
shell.css	The stylesheet file for the shell GUI.
stop_acquire.php	The page for stopping the acquisition.
stop_re-export.php	The page for stopping the re-export.
target.php	The page for viewing a target.
targetTemp.php	The page for posting the temporary target information to.
warrant_download.php	The page for downloading the warrant.

Table 2: Web server files

As an additional layer of security all pages are provided with a PHP header that checks the authorization for that page with the logged in user. This header is shown in figure 19.

```
if ($_SESSION['userOrganisation'] != "1") {
        header("Location: access.php");
}
```

Figure 19: PHP authorization header

mkdir_case.php The `mkdir_case.php` page creates the case directory structure by executing a shell script as shown in figure 20. It passes the case path as an argument to the script. The page only receives the case ID as a PHP variable and fetches the case's path from the database, so the used path can not be altered by changing the variable in the URL.

This page and all PHP pages that execute shell scripts use `sudo` rights. This is required by the system or tools. Using the `sudo` command with PHP could be a severe security risk. Therefore the dashboard uses a best practice for using this command in a secure way. All scripts can be executed by the `www-data` (Apache HTTP Server) user with `sudo` rights by adding these to the `\etc\sudoers` file with a `NOPASSWORD` option. The script files can't be changed by the Apache HTTP Server user from a web perspective, because the scripts are not writeable and are owned by the `root` user and group. Trying to change them from the web will result in a permission denied message.

```
...
$shellCommand = shell_exec("sudo ./rabe-server_mkdir_case.sh ".$row['case_path']);
...
```

Figure 20: PHP mkdir_case (snippet)

rabe-server_mkdir_case.sh The associated shell script creates the actual case directory structure as shown in figure 21. This includes the case's root path and a `meta` folder for the warrant file. The script also changes the owner of the `meta` folder to the `www-data` (Apache HTTP Server) user, so the warrant file can be uploaded by the web server. The full directory structure of a case is shown in figure 22. The script only includes the case part of the structure.

```
#!/bin/bash

mkdir -p $1/meta/
chown www-data $1/meta/
```

Figure 21: SH rabe-server_mkdir_case

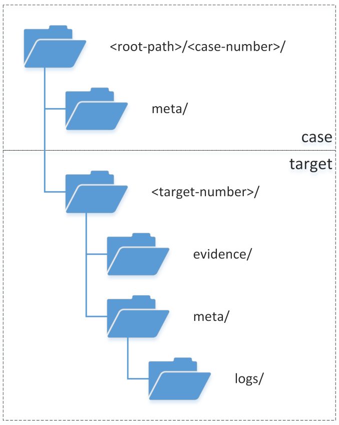


Figure 22: Case directory structure

mkdir_target.php The `mkdir_target.php` page creates the target directory structure within the case directory by executing a shell script as shown in figure 23. It passes the case's path and target number as arguments to the script. Again, the page only receives the case and target ID as a PHP variable and fetches the case's path from the database, so the used path can not be altered by changing the variable in the URL.

```
...
$shellCommand = shell_exec("sudo ./rabe-server_mkdir_target.sh ".$row_c['case_path']." "
    .$row_t['target_number']);
...
```

Figure 23: PHP mkdir_target (snippet)

rabe-server_mkdir_target.sh The associated shell script creates the actual target directory structure as shown in figure 24. This includes the target's root path, `evidence` folder for the acquisition and `meta/logs` folder for the log files. It also creates the shell log file and changes the owner of the file to the `www-data` (Apache HTTP Server) user, so the file can be written by the web server. The full directory structure of a case is previously shown in figure 22. The script only includes the target part of the structure.

```
#!/bin/bash

mkdir -p $1/$2/evidence
mkdir -p $1/$2/meta/logs
touch $1/$2/meta/logs/shell.log
chown www-data $1/$2/meta/logs/shell.log
```

Figure 24: SH rabe-server_mkdir_target

connect.php The `connect.php` page connects to a target's distributed storage device by executing a shell script and another shell command to fetch the new block device as shown in figure 25. It passes the target's VPN IP address, iSCSI target ID (by composing it from the MAC address and block device) and target path (by composing it from the case path and target number) as arguments to the script. Again, the page only receives the target connect ID as a PHP variable and fetches the rest of the information from the database. Then it fetches the new block device on which the iSCSI target is mounted from the `dmesg` command and updates the temporary target connect table (`rabedb_targetConnect`) with this information.

```
...
$shellCommand = shell_exec("sudo ./rabe-server_connect.sh ".$row_t['target_ipVpn']." ".
    str_replace(':', '', strtolower($row_t['target_mac'])).":".$row_tc['
    targetConnect_blockDevice']." ".$row_c['case_path'].$row_t['target_number']."");
sleep(1);
$shellCommand = shell_exec("dmesg | tail | grep 'Mode Sense: 77 00' | sed 's
    /.*\[\([^]]*\)\].*/\\1/g'");
$shellCommand = substr($shellCommand,0,3);

$query = "UPDATE rabedb_targetConnect SET targetConnect_blockDeviceMount = '".
    $shellCommand."' WHERE targetConnect_id = '".$row_tc['targetConnect_id']."';";
...
```

Figure 25: PHP connect (snippet)

rabe-server_connect.sh The associated shell script uses the `iscsiadm` program to actually connect the target's storage device as shown in figure 26. It uses the VPN IP address as the portal address for secure communication with the target (client), the iSCSI target ID as the targetname and the target's shell log file path to send the output to.

```
#!/bin/bash

iscsiadm --mode=node --portal=$1 --targetname=$2 --login >> $3/meta/logs/shell.log 2>&1
```

Figure 26: SH rabe-server_connect

disconnect.php The `disconnect.php` page disconnects all the target's distributed storage devices by executing a shell script as shown in figure 27. It passes the target's VPN IP address and target path (by composing it from the case path and target number) as arguments to the script. Again, the page only receives the target connect ID as a PHP variable and fetches the rest of the information from the database. Then it updates the temporary target connect table (`rabedb_targetConnect`) by deleting the mounted block device information.

```
...
$shellCommand = shell_exec("sudo ./rabe-server_disconnect.sh ".$row_t['target_ipVpn']."
    ".$row_c['case_path'].$row_t['target_number']."");

$query = "UPDATE rabedb_targetConnect SET targetConnect_blockDeviceMount = '',
    targetConnect_acquisition = '', targetConnect_reexportName = '' WHERE
    targetConnect_target = '".$row_t['target_id']."';";
...
```

Figure 27: PHP disconnect (snippet)

rabe-server_disconnect.sh The associated shell script uses the `iscsiadm` program to actually disconnect all of the target's storage devices as shown in figure 26. It uses the VPN IP address as the portal address and the target's shell log file path to send the output to.

```
#!/bin/bash

iscsiadm --mode=node --portal=$1 --logout all >> $2/meta/logs/shell.log 2>&1
```

Figure 28: SH rabe-server_disconnect

acquire.php The `acquire.php` page acquires a target's distributed storage device by executing a shell script as shown in figure 29. It passes the user's name (from the session), case ID and path, target number, target IP and MAC address, target's and host's block device as arguments to the script. Again, the page only receives the target connect ID as a PHP variable and fetches the rest of the information from the database. Then it updates the temporary target connect table (`rabedb_targetConnect`) by setting the acquisition status.

```
...
$shellCommand = shell_exec("sudo ./rabe-server_acquire.sh ".$_SESSION['user']." ".$row_c
    ['case_id']." ".$row_c['case_path']." ".$row_t['target_number']." ".$row_t['
    target_ip']." ".str_replace(':', '', strtolower($row_t['target_mac']))." ".$row_tc['
    targetConnect_blockDevice']." ".$row_tc['targetConnect_blockDeviceMount']);

$query = "UPDATE rabedb_targetConnect SET targetConnect_acquisition = '1' WHERE
    targetConnect_id = '".$row_tc['targetConnect_id']."';";
...
```

Figure 29: PHP acquire (snippet)

rabe-server_acquire.sh The associated shell script uses the `ewfacquire` program to actually acquire the target's storage device as shown in figure 26. It sets the case properties of the acquisition files by using the passed argument like user's name, case ID, target IP and MAC address. It combines the case path and target number for the target's `evidence` and `logs` folders. The `evidence` folder is used to store the acquisition files and the `logs` folder is used to store the acquisition error log file. It also sends the full shell output of the acquisition to a separate log file which is showed by the `acquire_log.php` page. At last it sends a confirmation line to the target's shell log file.

```
#!/bin/bash

ewfacquire -C $3 -d sha1 -D "$8 of $6" -e "$1 $2" -E $5 -l $4/$5/meta/logs/
    acquisition_error.log -N "iSCSI $7:$8" -S "4.5 GiB" -t $4/$5/evidence/$8 -u /dev/$9
    > $4/$5/meta/logs/acquisition.log 2>&1 &

echo "Acquire: started $7:$8 of $6" >> $4/$5/meta/logs/shell.log
```

Figure 30: SH rabe-server_acquire

stop_acquire.php The `stop\_acquire.php` page stops the acquisition of the target's distributed storage device by executing a shell script as shown in figure 31. It passes the same arguments to the script as `acquire.php`. Again, the page only receives the target connect ID as a PHP variable and fetches the rest of the information from the database. Then it updates the temporary target connect table (`rabedb_targetConnect`) by setting the acquisition status.

```
...
$shellCommand = shell_exec("sudo ./rabe-server_stop_acquire.sh ".$_SESSION['user']." ".
    $row_c['case_id']." ".$row_c['case_path']." ".$row_t['target_number']." ".$row_t['
    target_ip']." ".str_replace(':', '', strtolower($row_t['target_mac']))." ".$row_tc['
    targetConnect_blockDevice']." ".$row_tc['targetConnect_blockDeviceMount']);

$query = "UPDATE rabedb_targetConnect SET targetConnect_acquisition = '3' WHERE
    targetConnect_id = '".$row_tc['targetConnect_id']."';";
...
```

Figure 31: PHP stop_acquire (snippet)

rabe-server_stop_acquire.sh The associated shell script uses the ps and grep commands to find the PID of the acquisition as shown in figure 32. This is achieved by using grep to search for a process with the exact arguments of the acquisition (line 3). It then kills that process with an interrupt (line 5). This is a better way of terminating a process, because it allows the ewfacquire program to abort the acquisition and add this to its log. A direct kill would have terminated the process immediately. At last it sends a confirmation line to the target's shell log file (line 7).

```
1 #!/bin/bash
2
3 PID=$(ps -aux | grep "ewfacquire -C $3 -d sha1 -D $8 of $6 -e $1 $2 -E $5 -l $4/$5/meta/
      logs/acquisition_error.log -N iSCSI $7:$8 -S 4.5 GiB -t $4/$5/evidence/$8 -u /dev/$9
      " | head -1 | awk '{print $2}')
4
5 kill -INT $PID >> $4/$5/meta/logs/shell.log
6
7 echo "Acquire: stopped $7:$8 of $6" >> $4/$5/meta/logs/shell.log
```

Figure 32: SH rabe-server_stop_acquire

re-export.php The re-export.php page re-exports the target's distributed storage device as an iSCSI target by executing a shell script as shown in figure 33. It passes the target's iSCSI target ID (by composing it from the MAC address and block device), host's block device and target path (by composing it from the case path and target number) as arguments to the script. Again, the page only receives the target connect ID as a PHP variable and fetches the rest of the information from the database. Then it updates the temporary target connect table (rabedb_targetConnect) by setting the re-export name.

```
...
$shellCommand = shell_exec("sudo ./rabe-server_re-export.sh ".str_replace(':', '',
    strtolower($row_t['target_mac'])).":".$row_tc['targetConnect_blockDevice']." ".
    $row_tc['targetConnect_blockDeviceMount']." ".$row_c['case_path'].$row_t['
    target_number']."");

$query = "UPDATE rabedb_targetConnect SET targetConnect_reexportName = 'rabe-server:".
    str_replace(':', '', strtolower($row_t['target_mac'])).":".$row_tc['
    targetConnect_blockDevice']."' WHERE targetConnect_id = '".$row_tc['targetConnect_id
    ']."';";
...
```

Figure 33: PHP re-export (snippet)

rabe-server_re-export.sh The associated shell script uses the cat and grep commands to look in iscsitarget configuration file (/etc/iet/ietd.conf) for other iSCSI targets on the host system (line 3 of figure 34). If there are no other targets the counter for adding LUNs is set to 0 (line 5-6). If there are targets it searches for the last target (line 8) and increases its LUN by 1 (line 9). Then it adds the new read-only iSCSI target by adding the configuration line with the passed iSCSI target ID of the target's storage devices with a rabe-server prefix (line 12-13). The iscsitarget service needs to be restarted to start the new iSCSI target (line 15). The output is send to the target's shell log file, as well as a confirmation line (line 17).

```
#!/bin/bash

TARGET=$(cat /etc/iet/ietd.conf | grep '^Target')

if [[ -z $TARGET ]]; then
  COUNTER=0
else
  LUN=$(tail -1 /etc/iet/ietd.conf | awk '{print $2}')
  COUNTER=$[ $LUN + 1 ]
fi

echo "Target rabe-server:$1" >> /etc/iet/ietd.conf
echo "Lun $COUNTER Path=/dev/$2,Type=fileio,IOMode=ro" >> /etc/iet/ietd.conf

service iscsitarget restart >> $3/meta/logs/shell.log

echo "Re-export: started /dev/$2 as iSCSI target rabe-server:$1 (read-only)" >> $3/meta/
    logs/shell.log
```

Figure 34: SH rabe-server_re-export

stop_re-export.php The `re-export.php` page stops the re-exports of the target's distributed storage device by executing a shell script as shown in figure 35. It passes the same arguments to the script as `re-export.php`. Again, the page only receives the target connect ID as a PHP variable and fetches the rest of the information from the database. Then it updates the temporary target connect table (`rabedb_targetConnect`) by deleting the re-export name.

```
...
$shellCommand = shell_exec("sudo ./rabe-server_stop_re-export.sh ".str_replace(':', '',
    strtolower($row_t['target_mac'])).":".$row_tc['targetConnect_blockDevice']." ".
    $row_tc['targetConnect_blockDeviceMount']." ".$row_c['case_path'].$row_t['
    target_number']."");

$query = "UPDATE rabedb_targetConnect SET targetConnect_reexportName = '' WHERE
    targetConnect_id = '".$row_tc['targetConnect_id']."';";
...
```

Figure 35: PHP stop_re-export (snippet)

rabe-server_stop_re-export.sh The associated shell script uses the `sed` command to look in the `iscsitarget` configuration file for iSCSI targets with the same properties as the passed argument and deletes the associated lines (line 3 and 4 of figure 36). Then it restarts the `iscsitarget` service to stop the iSCSI target (line 6). The output is send to the target's shell log file, as well as a confirmation line (line 8).

```
1 #!/bin/bash
2
3 sed -i "/rabe-server:$1/d" /etc/iet/ietd.conf
4 sed -i "/Path=\/dev\/$2/d" /etc/iet/ietd.conf
5
6 service iscsitarget restart >> $3/meta/logs/shell.log
7
8 echo "Re-export: stopped /dev/$2 as iSCSI target rabe-server:$1" >> $3/meta/logs/shell.
      log
```

Figure 36: SH rabe-server_stop_re-export

targetTemp.php The `targetTemp.php` page posts the temporary target information of targets (clients) to the the databases as shown in figure 37. It requires the IP and VPN IP address, MAC address, block devices and disk information from the target, so it can be merged by a user with the correct target in the database. Such a page is very vulnerable to Denial of Service (DoS) attacks. Therefore a security measure is implemented which does not allow more then three records with the status 'open' from the same IP address.

```
...
if ((int)$row['COUNT(targetTemp_ip)'] <= 3) {
        $ip=$_GET['ip'];
        $ipVpn=$_GET['ipVpn'];
        $mac=$_GET['mac'];
        $blockDevices=$_GET['blockDevices'];
        $diskInfo=$_GET['diskInfo'];

        $query = "INSERT INTO rabedb_targetTemp (targetTemp_ip, targetTemp_ipVpn,
            targetTemp_mac, targetTemp_blockDevices, targetTemp_diskInformation,
            targetTemp_on) VALUES ('".$ip."', '".$ipVpn."', '".$mac."', '".$blockDevices
            ."', '".$diskInfo."', NOW());";
...
```

Figure 37: PHP targetTemp (snippet)

B.2 Client

B.2.1 Services

The improvements on the live image resulted in an improved client side implementation with the following changed services as shown in figure 38.

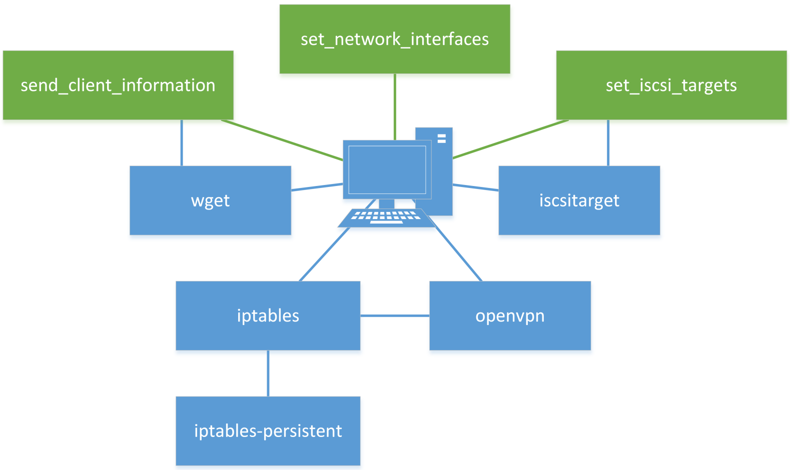


Figure 38: Improved client side implementation

send_client_information The improved `send_client_information` service extracts the IP address of the `eth0` interface (line 13 of figure 39), the VPN IP address of the `tun` interface (line 16) and the MAC address of the `eth0` interface (line 19), all with the `ifconfig` command. It extracts the block devices from the `ls` command, adds each device with an ending comma to a string (line 26-30) and strips the last comma from that string (line 33). Then it extracts the disk (meta) information from the `lshw` command (line 38). All information is then posted by the `wget` program to the `targetTemp.php` page on the web server (line 43). A simpler and more efficient solution then the initial one.

```
#!/bin/bash

# SERVICE: send_client_information
# USAGE: start

case "$1" in
start)
  echo \[send_client_information\] starting service

  echo \[send_client_information\] gathering network information \(ifconfig\)

  # extract IP address (eth0) from ifconfig
  IP=$(ifconfig eth0 | grep -i 'inet addr' | awk '{print $2}' | sed 's/addr://g')

  # wait 5 seconds for the VPN tunnel (tun0) to come up
  sleep 5

  # extract VPN IP address (tun0) from ifconfig
  IPVPN=$(ifconfig tun0 | grep -i 'inet addr' | awk '{print $2}' | sed 's/addr://g')

  # extract MAC address (eth0) from ifconfig
        MAC=$(ifconfig eth0 | grep -i 'hwaddr' | awk '{print $5}')

  echo \[send_client_information\] gathering block devices \(ls\)

  BLOCKDEVICES=""

  # extract block devices from ls
        for SCSI in $(ls /dev/sd* | grep 'sd[a-z]$')
        do
                DISK=$(echo $SCSI | sed 's/\/dev\///g')
    BLOCKDEVICES="$BLOCKDEVICES$DISK,"
        done

  # remove last character
  BLOCKDEVICES=$(echo $BLOCKDEVICES | sed s'/.$//')

  echo \[send_client_information\] gathering disk information \(lshw\)

  # extract disk information from lshw
  DISKINFO=$(lshw -c disk)

  echo \[send_client_information\] POSTing to server \(wget\)

  # POST gathered information with wget
  wget "http://192.168.10.100/targetTemp.php?ip=$IP&ipVpn=$IPVPN&mac=$MAC&blockDevices=
      $BLOCKDEVICES&diskInfo=$DISKINFO"
;;

*)
echo SERVICE: send_client_information
echo $"USAGE: $0 {start}"
exit 1

esac
exit 0
```

Figure 39: Improved send_client_information service (code)

set_iscsi_targets The improved `set_iscsi_targets` service extracts the MAC address of the `eth0` interface with the `ifconfig` command (line 11 of figure 40). It searches for SCSI block devices (sdX) on the client (line 19) and strips the actual block device from the string (line 21). Then for each device it configures an iSCSI target in read-only mode, using the MAC address and block device as the addressing name (lines 26-27). A more forensically sound solution then the initial one.

```
 1  #!/bin/bash
 2
 3  # SERVICE: set_iscsi_targets
 4  # USAGE: start
 5
 6  case "$1" in
 7  start)
 8    echo \[set_iscsi_targets\] starting service
 9
10    # extract MAC address (eth0) from ifconfig
11    MAC=$(ifconfig eth0 | grep -i 'hwaddr' | awk '{print $5}' | sed 's/://g')
12
13    # set counter
14    COUNTER=0
15
16    echo \[set_iscsi_targets\] searching SCSI storage devices
17
18    # search SCSI storage devices
19    for SCSI in $(ls /dev/sd* | grep 'sd[a-z]$')
20    do
21      DISK=$(echo $SCSI | sed 's/\/dev\///g')
22
23      echo \[set_iscsi_targets\] setting iSCSI target $MAC:$DISK
24
25      # add target to configuration file
26      echo Target $MAC:$DISK >> /etc/iet/ietd.conf
27      echo Lun $COUNTER Path=$SCSI,Type=fileio,IOMode=ro >> /etc/iet/ietd.conf
28
29      # increase counter
30      COUNTER=$[ COUNTER+1 ]
31    done
32  ;;
33
34  *)
35  echo SERVICE: set_iscsi_targets
36  echo $"USAGE: $0 {start}"
37  exit 1
38
39  esac
40  exit 0
```

Figure 40: Improved set_iscsi_targets service (code)

C Testing

C.1 Environment

The test environment consisted of a small network with four component; two computers (desktop and notebook) for using the dashboard, the RABE server and a RABE target (client) as shown in figure 41.

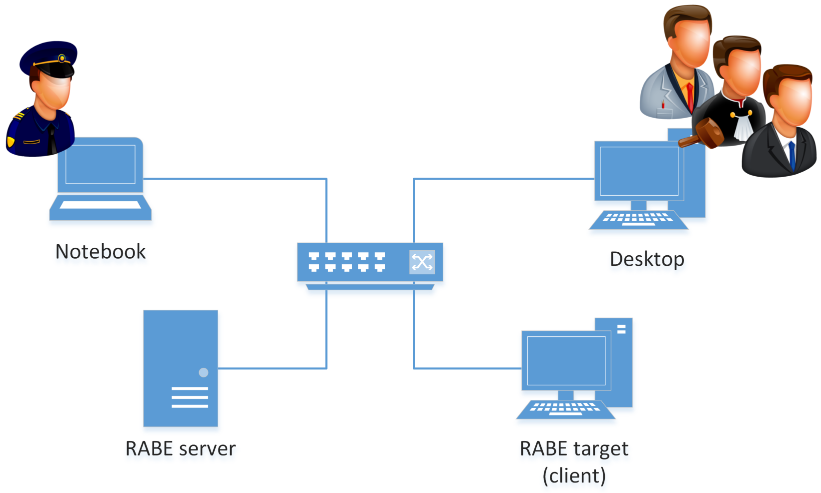


Figure 41: Test environment (simplified)

C.2 Tests

To test the workflow of the process the following two scenarios where created. These scenarios each describe a different workflow. The used method is grey box testing in which only the outcome is tested. The code is tested when necessary.

Test scenario 1 This test follows the workflow of the public prosecutor. The police creates a case with one target. The case number is 2014000005, the target number is ABCD0001NL and the target's IP address is 192.168.10.101. The company that hosts the target is Cheap Hosting and is located at the Industriestraat 10 in Den Haag. The police creates a legal request and sends it to the public prosecutor. The public prosecutor receives it, accepts the request and uploads the warrant. Then the police creates a technical request and sends it to the hosting provider. The hosting provides receives it, views the warrant and rejects the request. After a conversation on the phone between the police and hosting provider, the hosting provider accepts the request. The hosting provider downloads the default live image (stop at 5%), boots the target (client) and marks the target as booted. The police finds the target in the online target section of the dashboard and merges the temporary data with the actual target. The police then connects to the distributed storage device sda and starts the acquisition. Another storage device sdc is connect and re-exported. The police previews this device by connecting it to the investigator's computer and adding it to FTK Imager. The device is browsed for a second and then disconnected from the computer. The acquisition and re-export are stopped and the devices are disconnected. All participants log out of the dashboard.

Test scenario 2 This test follows the workflow of the examining judge. The police creates a case with one target. The case number is 201400006, the target number is ABCD0002NL and the target's IP address is 192.168.10.101. The target it hosted within a residence at the Nieuwstraat 20 in Den Haag. The police creates a legal request and sends it to the public prosecutor. The public prosecutor receives it and forwards the request to the examining judge. The examining judge receives it, accepts the request and uploads the warrant. A technical request to the hosting provider is not needed, because there is none. The target is hosted by a person at his residence. At the scene and during the search the police boots the target (client). The police finds the target in the online target section of the dashboard and merges the temporary data with the actual target. The police then connects to the distributed storage device sda and re-exports it. The police previews this device by connecting it to the investigator's computer and adding it to FTK Imager. The device is browsed for a second and then disconnected from the computer. Another storage device sdb is connect and acquired (stop at 5%). The re-export and acquisition are stopped and the devices are disconnected. All participants log out of the dashboard.

C.3 Results

All steps of the scenarios were followed. Scenario 1 had two remarks which were fixed quite easily and is mentioned in table 3. Scenario 2 was completed without any remarks.

Action	Participant	Remark	Fix
create case	police		
create target	police		
create legal request	police		
send legal request	police		
view legal request	public prosecutor		
accept legal request	public prosecutor		
upload warrant	public prosecutor		
log out	public prosecutor		
view case	police		
create technical request	police		
send technical request	police		
view legal request	hosting provider		
view warrant	hosting provider		
reject legal request	hosting provider		
accept legal request	hosting provider		
download default live image	hosting provider		
boot target (client)	hosting provider		
mark target 'booted'	hosting provider		
log out	hosting provider		
merge temporary target	police	no VPN IP address was posted to the table	added a 5 second sleep to the send_client_information service
connect to sda	police	shell log file not found	added the creation of the shell log file to rabe-server_mkdir_target.sh
start acquisition sda	police		
connect to sdc	police		
re-export sdc	police		
connect sdc to investigator's computer	police		
load sdc in FTK Imager	police		
browse sdc	police		
disconnect sdc from investigator's computer	police		
stop acquisition sda	police		
stop re-export sdc	police		
disconnect all devices	police		
log out	police		

Table 3: Test scenario 1