

Remedium Structure

Describing the architecture of the cooperative virus defense mechanism.

Index

Glossary	4
Introduction	5
Context Diagram	6
<i>Internet context</i>	7
<i>Intranet context</i>	8
Dynamic view	9
<i>High level view</i>	9
Role play locations	10
<i>Castrum</i>	10
<i>Clans</i>	10
<i>Forum</i>	11
Roles	11
<i>Civis</i>	12
<i>Triumvir</i>	12
<i>Centrum</i>	12
<i>Praetor</i>	13
<i>Patrician</i>	13
<i>Quaestor</i>	14
<i>Architecti</i>	14
Clients application	15
<i>Characteristics of a clients application</i>	15
Installing a clients application	16
<i>Distribution channels of clients installer packages</i>	16
<i>Clients without installer</i>	16
<i>Mass deployment / silent install</i>	17
<i>Checking dependencies and conditions</i>	17
Clients configuration and connection	17
<i>Configuring the clients application</i>	17
<i>Connecting to a castrum network</i>	18
<i>Connecting to a clan</i>	18
Clan and Castrum restrictions for clients	19
<i>Exclusivity principle</i>	19
<i>Banning a clients</i>	19
Clients score	20
<i>Practical example</i>	21
<i>Periodicity</i>	21
Triumvirate	22
<i>Triumvir types</i>	22
<i>Triumvir major tasks</i>	22
<i>Triumviri minor tasks</i>	23
<i>Common tasks</i>	23

<i>Quality attributes</i>	23
<i>Load Balancing</i>	23
<i>Loss of trust</i>	25
Triumvir election process	25
<i>Tabula rasa election</i>	26
<i>Triumvir major election</i>	26
<i>Triumvir minor election</i>	26
Civis functioning	27
<i>Verify the reliability of trusted files on disk against the knowledge on database</i>	28
<i>Flag potentially dangerous files</i>	28
<i>Monitor the introduction of new files on the machine</i>	30
<i>Contribute with knowledge to the network about the files on his disk</i>	30
<i>React to files detected as malicious</i>	31
<i>Assume the role of triumvir au pair with the civis role when necessary</i>	31
<i>Definition of anomaly</i>	31
Civis scanning process and security enforcement	32
<i>Deep scanning</i>	33
<i>Quick Scanning</i>	33
Civis - USB removable drives and security policies	33
<i>USB removable drives</i>	33
<i>Security policies</i>	34
Database	35
<i>Quality Attributes</i>	35
<i>Choosing a database system</i>	35
<i>Exposed issues</i>	36
Data stored in database	37
<i>File record</i>	37
<i>File record size per field</i>	38
<i>Handling file record collisions</i>	39
<i>File index</i>	39
<i>Seals of trust</i>	40
<i>Ignored files</i>	42
File system	43
<i>File Operations</i>	43
<i>Exposed issues</i>	44
Pending details	45
References	46

Glossary

To properly define our system we have selected particular keywords and designations that would be common to find during the roman era.

To some extent they hold a very reasonable fit to understand in a clear manner the expected role and characteristics for each of the actors in our system.

This glossary clarifies the meaning of latin terms employed in this documentation.

Term	Description
<i>fidem habere</i>	To be credible, or more literally “to have trustworthiness”
<i>census</i>	An official count of the members of a population
<i>tabula rasa</i>	To start without prior knowledge, from the beginning
<i>cliens</i>	Translated literally to client (<i>cliente</i> is the plural of <i>cliens</i>)
<i>auctoritas</i>	Power, prestige and authority inside a group
<i>a priori</i>	In advance, before a specific action takes place
<i>Forum</i>	A place where people can participate to share knowledge and decisions

Table G1 - Glossary

Introduction

As important as presenting our project, presenting the reasons that motivate the need for this development allows to understand where and how our project will make a difference to make the world a better place to live.

Viruses have been a reality that plagued computer users and organizations since the past few decades. It is a fact that no system is perfect when it comes to protect a machine from malicious actions.

In the past anti virus solutions would rely on databases of known attacks to identify and eliminate attackers and prevent security threats. With the growth of virus variants, the difficulty of indexing them also increased and these products incorporated behavioral and signature based solutions to gain flexibility and higher performance when scanning a given file.

Nowadays, the defense trend is moving to cloud computing environments where an anti virus will learn new threats based on the feedback from the clients installed on desktop machines. But even with this new method, none of these techniques has still been proven as effective enough to eradicate threats that menace computer users.

There is a visible war between anti virus companies and organizations against malware authors. The victims are computer users whose computers are used as weapons by either sides of this conflict.

As in any guerilla war, regardless the size of a given company or organization – it is not possible to effectively fight against an enemy that is anonym and constantly changing their attack techniques.

Cloud computing is a strong buzz word that has grown popular amongst anti virus products but we propose ourselves to move further and adopt a concept of human computing such as the one suggested by Luis von Ahn in CMU during his “Human computing” research.












We recognize that a system should allow human computing cycles to improve the available methods for handling a threat and secure a system.

This way, we are fighting fire with fire and ensuring that a new threat has better odds of being balanced with an efficient answer, provided not by anti virus makers but rather by the users themselves who are in direct contact with the threat.

We call this project “remedium”: a remedy for computers and users.

Context Diagram

A context diagram provides a notion of the scenarios where our system will be used, there are two distinct types of environments: Internet and intranet that are described on their own context diagrams. Below is a table explaining the elements that are found inside both context diagrams.

Elements	Description
 User	A human operator that uses a computer for his daily work tasks.
 Admin	A human operator responsible for administering a LAN (local area network).
 Attacker	A human operator with malicious intentions.
 Trusted workstation	A desktop computer machine considered trustworthy until proven otherwise (trustworthiness is defined further in this document).
 Compromised workstation	A desktop computer machine recognized as compromised in terms of security (security compromise is defined further in this document).
	A server computer machine that is used to serve a specific service.
 Gateway Router	A typical network router acting as gateway between two distinct types of networks.
	A network connection between two network devices. We make no distinctions between the physical characteristics for the purpose of the diagrams.
	A network area (public or private) to which other network devices are connected.
	A block representing the scope of a given network area.
	A block whose elements on the top block represent a typical scenario that can be found repeated multiple times.

Internet context

Figure CD-1 details the Internet environment context for our design.

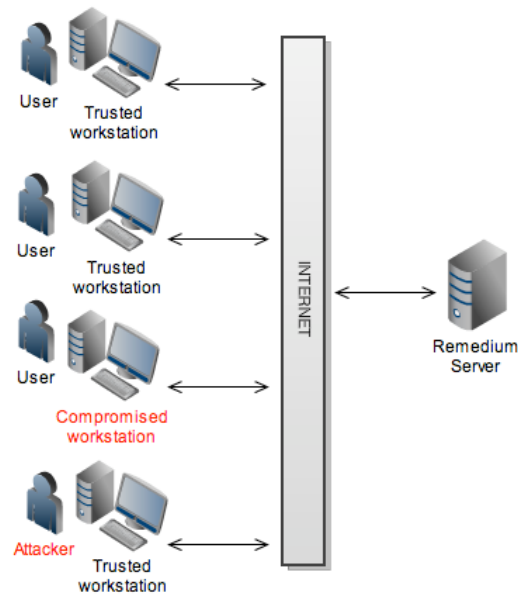


Figure CD-1

On this environment, users are people who own the workstation machines and connect regularly to the Internet. Our system provides a client application that runs on their workstations and interacts with a server of type “remedium” that forms a group of defense.

When a workstation is reported as compromised on this system, information about the threat is collected and transmitted to other workstations that increment this information to their internal threat detection mechanism.

The compromised workstation has an internal mechanism to attempt quarantine the affected files and replace them with equivalent files from trusted workstations.

Attempts of sabotage by human attackers from trusted workstations are expected; to mitigate their impact, we ensure that each element has a limited level of influence inside the group and his actions are always verified by other elements. We follow the design principle: “trust no one because no one trusts me”.

Each element (workstation + user) contributes to make the defense group stronger by submitting and exchanging information about possible anomalies that occur, allowing a proactive defense mechanism that allows isolating threats before they manage to attack other elements in the system.

Intranet context

On figure CD-2 we depict the design diagram when our system is interacting an intranet based environment.

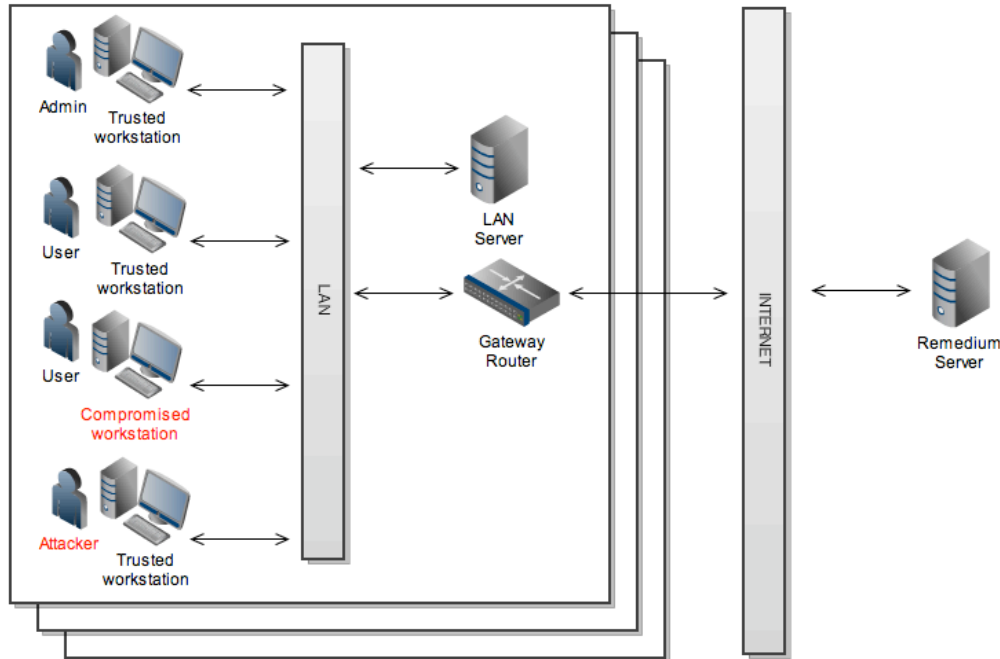


Figure CD-2

We assume an intranet as a private network where all computer workstations are managed by a given organization. The main different is that we assume users to have no administration permissions on their assigned workstations and a specific user with the role of administrator over these machines.

Workstations inside an intranet may be restricted from access to a public “remedium” server on public networks. Under this scenario, a workstation defined by the administrator shall be allowed to connect through the gateway router to access the Internet and the system itself is also prepared to run in isolation.

We react to situations of compromised workstations and attackers as described on the previous diagram to minimize the impact of these threats.

Dynamic view

A dynamic view presents the components and connectors of our system while running.

High level view

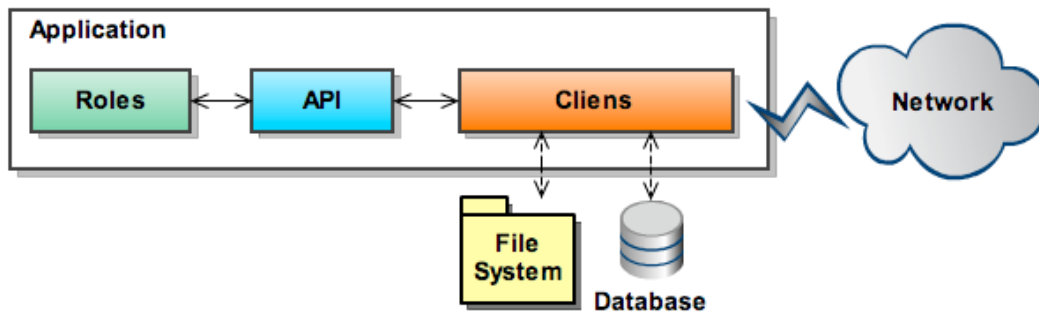



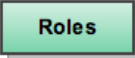








Figure DV-1

Elements	Description
	The cliens application is the core element of the application where other elements connect to compose our system.
	The API (Application Protocol Interface) provides an abstraction layer to interface the roles and the cliens application.
	Elements inside the application exchange data using the API that is detailed on the "API" chapter of this document.
	Each type of role has a specific behavior that is detailed o the "Roles" chapter on this document.
	File system refers to the file system available inside each workstation.
 Database	Database represents knowledge repositories where data is stored. These databases are hosted outside the application.
	The application box represents elements that are inside the cliens application.
	The cloud represents a network regardless of being private or public.
	Exchange of data between the cliens application and an element physically located inside the workstation.
	Exchange of data between two cliens applications using the cloud network as mean of data transmission.

Role play locations

Inside our system, we follow the Roman theme and defined three distinct spaces where our actors will intervene.

- Castrum
- Clan
- Forum

On this section we also detail the organization and structure for each of the locations where our system will act. It is not an exhaustive level of detail, the technical context under which a given location is appropriate is detailed further ahead in this document.

Castrum

The word castrum was used by the ancient Romans to designate buildings or plots of land reserved to or constructed for use as a military defensive position.

In our context, we define castrum as a private network that we are defending.

This fortified territory can typically assume the range of a LAN (*Local Area Network*) or WAN (*Wide Area Network*) under the control of an organization. The administrator of the castrum is denominated as praetor.

The roles played by machines and humans inside the castrum are described in further paragraphs of this document.

Clans

A clan is part of the cliens population that is identified as belonging to a specific group. In the case of our system, we assume a clan to be a group of cliens that use the Internet as the common network that is shared amongst them all.

The structure of a clan is similar to a castrum but is lead by a Patrician, whose roles are similar to some extent as administrator of this group albeit with reduced control over each cliens on his group when compared to a praetor.

The extent of action of the actors inside a clan is defined in the section that describes each role in detail.

Forum

In resemblance to a forum in the Roman days, a forum is a central location on the Internet where human actors of this system and also people outside the system visit to exchange knowledge, feedback and make decisions regarding the future.

The decisions that are made by specific roles located at the forum might influence the knowledge that is available to clans and castrum around the world.

As mentioned for previous locations, the interactions for each actor is described on their own section to help understood the context where the forum is integrated.

Roles

Also following the Roman theme, inside our system we define the roles that are currently available to be played by each actor.

These actors can either be human or machines depending on the purpose of the role that is played. Each role has a status of *active*, *inactive* or *omitted* depending on the intended behavior profile for each cliens application.

Role status	Description
Active	Role is active inside the cliens application.
Inactive	Role is available to run on the cliens application but deactivated.
Omitted	Role is not available to the cliens application.

More roles should appear in the future as necessary and we dedicate a sub-section describing the characteristics and behavior for each role that is currently identified.

Role	Type	Location(s)
Civis	Human	Castrum, Clan
Triumvir	Machine	Castrum, Clan
Centrum	Machine	Castrum, Clan
Praetor	Human	Castrum
Patrician	Human	Clan
Quaestor	Human	Forum
Architecti	Human	Forum

This chapter details the roles of these actors inside our system from a high level perspective. The technical detail of the behavior and interaction with other roles is defined at a later point on this document.

Civis

Civis is the direct Latin translation of “citizens” and represents the largest percentage of active elements in the system.

This role is responsible for enforcing the local security of the machine where it is running and is intended to run inside the context of a castrum or clan.

This role will interact with a machine in the role of triumvir that has been assigned to bridge his contact with the other elements in the network. The civis is also equipped with recovery mechanisms that allow him to reconnect to the network in case of this triumvir connection fails.

The technical details for this role are documented in the section “Civis functioning”.

Triumvir

Triumvir is the direct Latin translation for “three men” and is used to designate a group of three persons (in this case machines) that perform an important role inside the group where they are integrated.

A triumvir is the main communication bridge between each civis and the rest of the network.

This role is responsible for ensuring that all cliens assigned to him get timely knowledge about the status of the clan or castrum. It is also responsible for warning other triumvirs about status of his own cliens.

In case of fail or mistrust, another triumvir is elected from the list of civis under his charge. The details for elections and interactions with other roles are described in the section entitled “Triumvirate”.

Centrum

Centrum, from the Latin translation of “center”, is a role assigned to a machine that become considered as the central point of a given network, let it be a clan or castrum.

Actors inside a given network will resort to a specific centrum whenever in need of enrolling for the first time on a clan or castrum. This act is also valid for cases when the connection between two actors is broken and needs to be reestablished.

The interaction between centrum and other roles is described on the “Centrum behavior” section of this document.

Praetor

In Latin, praetor is the military role assigned to commander of a given castrum.

In general terms, the praetor has the means to monitor and administrate the castrum. Below is a more detailed list of his specific characteristics:

- Authorizes the software update of the cliens under his control to newer versions;
- Decides which plugins are made available to the cliens;
- Authorizes or denies the entrance of new cliens on the network;
- Sets the security policies to be followed by other cliens;
- Sets threat response policies in the castrum:
 - Policies of usage for USB storage media;
 - Activates the automatic overwrite of untrusted files in the system;
- Response measures in case of attack:
 - Warn cliens user about threat entry;
 - Disconnects cliens from network;
- Calls the dismissal of the established triumvirates in case of mistrust.

Patrician

In the roman days, patrician would be the head of a family and thus explaining the evolution of root word Pater to Patri and later Patrician in our days.

In our system, patrician is a member assigned with this role inside a given clan. He can be the founder of the clan or be someone assigned to this role by the former patrician.

Below we list some of the relevant characteristics for this role:

- Leader of a clan;
- Defends the interests of the cliens in his clan;
- Promotes his own internal forum and set of principles;
- Often, he is the most trusted person on the clan;
- Sets objectives or goals for his clan.

The purpose of a patrician is to customize a clan to fit the interests of a specific community.

For example: when English is not a suited language for the members of clan, a patrician can define the language of the forum and messages interchanged between clan members to be performed exclusively in a language that is more comfortable to the clan elements.

The patrician therefore becomes a link between the members of his clan and the rest of the world.

Quaestor

The main purpose of the quaestor is to assess the quality of the knowledge that is integrated on the common databases made available to every castrum or clan.

- Verifies accuracy of knowledge;
- Reports faults of citizens with auctoritas;
- His validation is required to allow new knowledge to be added on the databases available at the official Internet site;
- Works at the official forums on the Internet.

Architecti

The purpose of an architecti is to coordinate the joint effort of knowledge gathering to areas where it is deemed as needed.

For example, if there is the need to catalogue all the possible MS Windows XP versions, then he will seek the support of a clan or praetor that is willing to assist him in the task of adding these files to the knowledge database.

Being the most trusted element on this system, he can also add new knowledge without requiring a more formal process of inspection.

- Specialist in defense;
- Asks assistance to praetors and patricians to provide manual labor;
- Located at official forums on the Internet.

Clients application

Our system is based on the concept of “clients”, a software application that is shared across all the actors in the system that was described on the previous chapters.

Clients is the direct Latin translation for “client”. Inside our system we find each actor, let it be a machine or a human, playing a specific role where the clients application is used as the base tool that interacts with other actors.

The clients is an instrument to carry forward the actions assigned to each role.

Characteristics of a clients application

The list below describes the basic features shared across each clients application.

- Modularity – new roles can be added as modules to the clients whenever available and authorized;
 - New modules take advantage of the clients API to interact with the system;
- Modifiability – clients functionality can be expanded with resort to external plugins:
 - Add new malware detection techniques available as API;
 - Customizes existent functionality for specific scenarios;
 - Third party providers implement customized plugins;
- Communication – a clients is capable of talking with other clients and exchanging information:
 - Peer to Peer communication is the standard of way of connection between clients;
 - Mass broadcast of messages is restricted;
 - Each message is synchronous between both ends;
- Security:
 - Communication protocol between two clients is encrypted with a dynamic algorithm;
 - Communication ports can be customized to prevent jamming;
 - Communication between two points requires authentication;

- The cliens application is capable of verifying if the internal components have been modified without permission;
- Some roles performed by a cliens will only function when the user is properly authenticated;
- Robustness:
 - If a cliens is separated from his contact link to the network (triumvir), it is still capable of reconnecting to the network using other link contacts considered trustworthy (centrum);
 - Exchanged messages are followed with a checksum of the previous message (if present) to ensure they were correctly received and sent by both ends;

Installing a cliens application

A cliens application will be made available with an installer appropriate for the Operative System where it is intended to run.

Although Java is in essence platform agnostic, each supported platform will have a native installer to provide a user-friendly installation process.

Distribution channels of cliens installer packages

The cliens software will be officially made available at <http://remedium.me> – this is the location from where potential users can download the application.

Although an official distribution site exists, many websites around the Internet will host the cliens software and make it available to their visitors regardless of consent from the development team.

In extreme cases, malware authors might modify the cliens software to incorporate malicious modifications that fools users into installing a trojan application.

Often, it is also noted the case where third-party download sites do not update often the version of the cliens software that is made available by their service. This situation raises a security risk as these users might become exposed to flaws only corrected on newer versions.

Cliens without installer

A portable version of the cliens software without installer to run the program directly is also made available to allow full control over the install procedure.

Mass deployment / silent install

System administrators will have available the option to run the installation setup in a fully automated manner.

An administrator can provide a text file where the custom settings are read and used to install the cliens application without requiring any user input.

Checking dependencies and conditions

- Java support: The cliens installer is a native executable of the target operative system that allows verifying if the system contains Java installed before installing the cliens application.
- Available disk storage: we define that each machine needs at least 5Gb of free disk size to allow the cliens application be installed.
- The cliens application is only supported on MS Windows above Windows 2000, Apple Mac OS version 10 and Ubuntu Linux.

Cliens configuration and connection

This chapter specifies the possible modes of configuring a cliens to connect onto a given type of network.

Configuring the cliens application

The installing setup provides a GUI wizard to configure the cliens on the target workstation.

Configuration differs between Castrum or Clan enrollment.

If the cliens is meant to run inside an intranet environment, a system administrator or user will need to manually define the centrum machine where the cliens connects to enroll on the castrum network.

If the cliens is meant to run at a home computer with access to the Internet, the cliens software accesses the official site at <http://remedium.me> to retrieve a list of available clans in which he is allowed to enroll.

Configuration options

- Configure the castrum communication port (default is 80);

- Select the folder where the cliens application will be installed;
- Opt by joining a clan or castrum at a later moment.

Configuration for intranet mode

- User needs to type the name of the machine inside the castrum with the role of centrum.

Configuration for Internet mode

- User selects a clan from an online list of official clans at <http://remedium.me>.

Connecting to a castrum network

If this is the first time that a cliens is connecting to a given castrum, the connection will begin with a direct contact of the cliens to the machine performing the role of centrum.

The centrum can be reached using either the domain name or IP address of the machine that was defined by the user when installing the cliens or modified at any given time using the configuration panel.

After the centrum is contacted it will require the explicit permission of the praetor to enroll inside the castrum.

Once the praetor accepts the enrollment of the cliens, the centrum will assign a triumvir to interact with the cliens and also create a unique identification number to distinguish this cliens from all the other elements of the castrum.

Connecting to a clan

If the cliens application is installed on machine not integrated inside an intranet environment such as a home computer then the user will be allowed to join a clan or run in standalone mode.

When connecting to a clan, the procedure is slightly different. With access to the Internet we can ensure that all cliens connect to our official site as the central point of communication.

On the official site we provide a list of official clans available to enroll. A clan is similar to some extent with an intranet castrum and will only allow the cliens to take active part inside a single clan while allowing its enrollment up to 10 different clans or castrums at any given time.

The list might present other relevant details such as the popularity (measured by the number of cliens associated), by geographical location, language, mission statement and so forth.

Clan and Castrum restrictions for cliens

This section details the restrictions imposed to all cliens applications integrated inside a castrum or clan defense group.

Exclusivity principle

By definition, each cliens application can only take active part on a single castrum.

This measure ensures that machines running cliens applications with malicious intents become limited in their degree of possible influence to other castrums.

Each cliens application can present their request of enrollment up to 10 different castrums or clans. The candidate might be accepted on several of these networks but remains restricted to choose a single castrum at his choice where he will participate.

If the cliens user opts to take active part at a different castrum, his level of fides habere at both the new and old castrums will start from zero.

This reset on the fides habere score is intended to secure the election process conditions. A machine with a higher level of trustworthiness gained over time should preferably be elected to perform critical management roles in the castrum.

Banning a cliens

A cliens might be considered as compromised or untrustworthy at some point. In the case of a castrum, in typical conditions we may consider that the praetor has physical access to the machines where the cliens applications are running and solve the issue.

However, for the case of clans we cannot assume the same method since the machines are considered as being geographically dispersed.

If an attacker enrolls in a clan with the intention of manipulating the accuracy of the knowledge base for his advantage and is exposed by the patrician, the ban procedure will allow removing a given cliens from the clan without requiring his permission.

This operation is also valid for cases where the participation of a cliens in the clan community is considered as “troll” (aggressive to other members) or “spammy” (sends a significant amount of unsolicited messages).

The patrician has full control regarding the cliens that are authorized to be part of his clan and a cliens application can be banned regardless of the roles performed at the time of ban.

Cliens score

Some cliens machines are recruited to perform roles such as triumvir where an intensive consumption of physical resources such as RAM, CPU cycles, disk space and network bandwidth is expected.

To manage our networks efficiently, we devise a score that provides an idea of the physical stress that can be supported by each element in the network.

This score is associated to different weights of percentage that focus on specific characteristics of the machine.

On each test, there is a value that is set as the maximum score possible to reach for each particular test. If the machine scores above this maximum value, the resulting score is capped to the maximum value.

Table SC-1 Score assigned to each test

Test title		Overall Weight	Max value	Description
A	CPU speed test	20%	50 000 cycles	Run a cyclic loop test under a minute and count number of cycles.
B	RAM available	20%	2Gb	Free RAM currently available on the system.
C	Free disk space	10%	5Gb	Checks the free available disk space on the drive where the cliens application is running
D	Bandwidth speed	20%	1 second	Measure the time it takes to download a 1Mb file from a given triumvir.
E	Machine uptime	30%	2880 minutes (2 days)	Obtain the elapsed time since the last system reboot.

We ensure that some characteristics such as uptime more relevant than others. All individual scores are obtained using a direct proportion formula with the exception of the test D (bandwidth) that use inverse proportion, meaning that a higher time for downloading the file will progressively decrease the obtained score.

The overall cliens score is calculated with the following algorithm:

$$((A_x \times 20) / A) + ((B_x \times 20) / B) + ((C_x \times 10) / C) + ((D_x \times 20) / D) + ((E_x \times 30) / E) / 100$$

Where Ax, Bx, Cx, Dx and Ex represent the values obtained from tests on the machine where each alphabet letter in uppercase represents the type of test as detailed previously.

Practical example

We provide an example for a workstation that provided the following results:

- Ax - CPU test – 23 000 loop cycles under a minute;
- Bx – Free RAM test – 0.8 Gb of free RAM available;
- Cx – Free disk size test – 150 Gb of free disk space;
- Dx – Bandwidth test - Downloaded a file sized in 1 Mb under 11 seconds;
- Ex – Uptime test – the system reports the system as being online since 3 days, 2 hours and 19 minutes, equivalent to $(3 \times 24 \times 60) + (2 \times 60) + 19 = 4459$ minutes.

Table SC-2 Score gathered from a test example

Type	Initial result	Max accepted	Percentage of Max	Final weight in overall score
A	23 000	50 000	46%	9.2
B	0.8	2.0	40%	8
C	150	5	100%	10
D	11	1	9.09%	1.81
E	4459	2880	100%	30
Overall score:				59.01

The calculation process is translated to the following formula:

$$(((23000 \times 20) / 50000) + ((0.8 \times 20) / 2) + ((5 \times 10) / 5) + ((20 / 11)) + ((2880 \times 30) / 2880)) / 100 = 59.01$$

Periodicity

Each cliens will self-perform the cliens score algorithm when more than 24 hours have passed since the last score test has occurred.

After the score is completed, the results are sent back to the triumvir as soon as possible.

Triumvirate

A triumvirate is established amongst three powerful members of a given population.

We define “powerful” according to a self-test that each cliens performs to evaluate its own physical characteristics along with the reputation (fidem habere) earned during his time tied to the clan or castrum.

This cliens score allows ranking all the members and selecting the ones more fit and trusted to perform the role of triumvir.

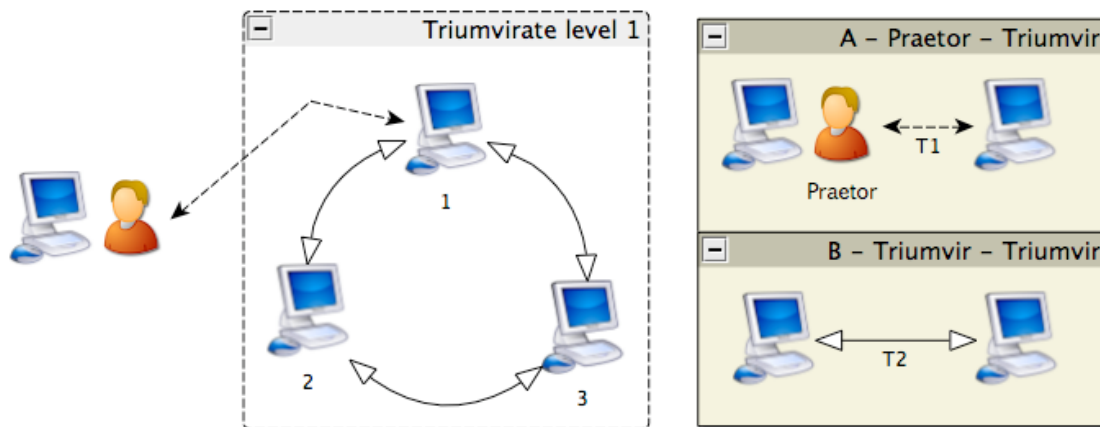


Figure TR-2

As depicted on the diagram above, hierarchy is also present. One of the triumvirs will hold the position of triumvir major and the others become triumvir minor. This hierarchy allows the praetor to monitor and control the triumvirate

Triumvir types

This section defines the behavior expected from each triumvir role that we categorize in three types:

- Triumvir major, the leader of the triumvirate;
- Triumvir minor, triumvirs at lower rank than triumvir major;
- Common tasks, performed by triumvirs regardless of their type.

Triumvir major tasks

- Distributes cliens amongst triumvirs;
- Direct contact with the official web site if Internet access is provided:
 - Gets updated DB knowledge from either the Internet or other castrums;
 - Contributes with new knowledge to other clans or castrums;
- Calculates global threat level in castrum or clan.

Triumviri minor tasks

- Monitors the administrative decisions of the triumvir major and evaluates if they are correct;
- Keep a synchronized copy of the knowledge present on triumvir major to assume his position if necessary.

Common tasks

- Keeps his own list of cliens;
- Gathers census from cliens:
 - OS platform / Version; Physical status (RAM, disks, etc);
 - Tracks uptime of each clien;
- Assigns fidem habere;
- Send and receive knowledge to/from his cliens;

Quality attributes

- Robustness – If one or two triumvirs fail, only one is required to restore control. If all triumvirs fail, there is a recovery protocol initiated by the castrum's praetor to reassemble the triumvirate structure;
- Security – a given set of knowledge must be equal on all triumvirs that allow comparing and assuring that no tampering occurs; each triumvir inquires other triumvirs to ensure they are not compromised.
- Load balancing – the management of cliens is distributed to each triumvir according to their physical capacities.
- Scalability – each minor triumvir can become the father triumvir major of a new child triumvirate control center.

Load Balancing

All the cliens on the castrum are assigned to a specific triumvir that is used as their gateway to exchange knowledge within the castrum.

Triumvirs are expected to have physical limitations regarding the number of possible cliens that they are capable of handling.

For example, in a small sized company with 50 workstations, it is feasible that three triumvir machines have the conditions to handle the interactions with cliens but over the Internet on the case of clans, it is a realistic scenario to see clans containing

over 100 000 cliens connected that will stress each triumvir machine to its physical limits.

The stress imposed to each triumvir is an equation that produces a percentage reflecting the average CPU usage and free RAM.

If the triumvir stress crosses a given safety level, the triumvir himself is capable of spawning a new sub triumvirate in which he will assume the lead role. This action allows sharing the process load with the new triumvirs at the lower level.

If this triumvirate spawn occurs, each new triumvir is assigned with an identification number to properly identify the triumvirate to which it belongs. On figure TR-1, an example is given regarding how this numeration should occur.

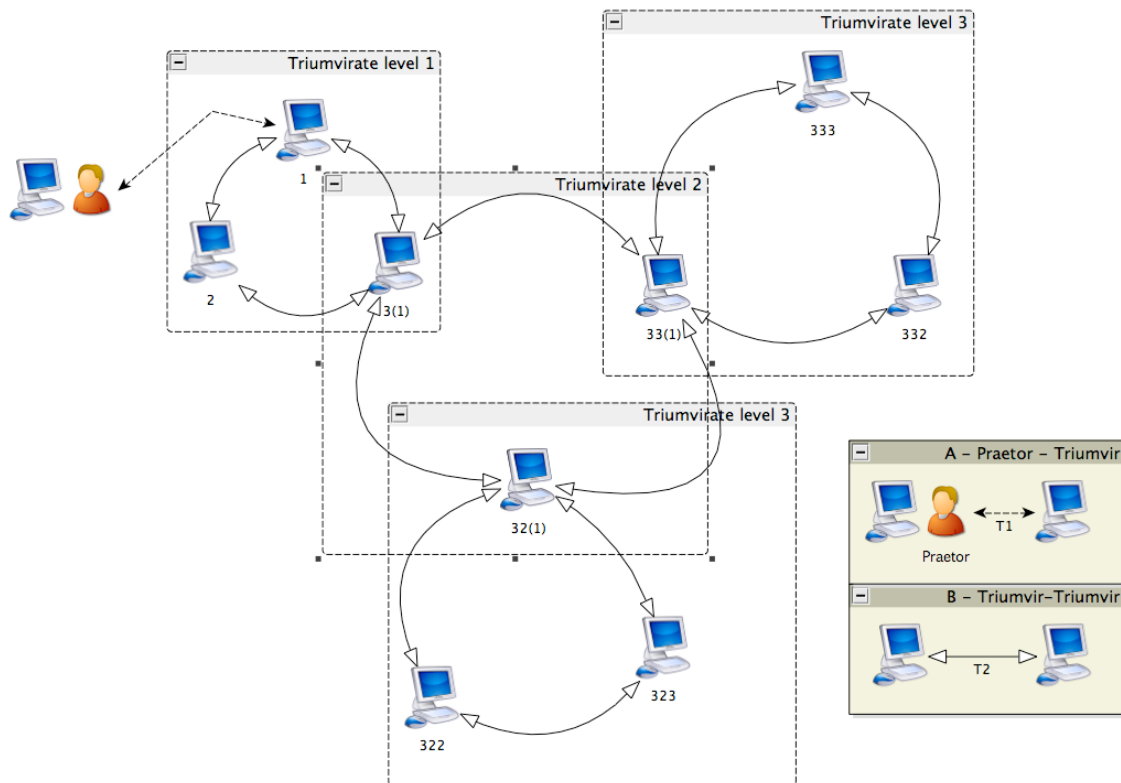


Figure TR-2

When using the above diagram as example, looking at the initial triumvirate at level 1 we can see that triumvir 3 has spawn a new triumvirate that is found at level 2. The triumvirs minor located within will be labeled 32 and 33, the triumvir major will assume the identification as 31 for the new triumvirate and preserve his denomination as 3 for the upper level triumvirate.

To continue exploring this example, a new triumvirate is spawn at triumvir 33 creating a level 3 triumvirate. On this case, the new triumvirs minor on the group will assume the numbers 332 and 333 and so forth.

Each triumvir major from level 2 and beyond is always follows the same protocol expected from a triumvir on the upper level. Meaning that any changes that occur on the upper level will reflect on the lower levels as well.

If a triumvir with attached sub triumvirates loses trust to perform his role, another machine from the lower ranks of his own sub triumvirates is automatically promoted for this position.

Loss of trust

This situation occurs when a triumvir is no longer considered as fit to perform his role; there are distinct conditions that might trigger this action and require a new election to occur:

- Time expiration: the praetor may set a time limit for cliens to perform the role of triumvir and force a renewal;
- Loss of communication: when a triumvir is offline for a given set of time;
- Loss of reliability: the triumvir uptime on the network becomes inconstant;
- Better candidates: if machines with a better score are available, they are used to suppress the need of spawning new sub triumvirates;
- Praetor decision: a tabula rasa election might be instantiated by the praetor;
- Security is compromised: when both triumvirs minor report a security mismatch on the triumvir major.

Triumvir election process

The discovery and selection process of cliens that can later become triumvirs is divided on the following types of actions:

- Tabula rasa instantiation;
- Triumvir major election;
- Triumvir minor election.

Tabula rasa election

When there is no initial circle of triumvirs already defined, the system will engage into the tabula rasa [3] election mode.

There are two distinct network scenarios under which any triumvir election might take place:

- Internet;
- Intranet.

Under an intranet environment, the same machine used by the praetor to manage the castrum will temporarily assume the role of triumvir major. The praetor can also decide to explicitly assign another cliens in the network to assume this role.

Whenever there is loss of trust in the triumvirate management, the praetor has enough control to begin (re)building the castrum from tabula rasa .

Triumvir major election

To provide an automated triumvir major election process, we require to have a triumvirate established a priori.

This election takes place on the following situations:

- A triumvir minor challenges the trustworthiness of the machine performing the role of triumvir major;
- The triumvir major loses the trust from the praetor.

The triumvir major is selected from one of the two triumvirs minor according to their cliens score. The praetor gets notified of this election and respective role change.

Triumvir minor election

Candidates for the triumvir role are selected from the list of available cliens in the castrum or clan.

The selection process is based on their individual score at the triumvir algorithm and also from the acquired level of fidem habere.

This selection process is lead solely by the triumvir major that enlists the possible candidates and then invites the selected cliens to join the available triumvirate positions.

Some questions are still left open in regards to the functioning of the triumvir role; they are listed below and will be detailed along with the development of the project.

- How many cliens can be managed at each triumvir?
 - Coordinate synchronization (stress algorithm that allows to preserve around 50% of free resources);
 - Calculate theoretical max number of simultaneous connections;
- How can the knowledge databases be secured or accessed?
 - Load balancing?

Civis functioning

This section details how a cliens application in the role of Civis is intended to behave and interact with other cliens applications on his network.

The first part will define the behaviors expected to find in common when the civis is connected to either a castrum or clan. After explained the common behavior patterns we will detail the particular behaviors inside each type of network.

Civis has the following responsibilities:

1. Verify the reliability of trusted files on disk against the knowledge on database;
2. Flag potentially dangerous files:
 - a. User of machine can manually report a file as dangerous;
 - b. Knowledge database can expose a given file as dangerous;
3. Monitor the introduction of new files on the machine (USB, Internet, shared folders and critical folders);
4. Contribute with knowledge to the network about the files on his disk.
5. React to files detected as malicious:
 - a. Apply quarantine;
 - b. Replace files with trusted versions from other cliens if possible;
 - c. Prevents machine from being used until administrator takes action.
6. Assume the role of triumvir au pair with the civis role when necessary;

Each of these topics is now detailed on the sub-sections below.

Verify the reliability of trusted files on disk against the knowledge on database

One of the most important tasks for our system is the detection of malicious activity affecting the files on disk. We do not compete against anti virus products already installed on a given machine; our verification is performed with focus on files that we verify against a knowledge database to check if they match or not.

For this task to succeed, there is the need to catalogue all files that are found inside the machine and use a database to store information about each of them:

- SHA-2 checksum;
- Filename;
- File version (if applicable);
- File size (in bytes);
- File creation date; (if applicable)
- Security classification: trusted, malicious, suspicious, cautious or ignored;
- Additional comments
 - Date added to database;
 - User name and clan or castrum of who added the file;

The database is kept on the same machine where the clients application is running; the knowledge is updated from a triumvir and also built from the profiling of file activity of disk activity inside the machine.

Flag potentially dangerous files

- a. *User of machine can manually report a file as suspicious or malicious;*
If a user suspects that a given file on his disk is potentially dangerous (email attachment, a file that appears on a public share and so forth), it is possible for users to flag these files and pass along this information to the rest of the network.
- b. *Knowledge database can expose a given file as malicious;*
Even though the primary objective of this system is to verify the trustworthiness of files on disk, it is also possible to keep in the database a list of files that are identified as malicious. The details of this detection are specified over the following paragraph.

The process of indexing new files on the database includes the task of detecting anomalies. On figure CF-1 we present the logical diagram of this process.

On our logical diagram, the square represents an action while the diamond form represent decision blocks.

The circle is the most significant action that occurs whenever an anomaly is exposed.

To keep the diagram concise, we don't represent actions such as indexing subfolders and other implementation details.

We start with a simple loop through all the files found inside a given folder.

If the file extension is recognized as an executable file (we are only covering a few selected extensions at the moment), then we proceed to verify if it is on the group of files that has a PE (Portable Executable) header from where we can read the file version (.exe .dll .scr .ocx .drv).

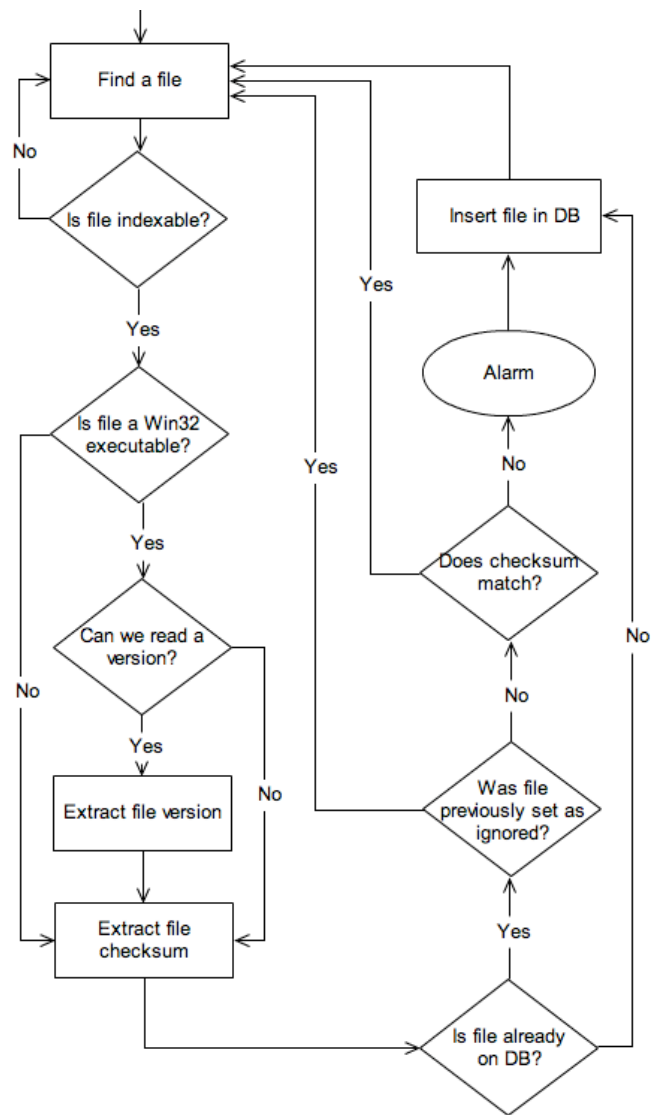


Figure CF-1

There are cases where a file with PE header does not bear a version. On these cases, we treat them as plain files.

The next step is obtaining the checksum. If the checksum is not recognized, it is added on the DB, otherwise we first check if it is on the list of files to ignore along with the version information (if available).

In case the file is not on the "ignore" list and the checksum does not match our records, an anomaly has been detected and the alarm sounds.

Monitor the introduction of new files on the machine

a. USB

A civis can detect whenever a removable drive is inserted on the system and proceed with scanning and cleaning procedures as necessary. We include not only USB drives but also any drive that may be introduced on the system while running such as memory cards and firewire drives.

b. Internet

Specific folders inside the system may be monitored constantly for the appearance of new files arriving from the Internet, folders such as the Desktop and Downloads are default locations where most people and browsers and users place their downloaded files.

c. Shared/Critical folders

When working at an intranet environment, it is common to share a given folder with other coworkers in the same team or department. Often is the case when an infected machine will place a new binary with a suggestive name at shared folders, eluding other users to infect their machines when running the binary.

Certain critical folders is OS such as the Desktop, Documents and Download folders in MS Windows should also be monitored for changes as they are often the preferred location for malware to host their malicious code.

Contribute with knowledge to the network about the files on his disk

Just as important as monitoring the files present on disk is to gather additional knowledge than can be shared with the network. For example, identifying an attack to a given file on disk will trigger additional attention on other cliens applications about the file was affected.

If more cliens are reported as being under attack, the network administrator will be better informed about this threat and have an opportunity to react accordingly.

React to files detected as malicious

- a. *Apply quarantine;*
When a file is flagged as prejudicial to the machine, it won't be erased from disk but rather placed in quarantine where it may be submitted in a later time to third party anti-virus that may assess if it is a threat or not. This feature might also contribute to improve the third party antivirus knowledge database about malicious files.
- b. *Replace files with trusted versions from other cliens if possible;*
The network administrator might also enable the cliens application to replace the malicious file with a version that is considered correct from another cliens application.
- c. *Prevents machine from being used until administrator takes action.*
Under an extreme scenario of security, whenever anomalies are detected - the cliens application can prevent any further participation of the machine in the network either by disabling the network interfaces or forcing the machine shutdown until an administrator reverts this condition.

This feature disrupts the normal operation of users at the affected machine but protects the network of attacks originating from this location.

Assume the role of triumvir au pair with the civis role when necessary

Each cliens application is enabled to perform the role of triumvir when elected from a pool of active civis inside the network.

Since each machine will differ in terms of physical characteristics to perform this role, a score is computed by the cliens application and also assigned a score according to its presence on the network. This score is detailed at the "Cliens score" section.

Definition of anomaly

On a given set of machines, knowledge will be gathered about the files that are found on the disk of each cliens. This knowledge is passed onto the respective triumvir to where a cliens is connected and eventually propagated across the other triumvirs as depicted on the diagram AN-1.

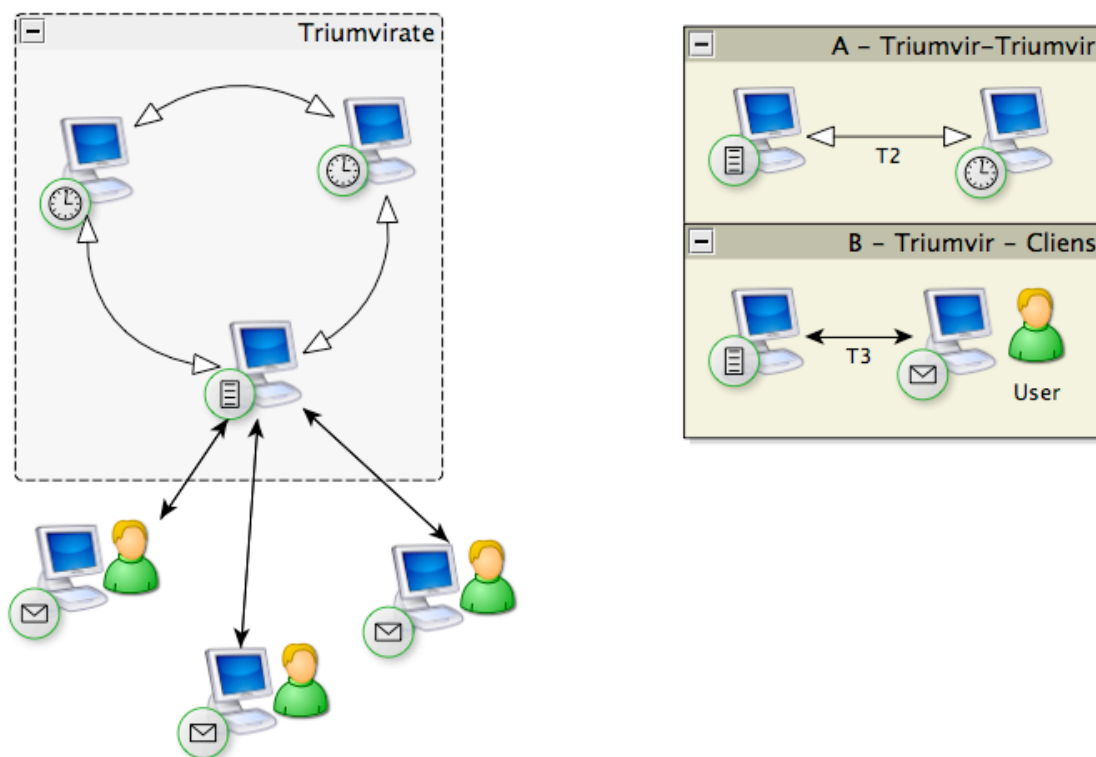


Figure AN-1

On scenario A (Triumvir - Triumvir), the triumvir is synchronizing the knowledge gathered from his clients with the other triumvirs.

On scenario B (Triumvir - Clients) we see the clients application sending his knowledge about the files on disk to the triumvir.

An anomaly is identified on the following cases:

- The information from the file on disk mismatches the information from the knowledge base;
- A prejudicial file is identified.

Civis scanning process and security enforcement

One of the most important protection mechanisms is the deep scanning that consists in comparing each file on disk against the available knowledge databases.

Deep scanning

Under typical conditions, it is a process that is performed upon the first time that the cliens application runs on the machine and then performed on a given time interval, preferably within 24 hours given the time it may take to complete this operation.

The deep scanning process consists on the following steps:

1. Create a list of all files locally available on the machine;
2. Extract information from each listed file where applicable (file version, file size and so forth);
3. Compare the information of each file against the database information for that file if available;
4. Apply the recognition of signature patterns on specific files;
5. Create a list of results.

The list of results may trigger the cliens application to react as specified by either the user himself or the praetor on the castrum.

Quick Scanning

The quick scan process is similar to the scan described on the deep scan process with the difference that it will be limited to the root of a drive or key folder without moving to inner sub-folders in order to keep this process as fast as possible.

Civis - USB removable drives and security policies

This section describes in detail the intended behavior of a civis regarding USB media.

USB removable drives

The cliens application will be prepared to continuously monitor the usage of USB removable drives on the system.

Upon detection of a new storage drive, it will perform a quick scan on the root of the drive to detect harmful files listed on the database.

Security policies

The security policies are set by the civis user or by the praetor whenever the cliens is connected to a castrum.

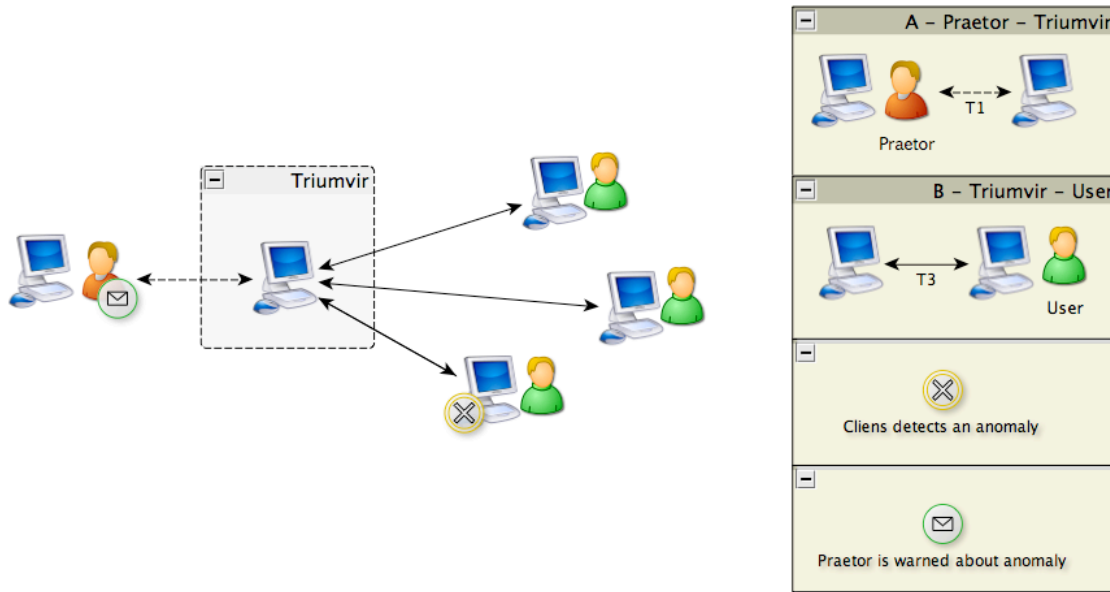


Figure USB-1

These policies define the cliens application behavior to the scenarios listed below:

- An anomaly between the information of a file on disk and the information on the knowledge database is found;
 - Possible reactions (order is not relevant):
 - Warning message is sent to praetor and user about mismatch;
 - Move file to quarantine location if it is not a critical system file;
 - Replace file with another file from a trusted location;
 - Preemptively shut down machine to prevent propagation risk.
- A USB storage device is introduced on the system;
 - Possible reactions (order is not relevant):
 - If USB identification is not on the list of allowed devices then disallow device from being made available to the system;
 - Log the identification of USB storage devices that are introduced on the machine;

- Run the quick scan process to detect viruses.

Database

The project relies on the intensive use of databases that are used to document the gathered knowledge about analyzed files.

Quality Attributes

Since one of the most relevant uses for the database is storage of details about files that are considered as safe, we need to take into consideration several factors that are defined as quality attributes.

We start by defining the ideal quality attributes for a database on the client side using as standard desktop machine a Pentium IV at 2Ghz with 512Mb of RAM.

- **Portability:** reuse the existing database across different operative systems and services;
- **Performance:** A standard desktop computer shall be able to operate a database with a maximum value of 4Gb worth of data stored per database;
- **Distributability:** The database is distributed with the software application without requiring installing external dependencies;
- **Responsiveness** – Usage of the database cannot deteriorate the response of the software application to user input in a timely fashion;
- **Costs and licensing** – The selected database technology must be available free of costs with authorization to distribute both on commercial and non-commercial work environments.

Choosing a database system

In the market there are some solutions that fit most of the requirements and quality attributes. Preference was given to flat file databases to ensure that Distributability and Portability of the database are met.

Free projects on this area include:

- SQLite - <http://www.sqlite.org/>
- TextDB - <http://textdb.sourceforge.net>
- Mimesis - <http://mimesis.110mb.com/>

- MySQL CSV engine - <http://dev.mysql.com/tech-resources/articles/csv-storage-engine.html>

When comparing each found option, the choice was made upon SQLite.

TextDB has little information or documentation on the site, not looking updated on a long while.

Mimesis is intended to function exclusively with PHP and our application would require running a PHP server on the client side.

MySQL CSV is an enterprise class project with a stable development but the CSV solution would still require running MySQL as a service on the client machine.

SQLite was chosen since it was designed for operating in serverless conditions and also possess an enterprise level support from relevant companies on the market.

Exposed issues

SQLite is open source without distribution limitations; it allows each client application to manage the data on disk. However, it is not appropriate to situations of high concurrency or server client applications where multiple clients write information back on the server.

A complete description of limitations on SQLite can be found at the following link: <http://www.sqlite.org/whentouse.html>

Data stored in database

On this chapter we define the structure for managing data that is be stored inside the database repositories. It is subdivided on the following sub-chapters:

- File record;
- File record size per field;
- Handling file record collisions;
- File index;
- Seals of trust;
- Correct, prejudicial and ignored files
- Configuration space.

File record

Each file requires a set of data fields that are gathered to ensure that it is unique and distinguishable from other files.

Below is a table detailing from a high-level perspective the records that will be retrieved from the records stored on the database.

Record	Purpose	Target files	Exception rules
Name	Distinguish files with different names.	Correct, prejudicial, cautious, ignored or conflict.	None.
Internal version	Distinguish files with the same name but different versions.	.dll; .exe; .scr; .ocx;	Returns null value on other extensions.
Security flag	Assigns a flag to each file that we assess as correct, prejudicial, cautious, conflict or to be ignored.	Correct, prejudicial, cautious, conflict or ignored.	None.
Size	Store the file size in bytes.	Correct, conflict and cautious.	Except ignored and prejudicial files.
Time stamp	File creation date and last access date.	Correct, conflict and cautious.	Except ignored and prejudicial files.
Checksum	A unique checksum of the file with resort to SHA-2	Correct, conflict and cautious.	Except ignored and prejudicial files.
Type of	Identify algorithm to allow	Correct, conflict and	Only present if

checksum	multiple checksums.	cautious.	checksum is added.
Index date	Date when the file was indexed on database.	Correct, prejudicial, cautious, conflict or ignored.	None.
Indexing cliens	Registers the name of the indexing cliens	Correct, prejudicial, cautious, conflict or ignored.	None.
Family	Clan or Castrum title where indexing cliens is integrated.	Correct, prejudicial, cautious, conflict or ignored.	None.
Binary signature	Adds a segment of bytes as binary signature.	Optional	Only one signature per file.
Additional comments	Comment from indexing cliens.	Optional	Only one comment per file.
Date of comment	Register when the comment was made.	Only if comment added	

Table DB-01

File record size per field

To keep track of the byte size used by each field of the record, we detail the maximum size that each field can use to store data.

Record	Field name	Type	Max size in bytes
Name	NAME	String	256
Internal version	VERSION	String	50
Security flag	SEC_FLAG	Byte	1
Size	SIZE	Int64	64
Time stamp	TIMESTAMP	Int64	64
Checksum	CHECKSUM	Int64	64
Type of checksum	T_CHECK	Byte	1
Index date	INDEX_DATE	Int64	1
Indexing cliens	ID_CLIENS	String	30
Family	ID_GENS	String	30
Binary signature	BIN_SIGNATURE	String	256
Additional comments	COMMENTS	String	256
Sum per record			1073

Table DB-02

Assuming the max allowable database size to be of 4Gb (4 294 967 296 bytes) and rounding the record size to an upper value 1100 bytes we'll be capable of storing around 3 904 515 records per database.

Handling file record collisions

A file record for a given file will eventually collide with other file records for the same file as the information flows higher in the system structure.

To handle these expected collisions, we define that each record is capable of holding information from different sources up to a predefined limit.

- How are collisions of same file records from different sources handled?
 - Answer: Locally, we assign a unique index number and while moving up the hierarchy of knowledge we merge the results.
- How are multiple comments by authors added to each file record? Are they needed?
 - Answer: Comments can be added and are merged onto a single entry that contains it's own format to allow managing each comment.
- Can these file comments be edited at a later time? By who?
- What happens if a file record is considered untrustworthy? Is it deleted?

File index

Knowing that we can store close to 4 million records per database, this value is still restricting the system to index more files in the future and worry about this upper level limit.

To overcome the limitation of a single database, we add support for indexing a file across a set of databases using a centralized database.

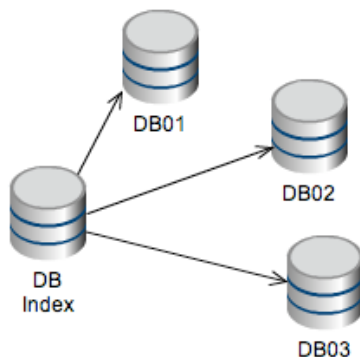




Diagram FI-01

Table DB-02

Element	Description
 DB Index	The master database that is used to index file records.
 DB03	The slave database that stores the full file record.
$A \rightarrow B$	A writes data to B

Pending questions:

- What format of data should be recorded on the DB index?
- What is the upper limit to the DB index – how many records can be physically stored?
- “Trust no one because no one trusts me” – how is this enforced for DB index? Who is ensuring that each DB index on the local host cannot not corrupted?
- If DB index is attacked, what measures are available to ensure that this attempt does not break the whole system? How are recovery/defense available to keep the index safe?

To allocate an index record, we specify on table DB-03 the fields that will be used

Record	Type	Max size in bytes
Name	String	256
Internal version	String	50
Security flag	Byte	1
Size	Int64	64
Time stamp	Int64	64
Checksum	Int64	64
Type of checksum	Byte	1
Index date	Int64	1
Indexing cliens	String	30
Family	String	30
Binary signature	String	256
Additional comments	String	256
Sum per record		1073

Table DB-03

Seals of trust

An anomaly is detected when we detect a mismatch between the information extracted from a file on disk and the information that is found on the database or when a prejudicial file is discovered.

In typical cases, the decision of declaring a specific file as correct or not will depend on the pool of information gathered from other cliens that is found stored on the knowledge database.

However, if the available information for a given file is reduced, there is the need to assign seals of trust to be stamped by actors with a certain authority within our system.

We trust on these actors to stamp files that they assess as correct, prejudicial or ignored.

Any file that mismatches the information stamped as correct or is identified as prejudicial will automatically flag an anomaly alert.

The roles inside our system with the power to assign a seal of trust are listed below by order of importance:

1. Architecti
2. Quaestor
3. Praetor / Patrician
4. Civis

A file with a seal of trust from a cliens will need to be verified by the praetor. If the praetor assigns his seal of trust to this file then the knowledge about the file is shared to the other cliens on the castrum.

The praetor can share the knowledge about this file with other castrums or clans on the Internet but the knowledge about this file in particular must be stamped by a quaestor before being added on the common knowledge databases.

A file with a seal of trust from an architecti will have predominance over the seal of trust passed by any other role beneath his own and is included directly on the common knowledge databases.

The accuracy of a trust seal can also be challenged in the case of false positives; the resolution of these conflicts is solved by the praetor/patrician at a lower level. If there is information mismatch between multiple clans/castrum then a quaestor should solve the conflict.

For the cases where two or more quaestors conflict on opinion about the stamp assigned to a file, then an architecti may resolve the question, or a public voting of quaestors can take place at the forums.

Ignored files

Although the intention of our system is to index and acquire knowledge of as many files as possible, there are files that cannot be indexed due to their constant modification.

A good example of this is the Pagefile.sys and Hiberfil.sys files that are used to cache the virtual memory for the MS Windows NT operative system.

It is not possible to index their checksum, filesize, version or any relevant detail besides the filename itself as all of these details will change at each MS Windows machine.

Files of this kind will be tagged with a “ignore” flag. Meaning that our knowledge databases will contain instructions to ignore them while processing the files on disk.

There is the risk that malware is eventually concealed inside these ignored files. To mitigate this risk, there is the option to move files from the list of ignored files and apply a signature-based pattern to identify malicious inside the file.

The ignore flag can for a given file can be set at all levels of authority starting at the cliens level except if someone in the upper level as explicitly included the file on the knowledge base.

If a lower level actor considers that an assigned tag at a file should be modified to “ignore”, it is possible to issue a request for the authority on the upper level to revise the file in question and decide if the tag should be modified or not.

File system

The clients application is running from a Java virtual machine and will make use the Java API to interact with the file system present on the workstation.

The use of a multiplatform development environment allows using a common implementation for most of the file operations that can be required by each role.

File Operations

This sub chapter details the available file operations that the clients application can perform on request from a given role.

- Create, erase, rename and move a single file or group of files;
- Change file permissions;
- Extract details from a file such as the time stamps of when the file was created and the last access;
- Extract the file size in bytes;
- Provide a list of available drives and the type of these drives (USB, CD-ROM, HDD and so forth);
- Identify whenever a drive is inserted or removed from the workstation;
- Calculate the checksum of a given file using a selected algorithm (SHA-2 is the only algorithm used for the initial project release);
- Operations on text files:
 - Add line or lines of text on a text file;
 - Remove lines matching a text portion.
- Operations on INI style files:
 - Read and write data in “Key=Value” format inside a specified INI section;
 - Read all text lines from an INI section;
 - Write a group text lines onto a INI section;
 - Create or delete INI section.
- Operations on binary files:
 - Find or replace a binary sequence inside a group of files;

- Monitor changes on target folders (for example, detect when new files are added);
- Identify system folders of the Operative System, in the case of Windows: report the correct location of “C:\Windows”, “C:\Users” and so forth if they differ from the default value.

Exposed issues

There are exceptions to the principle of a common implementation across platforms that we list below:

- Modification of file permissions requires running the native shell command “chmod” under Linux/Mac OS environment or “attrib” under Windows.
 - The issue is not the overhead of distinguishing between “chmod” or “attrib” but the absolute path used on the shell, there are increase risk of defects when handling folders with special characters such as “!” on folder names that need to be extensively handled and tested.
- Detection of drives available on the workstation is different for each platform and brings additional overhead to handle:
 - Mac OS and Linux: A folder representing the drive is mounted inside a specific root folder using the drive’s label to title the new folder:
 - Under Mac OS, drives are mounted in: “/Volumes”;
 - Under Linux in Ubuntu/Debian are mounted in “/media”;
 - Other Linux environments may use different root folder titles.
 - Under Windows each drive is assigned a letter, “d:”, “e:” up to a max of “z:”.
- Handling of situations when there is not enough available disk size to allow the cliens application to function as intended;
- Handling of slashes when using absolute paths under different platforms, for example, Windows uses “c:\Users\Test” while Unix uses “/home/Test”.

Pending details

This section attempts to concentrate some of the pending details and questions that have been raised during development that are incrementally addressed as the project progresses.

- We need to detail the triggers for the elections of role empowerment
 - How do we avoid malicious machines from being selected?
- What is the physical limit of the databases files in terms of storage?
 - How will databases be managed in case a limit is reached?
- Describe in more detail the information flow between cliens and triumvirs:
 - In cases of recoverability after a crash occurs
 - When there is no response from a triumvir
 - Detail the communication protocol used between them
- Conduct experiment to get a realistic view of the supported stress for triumvirs
- Describe situations where a bizantine type of decision is assumed.
- If we detect the same file with same version but patched voluntarily, isn't this a possible issue that should be addressed since it has a different checksum (mismatch) that needs to be handled
- How do you plan to see anomalies detected?
 - Which ones you will see
 - Which ones you will not see
- How can this system be best used?
 - What type of defense is intended?
- Better definition for seals of trust.
- Elections - are cliens voluntarily enrolled as candidates to become triumvirs?
- API concept needs to be better defined
 - Management Interface should be distinguished from API
 - Database in context diagram is too generic, applies to everything in life.
- Explore the alternative of using dryStone instead of loop count to obtain more accurate values on the cliens score.

References

[1] <http://en.wikipedia.org/wiki/Castra>

[2] <http://en.wikipedia.org/wiki/Clients>

[3] http://en.wikipedia.org/wiki/Tabula_rasa#Computer_science