SafeViewer

softwareunited by NOVENTIQ
Global expertise, local outcomes

Safe Viewer (PAM) Solution Documentation

1. Introduction

Purpose

This document is meant to serve as a detailed technical documentation for Safe Viewer (hereafter referred as "The solution") – the all-in-one PAM platform dedicated to provide a secure and controlled remote access to endpoints by tunneling various remote access protocols through an unique, simple, and efficient Web interface, delivering a full administrative control and forensics over undergoing remote sessions. Safe Viewer provides an unique ability to have a full control over every remote session aspect, especially forensics, scaling down very advance functionalities to an elegant, user-friendly way, thus ensuring a total and simple operation while keeping security in place.

2. Key Components

Administrative dashboard

An interface where an absolute platform monitoring takes place. Administrators can easily follow up with statistics, logged on users, admins, supervisors, active remote sessions, useful data graphs, etc.

User Management

A panel where all of user management takes place. Operations such as addition, deletion, blocking, resetting a password, changing email, and other crucial actions, including an overview of all users' information - at one place.

Administrator & Supervisor Management

As for users, there is a completely separate management console that provides a complete and on-point actions over administrators and supervisors. An administrator account is the most-privileged on the platform. They all have permissions over each other and of the same level. Supervisors represent bodies with read-only privileges over monitoring (supervising) functionalities, such as: remote session recordings, user activity log, reports, and certain statistics. Therefore, supervisors represent a perfect choice for people in SOCs (Security Operation Centers).

Remote Servers' Management

An intuitive and simple remote server configuration panel is one of the key ingredients for a successful PAM. With Safe Viewer, there is a way to easily establish a remote server connection parameters in just a couple of steps, and later modify it, if needed. It is possible to establish multiple remote configurations for the same server (different username, password, port, etc..). Additionally, there is a way to decide whether or not file transfer is allowed, or not, as well as an ability to only permit one way transport (upload / download).

Finally, administrators assign such servers to appropriate users on the platform itself – this is how users get an ability to connect to such remote targets. With that being said, only the servers that get associated with specific users can be seen by those users on the user side of the platform. Therefore, users get no ability, whatsoever, to see, modify and interact with the remote server configurations that administrators previously assigned. Users can only read that server configuration structs and choose to connect accordingly.

File Manager / Quarantine

Our platform possesses a special feature that allows administrators to intercept and analyze files that previously get downloaded by users within remote sessions. Such files get displayed on this panel for further overview and management.

Furthermore, administrators have an ability to enable file quarantine feature on a remote server level. With a quarantine enabled, no file is going to be available for an endpoint user without previously being checked by an admin, and approved. An administrator, in such a case, has an option to discard / reject a file if it is considered bad, malicious, or other. Upon rejection, there is a always a space for a comment so that the endpoint user knows what caused the rejection. It is worth noticing that the endpoint user gets notified upon file approval.

Quarantined files can be filtered out per a server session, or user, allowing for an easy and efficient file analysis. This case scenario is especially convenient in situations where multiple files get downloaded within one remote server session.

File Transfer

A unique, reliable, and secure way to deal with a file transfer throughout different remote control protocols, such as SSH, RDP..? Our platform provides just that. Regardless of a protocol in use, Safe Viewer allows you to establish a secure, reliable, and integrity-based file transfer channel which, not only guarantees great performances and stability while transferring large files, but supports compression and parallel downloads / uploads.

On top of that, we equipped Safe Viewer with a file integrity pre-download check. In fact, each file possesses a unique signature which can be used to identity that file among millions. Therefore, upon file download, Safe Viewer stores the signature for further comparison purposes; such that when a new file is about to be downloaded from from a remote server, the system checks for an already existing file on the system instead of re-downloading and overwriting the same file, as other similar systems typically do.

In order for the above to take place, we introduced file transfer certificates which are used to authenticate to target remote systems. Certificates are a typical SSH asymmetric cryptography private/public key pair, whose public components get available to administrators via the management panel, while private components stay under a dynamic double-layer encryption on the platform itself and never get exposed.

Furthermore, Safe Viewer utilizes and communicates with a special kind of a kernel's key- chain which gets initialized and destroyed with the platform. Therefore, private keys are treated with an ultimate security in mind, and yet, users' experience stays absolutely unaffected. Additionally, there are explicit options to lock (de-allocate certs from the key-chain) and unlock (load certs into the key-chain), for an administrator's convenience. Have in mind that by locking a key-chain, file transfer is disabled globally, and vise-versa.

Independently, file transfer can be easily toggled on a global level by a simple switch on top of this section in the Administrative panel. That way, it is easy to disable any file transfer from happening on Safe Viewer, no matter the reason, if that's what it is necessary.

Installation of a certificate is pretty simple. An administrator must put a certain public key component into appropriate **authorized_keys file** (typically located under **.ssh** directory) so that Safe Viewer can access it and initialize a file transfer session. With that being said, in a case of a RDP remote server in mind, it will also be necessary to setup a SSH server on a target machine and link a corresponding file transfer session after a RDP connection settings.

There are two types of file transfer certificates: **global** and **user-specific. Global** cert is meant to be used widely throughout remote servers, for which a root / admin account is recommended so that Safe Viewer could establish a file transfer for multiple users with the same certificate. On the other hand, a **user-specific** certificate overrides a global cert for a specific user. Therefore, it could be a great alternative for all remote servers that require a special, strict security approach. In fact, the certificate would be generated for a dedicated user only and the platform would only utilize this certificate (even with a global cert installed) for accessing remote servers upon this user's sessions.

Due to stability reasons, thus compatibility with the key-chain working principle, the number of file transfer certificates is limited to 10 at the time. The reason being is that this is a default number of allowed authentication attempts supported by a ssh daemon. After the threshold is reached, no more certificates can be issued without previously revoking some of the old ones. For a threshold, the total number of global and user-specific certs is taken into a count.

Single Sign On (SSO)

Another very significant functionality that makes the login process a way easier to grasp. Safe Viewer allows you to setup and link this authentication method with any standardized SSO client that can provide you with basic SSO configuration params (Issuer, Client ID, and a Client Secret), such as Microsoft, Google, Okta, and others.

Upon Addition, the successfully configured SSO providers will became available as new login options on a login screen.

Remote Sessions and Recordings

Safe Viewer provides a detail, comprehensive, and yet elegant and intuitive way to organize and manage remote server sessions. Each session is recorded and user's actions monitored and logged to ensure integrity and security. Administrators have an ability to follow up with all session data, including status, duration, start time, access reasons, and other important parameters.

Administrators, yet have an ability to interact, to a certain extent, with live remote server sessions and preforming important actions on them. For an instance, an administrator can decide to terminate (kill) a remote session due to certain malicious activities performed by a user, or any other reason, without elaborating on it, after which, the admin can jump on the session forensics, more in depth.

An important point to mention is that Safe Viewer uses a unique way to store session recordings in an secure, efficient, and storage friendly way. In fact, recordings are encoded in a text format and further processed as such. This approach was taken for a couple of reasons:

1. It is storage efficient and more sustainable.
2. It is quick and convenient to apply a Military-Grade Encryption over it.
3. Recording can be instantaneously analyzed for malicious activities and processed for further forensic purposes.
4. Recordings can be built in the background, on-demand of an administrator / supervisor and compressed back in the same manner for great performance.

Recordings have status, indicating their built state. Each remote server session, when completed, is automatically sent for a build. Therefore, a video format is going to quickly be available upon a session completion. It is important to mention, that recordings are kept safe within Safe Viewer and cannot be downloaded away from it.

On users' side of the platform, it is indicated that each session is being recorded as soon as a session starts. Also, when a user is about to start a remote session, it is required to provide an access reason and also, for certain situations, an underlying explanation (message) of an entry to the server. Later, administrators can have a better overview of why the user accessed the server in the first place.

Activity Log

A crucial security aspect of a Safe Viewer is definitely a great, concise, and comprehensive activity log. Each and every action that is either raised by a user or an admin is logged. Not only an event's data, but as well, metadata is available for any kind of forensics. Those actions can include basics, such as login, register, password reset, to something on a higher end, like building a session recording, or killing / terminating a remote session. Safe Viewer logs everything.

Logs can be more in depth filtered, sorted, and grouped, in order for the best results. Everything can be looked-up for in activity logs, including usernames, event names, a connection protocol, user access messages, and other important search parameters.

Reports

Another crucial forensics tool. Safe Viewer provides reports of a great depth which serve as a great way to follow up with what's happening on the platform. There are a couple main report types which can be managed through this Administrative panel section and, on top of that, their creation process can be scheduled in advance to facilitate staying updated.

Our reports are an all-in-one solution for keeping up with Safe Viewer's data. A couple of example data sections are: access log, user login, user statistics, server session analysis, and other.

3. Architecture & Network

Safe Viewer system infrastructure is consisted out of 2 virtual servers – back-end and front-end; each requiring its own separate VM and a static WAN IP address. The machines are logically separated in order to provide for the best performance and security. Back-end server possesses all the key components needed to establish the proper operation. With that being said, there is another key ingredient in the play - a Docker container.

The docker container represents a logical middleware for translating high-level instructions sent back and forth between remote endpoints and the back-end, allowing for a better performance and connection stability, while letting the applicative part of the solution free and ready for further processing operation. The middleware is a self-sustainable, low-impact, automatically built unit which perfectly corresponds with the rest of the platform by predefined protocols.

Other important point to mention is that, by default, the database is located on the same VM as with the back-end server. Of course, there is an option to relocate the DB server, if considered necessary, with an adequate change applied to the server configuration file.

Front-end unit is purposefully separated from the back-end with the main reason being a system stability. In fact, the remote server sessions are conducted solely between clients' browsers and the back-end. With said being said, once the web content is fetched from the front-end server and cached inside of a browser, there is no need for a client to contact this server again, which allows a client to become a standalone entity and communicates directly and securely with the back-end server. However, once a new web component is needed, the front-end server comes back into the play, serving the client.

As a conclusion, there is no direct communication conducted between the two main Safe Viewer servers (back-end & front-end), primarily for security and stability reasons. Therefore, 2 static WAN IP addresses are needed, 1 for each server, so that clients / users can contact the front-end server directly and fetch the web content in need. From that point forward, the communication is conducted between a client's browser and the back-end directly.

Safe Viewer' core is a perfect combo of low to high end programming languages, starting from NodeJS framework, all to way down to Bash and C, being the main actors in a fulfillment of important kernel operations. Angular framework, on the other hand, is used to pack a complex and flexible front-end

modules into a secure web package release to be served to a browser. The source code is tightly secured with advanced crypto methodologies and packed in a way to prevent any misuse, whereas, on the other hand, is able to deliver great operation performance.

When it comes to network, both back-end and front-end servers communication through HTTP protocol, sitting behind a Nginx virtual reverse-proxy. The virtual proxy middleware is introduced for a sake of an extra layer of security and network reliability. Safe Viewer needs an SSL certificate to be installed for a successful operation. The platform is able to automatically ensure itself with a SSL certificate (Let's Encrypt) by the time of an installation, if needed. HTTPS is an only accepted way of communication on Safe Viewer's behalf, therefore Safe Viewer does **NOT** guarantee a successful and integrity-based operation if used otherwise.

In a case that Safe Viewer is hosted behind WAF (Web Application Firewall), is it alright to configure the virtual servers to accept HTTP traffic only and host a SSL certificate globally on the WAF itself, configuring Safe Viewer servers as WAF nodes. Have in mind that, in such a case, the virtual servers need to be introduced to a proper DNS / sub-domain setting, as well as the proper ports.

Furthermore, the platform supports any kind of a HTTP proxy server, if Internet access is not directly available. These settings need to be and can be easily applied in the back-end's configuration file.

For further clarification, if Safe Viewer is the entity responsible for issuing and hosting a SSL certificate, both Web ports, 80 / TCP and 443 / TCP need to be open for incoming as well as an outgoing traffic. This configuration applies for both back-end and front-end servers. If a case where a WAF is involved, HTTP port should be defined through the WA

5. Installation and Setup

5.1 Prerequisites

Safe Viewer system architecture is meant to be installed and run on a dedicated Linux-based virtual machines as an on-premise solution. A preferred operating system is, at this point (June 2023), **Ubuntu 22.04 LTS.** The installation requires a superuser privileges and an active, stable Internet connection. The installation process the completed automatically via predefined generated setup scripts. The setup scripts, on the other hand, are totally unique per client, so that all the randomly generated passphrases and identifiers stay secret – even to us. The scripts are very advance and, therefore, are capable of completing every aspect of the setup process, including the final touch.

Below, provided are the minimum to recommended requirements needed for Safe Viewer to operator properly and achieve maximum performances:

	BACKEND	FRONTEND
CPU	~ 2Ghz (4-8 core)	~ 2Ghz (4 core)
RAM	8 GB	4 GB
SSD	512 GB	25 GB
SSD1	Upon needs	/

5.2 Installation Steps

Once the prerequisites are met, the installation procedure takes place. As mentioned previously, the setup is automatic and is initialized by executing an appropriate setup script, for the adequate server. For the script to run in an expected manner, the proper environment needs to be provided. Furthermore, each and every Safe Viewer element is meant to run as a dedicated user – which can be created at this time by, also, running the prescribed script. Safe Viewer servers themselves should not be run as root user due to security reasons.

The setup script is going to require some important information to be known ahead of time. Therefore, installation should not be started without ensuring that this requirement is fulfilled. For an instance, domains, desired Web protocols, and other carrier parts could be a part of it. Everything else is taken care by the automated script.

Afterwards, once all important elements of the platform are installed safe and sound, there are a couple of post-installation actions that will take place for a sake of a better security practices. Safe Viewer automatically instantiates a TDE encryption for a MySQL database in use. Furthermore, DB encryption-at-rest becomes another security advantage for the complete Safe Viewer ecosystem. This protection mechanism plays an important role of storing the data in a safe way, by which only authorized channels could access it.

On top of that, the previously mentioned scripts for issuing an SSL certificate (if not behind WAF) are to be executed at this point.

Finally, after the setup process is completed on both servers, SSL certificates installed, and other custom setup completed, Safe Viewer is ready to shine.

6. Security and Compliance

Safe Viewer goes along with modern, world-wide accepted security policies and regulatory standards about digital security, data storage, authentication procedures, and other directives. The platform's data underlies top level security approach on all levels and processing phases. User data is encrypted in transit and at rest. During data processing phases, only the minimum amount of required information is fetched and parsed in order to establish the proper operation. Authentication data is accessed in a very strict manner in such that even Safe Viewer servers do not posses the clear ability to read it. This approach mitigates all chances against malicious data use.

Additionally, Two factor authentication (2FA) is a step each and every user / admin entity must undergo in order to register an account. User secrets, access tokens, and other sensitive information is kept under strict encryption protocols, whereas, for some on them even a multilayer encryption is introduced.

Regarding remote sessions, Safe Viewer features an ability to setup and configure remote users' passwords and usernames for a quicker remote server login. Those information are treated as a top secret – they are stored in a way which is inaccessible to every entity expect for the endpoint remote server, to which it is passed.

On the other hand, it is important to mention that database undergoes a TDE (Transparent data encryption) mechanism which makes the data readable to authenticated entities only, on a connection level. Therefore, any unauthorized connection attempt to the database will result in a failure. Bu default, the database server is only available locally and does not expose ports on the dedicated virtual machine, thus, a remote malicious connection attempt is impossible.

When it comes to remote server session recordings – encrypted with a Military grade standard (AES). Advanced Encryption Standard, when used properly, provides a great security mechanism for symmetrically encrypting and storing sensitive data. It is fast, reliable, and, most importantly, secure.

On top of that, each and every sensitive information / file is hashed. With that being said, the information's integrity, in other words – its cryptographic signature, is checked previously upon an inserting / storage. Later, when the data is needed again, its signature is verified against the previously saved one. Therefore, Safe Viewer recognizes any kind of a malicious attempt of data manipulation / corruption and aborts the current process.

6.1 Security Features

Cyber Security is Safe Viewer's foundation. Every feature / functionality is written with this concept in mind. Safe Viewer takes care of security while providing space and time for a successful yet easy privileged access management to take place. Some of the key security concepts are:

- Encrypted and tunneled remote session management
- Military Grade Encrypted remote session data
- Crypto managed, integrity-based file transfer
- File quarantine analyzer
- Fully equipped Administrative panel for an ultimate control
- 2-Factor authentication (2FA)
- Comprehensive action log
- Automatic security report generation
- Malicious user activity detection & prevention
- Multi-layer data encryption mechanisms
- Automatic updates through secure channels

7. Maintenance and Support

7.1 Software Updates

Safe Viewer supports automatic updates which are conducted solely through secure proprietary channels, specifically designed to allow for great performance and stability. An important point to mention is that Safe Viewer servers do not required a listening port so to be able to execute updates. In fact, Safe Viewer knows to check for and "pick up" the latest release, once available. Therefore, the updates can securely be conducted within more strict infrastructural network environments due to the fact that the same protocols (SSL) which allow the platform to perform a normal operation are used to fetch and install updates.

Updates can be forced to a manual procedure, by specifying the adequate flag in the server configuration file. Thus, if a specific version of the platform is wanted, it could easily be kept. However, it is important to have in mind that missing important updates is not recommended and can cause greater security risks.

By default, Safe Viewer check for latest releases on a daily basis, through midnight hours. Therefore, if a later release is to be installed, it takes a matter of minutes to update and install, and by working hours, the platform is easily ready to perform to its best extent. Just a note that, for a successful update, there must be a stable Internet connection provided.

Safe Viewer virtual servers posses a mechanism to sync the latest version among themselves. Therefore, updates are integrity based. Each server, upon a successfully conducted update, informs other instances so that they can proceed with the update locally, as well. Once all the virtual servers update, the platform reboots, applies appropriate changes, and continues with a normal operations.

Troubleshooting

Of course, as with every platform, there could exist a bug, a glitch, or other technical problems or operation misunderstanding that have not been previously detected by developers or a testing team. Those are very rare, however – they might happen. Therefore, if such a case, it is encouraging, and sometimes, necessary to contact the support team, if the problem is considered unsolvable on the spot.

For a sake of better understanding and solving a problem or a bug, it is really helpful to collect as many evidences as possible on a client side. Therefore, please be advised that upon reporting such scenarios, the developers' team might ask you to further elaborate on the problem so it is approached in the best possible way, thus – the solution becomes available sooner.

Have in mind that the Safe Viewer team is open to hearing new ideas, suggestions, or critics at any point. Therefore, please do not hesitate to participate in a development process by providing a helpful input.

Support Contacts

Main support team: **support@fiksni-online.rs**