KEEPING YOUR SECURITY MONITORING TOOLS AND TASKS SAFELY SEPARATED

INTRODUCTION

Security tools and appliances typically run with high privileges, have access to sensitive traffic and their security has to be taken at face value. Luckily zero trust, containment, least privilege and privilege separation are not just trendy topics and security design principles for ordinary software. They are also practical ways to extend your defence in depth to security tools themselves and to meet regulatory requirements such as personal data protection.

Security monitoring is not a task for a one man band. Different tools have the best edge in intrusion detection, vulnerability scanning and asset monitoring. We developed SensorFleet to let you run multiple security tools with ease, each best fit for the task at hand. Each SensorFleet Sensor can run multiple tools, which we call Instruments.

One of the primary design goals for us was to keep these Instruments safely separated. This means that while allowing the Instruments to interconnect with each other when necessary, Instruments should not be able to interfere with each other or with the Sensor platform itself. Furthermore, Instruments should have as little access to the network as it is necessary for them to function. For example, passive Instruments should only be able to receive network traffic, they should not be able to send any packets to the protected network or call home without your explicit permission.

Instruments should be able to store persistent data on Sensor platform, but they should not be able to access data stored by other Instruments or any data of the Sensor itself. While isolated, the Instruments should still be able to provide user interface or API endpoints. These should be available through the Fleet Management.

Next we will go through how this isolation has been implemented in SensorFleet platform.

INSTRUMENTS WITHIN SENSOR

Benefit of having a common Sensor platform is that you can run multiple Instruments within the same Sensor. These Instruments can do different tasks from running a honeypot to hosting an IDS to check the traffic.

It is vital to keep Instruments contained and not allow them to interfere with each other or the Sensor itself. This way, a malfunctioning Instrument can not bring down or compromise the whole platform and/or other Instruments running on it. If there is a security breach or supply chain risk in one of the Instruments, the attacker should be contained in the affected Instrument and will not be able to leverage other Instruments or the platform to harm you.

CONTAINMENT

The primary way to separate Instruments is to run them in different containers. SensorFleet platform uses Linux Containers (LXD - https://linuxcontainers.org/lxd/introduction/) Each Instrument is run on its own LXD container. This will separate the processes of each instrument from other Instruments, while they still run on the same Operating System kernel. Containers are more lightweight than virtual machines, and still provide the separation for processes and network access. Within Instrument containers we run tasks with as little privileges as possible. When designing Instruments for SensorFleet, we keep them as simple as possible. For example, we use the Alpine Linux (https://alpinelinux.org/) as default base image as it has a small footprint and does not have unnecessary services by default. This allows us to build the Instruments to contain only necessary components and makes it easier to verify their operation.

CONTROLLED NETWORK ACCESS

Instruments have different kinds of networking requirements. Some instruments, like IDS or Netflow collector need to be able to read network traffic, but do not need to send data. These kinds of instruments are called passive and can be configured to have a mirror-bridge network interface which will only receive mirrored traffic from the actual physical interface. This lets us stop passive Instruments from sending traffic when they err or get compromised.

For Instruments requiring network access, like honeypot Instrument, users can either dedicate physical network interface available on platform or create bridged interface to physical interface shared with multiple Instruments. This allows the user to control which network segments Instruments have access to and the degree of access.

Further, if Instrument needs to periodically download resources from remote location (IDS rules, for example), but does not need read-write access to network otherwise, the Sensor platform provides helpers which can be used to download these resources from a safe egress point that is fully under your control. Instruments will only have the absolutely minimum network access they really need for their core operation.

CONTROLLED STORAGE
Instruments often need to store data. Our platform provides instruments with two different storages, transient and persistent. Neither of these can be accessed by other instruments, disallowing other Instruments from reading the possibly sensitive information stored. The filesystems are encrypted at rest.

Each instrument has its own time-based retention policy and is responsible for deleting any data that is older than set retention policy.

Transient data
Transient storage is initialized when Instrument is started and deleted when it is stopped or removed. It can be used by the Instrument to store runtime data that does not need to be stored after Instrument is no longer running.

Persistent data
Persistent data can be used by the Instrument to store databases or any other artifacts. This data is kept after the Instrument has stopped. Each Instrument is required to make sure persistent data does not contain data older than its data retention policy.

SUPERVISED INTERCONNECTIONS AND ORCHESTRATION
Since Instruments can not connect with each other using network or shared filesystem, the platform provides Instruments a communication bus where they can either send direct messages to each other or provide events which other Instruments can act upon to react or augment.

Message bus enables creating manager instruments which can operate, or command, other Instruments. One example of this is an Instrument which can create rules for IDS. The IDS Instrument itself does not need to have all the application logic or UI to allow creation of IDS rules, a rule manager can handle that and use the message bus to send rulesets to the IDS.

MESSAGES
Messages are either sent from the Sensor platform or other Instrument to one Instrument to command or configure it. Messages are only seen by the intended recipient and permissions can be configured to specifically allow messaging between specific Instruments.

EVENTS
Instruments can also produce events. Some of these events are directly aimed for users (IDS alerts, for example), but some events are aimed for other Instruments to trigger some action.

For example, using DNS events produced by IDS instrument whenever name resolution is detected in protected network allows Passive DNS Instrument to collect passive DNS database without actually needing access to network or do any of the complex DNS message parsing itself.

HELPER INSTRUMENTS
Sensor provides some Instruments whose purpose is to provide operations for other Instruments. These Instruments allow to reduce privileges from Instruments doing network packet analysis or other parsing of non-trusted data. These also make auditing higher privilege operations easier due to reduced scope.

DOWNLOADER
Downloader Instrument can be configured to periodically download resources from the network. The download operation is isolated to a single Instrument with network access. This network can then be isolated from the actual protected network. Downloader can be used to fetch IDS rule updates from providers periodically, but the actual IDS Instrument or even the rule manager does not need to have access to the Internet. Furthermore, this also reduces code duplication since the download functionality does not need to be implemented on all the Instruments.

Downloader will download resources to its persistent storage and then use Sensor platform services to distribute the data to other Instruments. This way, the downloader Instrument itself does not need access to other Instruments storage. This way compromised downloader still does not provide access data of other Instruments running on platform.

contact@sensorfleet.com

Cyber security sensor solution

Open. Fast. Scalable. Transparent.

CAPTURE ENGINE INSTRUMENT

Capture Engine Instrument provides network packet access to passive Instruments. Any Instrument doing packet analysis can be configured with a read-only bridge interface to which Capture Engine Instrument then mirrors packets from actual physical interface connected to protected network.

Capture Engine Instrument is designed to be efficient in copying and does not do any packet analysis itself, reducing the attack surface as the packet analysis is normally the crucial part and should be run with as low priorities as possible.

INSTRUMENT UI

Some instruments can provide their own UI or HTTP API for users. This access is provided by Fleet Management UI and can not be accessed directly through other channels. All HTTP access is reverse proxied by Fleet Management UI and thus can be access-controlled on the Sensor if needed. This allows the Sensor platform to have a single point through which all user interfaces and APIs are accessed.

IN A NUTSHELL

Who watches the watchmen dilemma haunts the security industry. However solid design principles such as containment and least privileges can be successfully applied to security tools and monitoring tasks. We have designed SensorFleet for containment and user control both on level of access and data retention. We believe that these straightforward and transparent design principles will help our users to stay safe and few steps ahead of the game.

contact@sensorfleet.com