**Git Hub Link**: <https://github.com/S-Spencer-Bellevue/HackBrightCapstone>

**CAPSTONE PROJECT – HackBright Academy**

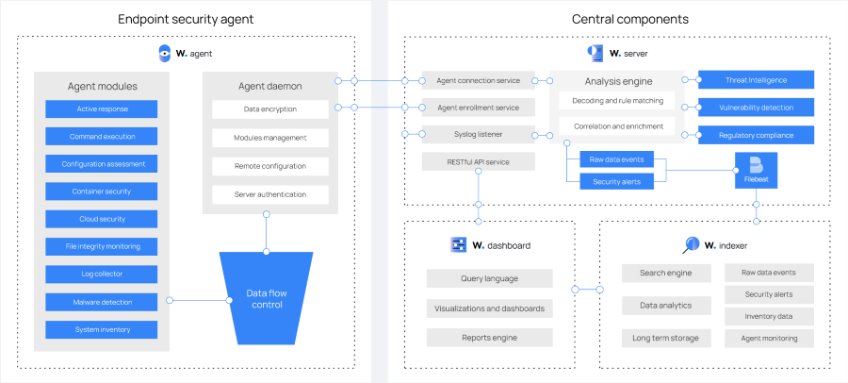
**Goal:**

Implement and utilize Wazuh (SIEM/XDR) in a home lab environment to monitor, manage, alert upon, and scan agent systems with threat intelligence integrations.

**INVESTIGATION**:

1. Research Wazuh software (Use, Setup, Documentation, Network Setup, Persistence)

The Wazuh software is available at <https://wazuh.com> for Windows, Linux, MacOS, Unix, and several other variants of each kind. It is an opensource SIEM/XDR solution that can be used with local environments, or cloud- based environments through an API interface. The address <https://documentation.wazuh.com> was used for in depth to first read about the software and its usage.



Research Integrations

By default, Wazuh utilizes the OSSEC system for management and underlying infrastructure. It follows the ELK stack for integrations and allows for additional third-party integrations including VirusTotal and Maltiverse along with the ability to add custom integrations via the open-source RESTful   
 API endpoint and the JSON interpreter. Documentation on some common   
 types of integrations can be found at <https://documentation.wazuh.com/current/user-manual/api/index.html> .   
 Aditionally, further community support can be found via the forums, Wazuh   
 email group, or the r/azuh subreddit. For this home lab initial set up, I   
 elected to implement the virusTotal and Maltiverse API integrations.

1. Research Rulesets/usage

Wazuh allows for a default number of rulesets pre-configured and based on the OSSEC framework along with the requirements for PCI-DSS, GDPR, HIPAA, NIST 800-53, and MITRE ATT&CK. These rule sets can be customized via the ossec.conf file on either the Wazuh Server or each individual user- agent. Several key rules that are pre-configured include specific registry key modifications, application and security files, and the ability to include more that are included, but disabled by default to prevent overloads in logging. It is key to note that the log files that Wazuh can access are hindered by any security policies or lack thereof, such as not logging activities on the user- agent. If there are no captured logs, there is nothing to report. Within the ossec.conf file, the pre-configured rule sets can be enabled or disabled via setting the <enable>yes/no</enable> or <disable>yes/no</disable> options. Through documentation and further implementation types for custom rulesets and detections are available via a quick internet search or via the <https://documentation.wazuh.com> website.

1. Research Raspberry Pi setup, and manager/agent setup

To gain further experience with different systems and integrations, I elected to purchase, build, configure, and install a Raspberry Pi to satisfy my curiosity regarding the RaspOS and its default security mechanisms as well as challenge myself with utilizing unknown systems and software for evaluation and future integration.

1. Set up VM for Kali, Windows- Install Wazuh manager/agent

For this home lab, I elected to utilize a segmented network consisting of a VM for Kali, Windows 10 client, Windows 11 client, and the RaspOS system with the Kali VM machine hosting the Wazuh Server, Indexer, and Dashboard. Other selections could have been made for the Wazuh installation OS, but I chose to use a system that I could quickly make portable with a reliable OS that contained built in tools for potential future alerting use.

**PLANNING**:

1. Gather Documentation on Wazuh Installations

Utilizing the well-organized documentation available via the Wazuh site as noted above, I discovered information relating to the installation methods based on the known supported operating systems, as well as manual methods for issues with direct package installation. The documentation serves as an excellent starting point for server, indexer, dashboard, and user-agent installation. Additional information is provided on integrations, setting rules, and understanding how the system works. I utilized this information in addition to the community support section to gather methods and actions to take for the various types of installations included within this home lab environment.

1. Gather Raspberry Pi Documentation on Set up

The Raspberry Pi 5, if purchased as a kit, does include a quick start guide for setting up the OS with power requirements and an overview of the RaspOS system. I opted to purchase housing, a cooling fan and heat sinks for active cooling, and a preinstalled micro-SD card for running the operating system. Additional documentation was collected during the implementation phase to address issues as they arose and research solutions as is often necessary when attempting to connect and set up distinctly different open-source systems. I found information about issue resolution via common community locations as mentioned above and internet searches for known issues.

1. Gather/Create rule sets for detection

By default, Wazuh does include some rulesets as part of the configuration file in both the server and user-agent locations. These rules are primarily based around the implementation of security settings, the speed and limiting of network activity, log transfers, and size, and known locations of interest to malicious threats or users. The Wazuh community also provides additional rules that could be useful such as the monitoring of specific public files and other critical changes. These rulesets can be configured to simply alert the user, either via the dashboard or through set up email alerting, or may take action as an IDS/IPS to delete or record changes to potentially harmful files. The who-data ability, for example, allows for the Wazuh administrator to identify the process and user under which an event occurred; this is not a default setting. Additionally, changes can be configured within the event logs that will show precisely the information changed in critical files, such as a static file that should not be changed often. Monitoring many files or entire directories in this nature, especially in an active network will lead to degradation of the network available as well as quickly fill storage space. It is recommended that this type of rule be limited to very specific files.

1. Implement detection rulesets – Plan

I chose to explore the current default rulesets in planning including the non- standard file integrity monitoring and threat hunting rulesets that were disabled by default. Additionally, I researched adding additional rulesets to integrate with VirusTotal or Maltiverse. I found that the community support forums and the subreddit contained vast amounts of information to assist with configuring and adding custom rulesets. Many of these rulesets were geared towards an enterprise environment and did not fit the current use- case for my home lab, though I have bookmarked many of these for creating an enterprise version of my home lab to further my experience with large scale environments.

1. Integrate Virus Total – Plan

I chose to utilize VirusTotal as a method of integration to expand the current default environment and add additional community support in identifying malicious files or known trust issues with websites, IP addresses, and other common indicators. This plan as a free level is rate limited but provides enough checking for this environment daily to complete this lab as well as for future use as I intended.

1. Integrate Maltiverse – Plan

I chose to utilize Maltiverse to further expand and demonstrate the integrations of outside systems for threat intelligence as Maltiverse provides a growing list of known IoC’s and the API allows for comparison between file changes and these IoC’s. This is rate limited at the current community level, and I chose to utilize this as the next level up is an enterprise version that costs nearly a thousand dollars to deploy and was beyond the scope of this project.

1. Scan/set baseline for agents (PI, Kali, Windows, network) - Plan

I plan to utilize the PCI DSS, NIST 800-53, and GDPR reporting metrics to analyze my current lab environment for baselines. This lets me see how a default home setup compares to what is considered compliant for common enterprise use. I fully expect that it will be far from adequate, but it allows me to further my lab in my future free time to address issues on specific systems. My end goal in the future, outside of this lab setup, is to use a well- ranked system to attempt penetration testing and vulnerability analysis to identify further issues that can correlate to an enterprise environment.

1. Generate data (attempt login, generate reports, address/remediate/accept risk on vulnerabilities) - Plan

In this portion, I intend to generate reports based on the events that commonly occur on the network via normal usage. I will be browsing the internet, downloading files or programs, and updating systems to generate data that can be compiled into reports. Further, I will look at what the system currently reports as risk, including any CVE issues to either mitigate or accept the risk associated with continued usage. If available and within the scope of the initial lab setup, I will work to remediate high and critical level issues, provided it does not hinder my future plans of deliberately pen testing active vulnerabilities within these systems.

**IMPLEMENTATION:**

1. Built Raspberry Pi 5, Installed RaspOS on MicroSD. Configured to connect to the local network (Debian)

To achieve this goal, I purchased a Raspberry Pi 5 with 8GBs of RAM from the Vilross website, an official retailer of Raspberry Pi devices and accessories. As this is primarily a one board system, it included common connectors for modern system control. To address heating issues, I purchased an active and passive cooling solution. This included six heatsinks for each of the main chips on the board, easily installed by size and with the included heat transfer adhesive. For active cooling, I attached an RGB fan, powered by the board, to further control the heat generated by active use. This solution has kept the board cool and stable without large temperature fluctuations or the activation of the built-in temperature throttling system designed to prevent damage to the board.

I purchased a 128GB microSD card, as I wanted additional space beyond the required 8GBs, for usage as a storage device and operating system bootable. The Raspberry Pi organization provides an imager that I utilized to install the Raspberry Pi OS onto the microSD. This imager can be downloaded at the Raspberry Pi website at <https://raspberrypi.com/software/> . The installation took considerable time and forced erasing the entire microSD to complete. When this was complete, I installed the microSD into the slot at the bottom of the Raspberry Pi and booted the system. Installation of the RaspOS was quick as it is built upon a version of Debian Linux. It is important to note that the Raspberry Pi utilizes an ARM architecture which is different than most other common types of architecture and thus requires different installation methods or installers than what is generally used.

The setup of the RaspOS is similar to most other operating systems as it  prompts for internet connectivity, time zone setup, and account setup. The software does require immediate updating as it does contain vulnerabilities right out of the box, some of which are quite severe. As the goal of this project was to create my own home lab, I found that the Wazuh user-agent was not easy to install, and after several fresh wipes of the microSD, I was able to get the user-agent to install correctly. The Wazuh documentation at [https://documentation.wazuh.com/current/installation-guide/wazuh-](https://documentation.wazuh.com/current/installation-guide/wazuh-agent/wazuh-agent-package-linux.html) agent/wazuh-agent-package-linux.html provides a method of installation via the command “*c****url -s https://packages.wazuh.com/key/GPG-KEY-*** ***WAZUH | gpg --no-default-keyring --keyring gnupg-*** ***ring:/usr/share/keyrings/wazuh.gpg --import && chmod 644***  ***/usr/share/keyrings/wazuh.gpg****”*  but I needed to alter the basic instructions using the apt installation method to include the wazuh IP address and reinstall/force remove the keyring multiple times to take effect.

The RaspOS did not like the installation of the keyring for the SSL certificates and generated a number of errors as it viewed this method as insecure. However, it can be forced to accept the keyring with fully switching into the root user as sudo would not work correctly for this part. Next, I added the repository with the command *“****echo "deb [signed-*** ***by=/usr/share/keyrings/wazuh.gpg]***  ***https://packages.wazuh.com/4.x/apt/ stable main" | tee -a***  ***/etc/apt/sources.list.d/wazuh.list****”.* Finally, I updated the system via the “***apt-get update***” command and installed the Wazuh user-agent using “***WAZUH\_MANAGER="IP ADDRESS OF MANAGER" apt-get install wazuh-*** ***agent***" . It is important to note that the service daemon will need to be started and restarted anytime that the configuration file has changed, or the user-agent is installed for the first time. This can be accomplished with the  “*s****ystemctl daemon-reload***

***systemctl enable wazuh-agent***

***systemctl start wazuh-agent*** “commands. Once installed, the user-agent maintains persistence by adding the startup to the startup tasks when the system is rebooted.

To configure the network, I had to first get the Wazuh server, indexer, and dashboard working properly, which is detailed in section three below, and then alter the configuration file located at /var/ossec/etc/ossec.conf. This is where it is possible to control what information and logs are sent to the indexer for processing via the server and the RESTful API. In this lab, I added the VirusTotal and Maltiverse API integrations utilizing the code: "

**<integration>**

**<name>virustotal</name>**

**<api\_key>API\_KEY</api\_key> <!-- Replace with your**  **VirusTotal API key -->**

**<group>syscheck</group>**

**<alert\_format>json</alert\_format>**

**</integration>**

The integration of Maltiverse is the same except replacing with the appropriate API key. I utilized my current API keys for these integrations, though anyone can sign up at the websites <https://virustotal.com> and <https://whatis.maltiverse.com> . Once signed up, an API key will be generated and should be secured from external users. Even though the API key is being hardcoded into the configuration file, the entire directory is owned by the root user so regular users without root permissions cannot access these files or directories.

To configure the network, I had utilized my router/gateway portal to implement a segmented network, a /26 VLAN, to place this home lab within. This process is different for every user, typically only involves a few clicks or selections to create and the subnet mask is often auto generated based on the selections.

1. Utilized fresh install of Windows 10, updated Windows 11 system.

To expand my home lab with the use of several different systems, I utilized a fresh installation of Windows 10. The system contained no additional software other than what is included as the default. I utilized the Windows Updater to Update the system to be as current as possible. This effort took several hours and several restarts to take effect. Additionally, I updated the Windows 11 machine that was hosting the Kali VM machine, which is running the Wazuh Server, Indexer, and Dashboard. This part of the lab took nearly five hours, though the time is relevant to the current update level of the systems being used. While it was not necessary to fully update the systems to utilize Wazuh, I did want to have the latest updates as mentioned above to set a baseline of how Windows and other OS’s are configured which gives an indication of the base level of compliance that I could expect from a fresh installation in a larger or enterprise environment that does not already have a golden image in place. Updates on Windows systems are easily accomplished by simply clicking on the update button under the Windows Update setting if not already set to auto download. For the Kali system, ***sudo***  ***apt update*** and ***sudo apt upgrade*** were sufficient to update the current packages and systems to the most recent 2024 version.

1. Reviewed Wazuh requirements, verified that Server system (Linux) had enough resources as a VM to run. Mitigated multiple space issues, installed dependencies of C++ 2019 Redistributable, and Python3. Updated and upgraded all packages. Installed Wazuh Indexer (Database), Wazuh Server (Opened ports 1515, 1514, 50500 (for registration with SSL certificate generation), Wazuh Manager (Dashboard). Configured firewall and gateway settings to allow for port access and network segmentation at system and gateway router levels.

Per the documentation that I had gathered in earlier stages, I reviewed the Wazuh requirements for my home lab build, which increase with the number of user-agents or shards that are implemented. With this setup, at least 4GBs of RAM, 4 CPUs, and 80GBs of storage space is required to run the system. While the minimum does work, it can be a bit slow in response to compute heavy tasks such as the Wazuh Indexer retrieval of all logged events or other similar actions. My initial setup did not provide enough resources to effectively run the system, likely due to it being in a VM itself, so I altered the resources to find a balance point that worked effectively. The Linux systems required dependencies such as C++ 2019 Redistributable and Python3 to effectively log the required files and be able to communicate as intended. This command “***apt-get update && apt-get install python3***” allows for the installation of python and a simple search engine assisted with determining how to get the c++ dependency included. The Wazuh server installation included these packages so it is not necessary to perform this step on the system that is running the server.

As mentioned in the prior portion, I had updated and upgraded all packages on the Kali system that was hosting my Wazuh Server, Indexer, and Dashboard. The Wazuh system utilizes FileBeat as a method of forwarding and centralizing the logs which are pushed to the Indexer for processing and storage. The Wazuh server provides the middle functionality of pushing requests between the Indexer and the Dashboard as well as controlling the SSL certificates that are generated for identification of the user-agent during registration. This effort prevents unknown systems from attempting to join the Wazuh system or manipulate the integrity of the log files due to forcing a DoS on memory or space constrained systems.

To facilitate the Wazuh system’s effectiveness, I utilized the system controls to open ports 1515, 1514 (for communication) and 55000 (for user-agent registration) on the VLAN network.

The documentation provided at <https://documentation.wazuh.com/current/installation-guide/> assisted with installing the indexer, server, and dashboard in concurrent order. As expected, this installation requires root access and the indexer should be deployed first as it is the means of communication and password generation to secure the server and dashboard. Downloading the installation and configuration file was accomplished with the commands “***curl -sO***  ***https://packages.wazuh.com/4.8/wazuh-install.sh***

***curl -sO*** [***https://packages.wazuh.com/4.8/config.yml***](https://packages.wazuh.com/4.8/config.yml)” . Once this occurs, the configuration YAML file must be updated to reflect the node names and IP values of each node. In this lab, I only had one node which was running the entirety of the environment so only the first option had to be altered under nodes -> ip in which the IP address of the system to be running the server and indexer was altered. For more diverse systems, such as distributed systems using the cloud, a secondary (or more) node may be altered to input the gateway address for connection. This file update MUST be completed on the Wazuh Server, Indexer, and Dashboard in order to properly reflect the available systems and ensure connectivity between each user-agent and the server.

Once this YAML file has been updated, the Wazuh server can be installed. Of note when setting up the Wazuh server, the storage must be adequate to address most log file transmissions and may require significantly more with custom integrations, additional file monitoring beyond the default, and API integrations. With the default selection of file checking every 24 hours, a system such as this lab, with three user-agents, plus the default fourth user- agent on the server system can utilize nearly 10GBs every 90 days. Additional resource requirements can be ascertained by monitoring the /var/ossec/var/run/wazuh-analysisd.state and the events\_dropped option to determine if events (logs) are being dropped due to lack of resources. The command “***curl -sO*** [***https://packages.wazuh.com/4.8/wazuh-install.sh***](https://packages.wazuh.com/4.8/wazuh-install.sh)” is used to start the download and “***bash wazuh-install.sh --wazuh-server***  ***wazuh-1***" installs the server. As before, the YAML file must be edited with the node and IP address as described above.

Finally, the Wazuh Dashboard can be installed similar to above with the commands “***curl -sO*** [***https://packages.wazuh.com/4.8/wazuh-install.sh***](https://packages.wazuh.com/4.8/wazuh-install.sh)” and “***bash wazuh-install.sh --wazuh-dashboard dashboard***”. Wazuh Dashboard utilizes the port 443 on the local system for access and once  
 installed, the information summary will direct the administrator to the appropriate port as it may also choose 8443, 8444, 8080, 8888, or 9000. It is **CRITICAL** that the administrator record the username and password that the server generates as it is not possible to gain entry or edit the password if it gets lost or forgotten, as I discovered when I was forced to complete this entire installation process a second time as the system no longer accepted the password that I had previously saved. Additionally, if the system sits idle for a long period of time, particularly on a VM system, it will require a reboot to accept the correct password. Some of these issues may be remediated by installing the indexer and server systems using Docker or Deployment on Kubernetes. Additionally, pre-built VMs for Amazon Machine Images are available, but I chose to perform all actions personally to gain the experience and understand both what was being installed and how these systems interoperate leading to a much better personal learning environment. Many common errors that may be generated during this installation and setup process can be resolved via the community forums and the reddit page including missing dependencies or component failures.

1. Utilized documentation and community support to install user agents on Windows 10, Windows 11, Kali, Debian systems. Utilized agent management to enroll devices. Updated configuration files to allow for file integrity monitoring, threat hunting, and integration with VirusTotal and Maltiverse.

The Windows user-agents are easy to set up via the Wazuh Windows Installer available at [https://documentation.wazuh.com/current/installation-](https://documentation.wazuh.com/current/installation-guide/wazuh-agent/) guide/wazuh-agent/ . As expected with a windows system tool, once installed, a GUI system is available to set the IP address, start and stop the services, and update the configuration file. If the Wazuh server service is currently up, a few clicks on the configuration file and service button will result in the generation of an SSL certificate and registration on the Wazuh Dashboard. Additionally, if this method is not available to a client system due to restrictions, the Wazuh Dashboard allows for a CLI registration via a configurable installation with the add agent button. This button will generate results to add the specified client and can select Windows, Linux, Unix and more. I found this portion of the system to not fully work as intended and instead manually installed all user-agents on each machine. The Linux user- agent does not have a GUI version and I utilized the steps mentioned in a prior section above to install the keyring for the SSL certificates and to install the wazuh agent. Once operational, several minutes are needed as the system works to pull current logs and generate an analysis of the environment. I then updated each user agent’s configuration file to match the configuration set by the server including the enablement of file integrity monitoring and the subsequent integration with VirusTotal and Maltiverse as detailed in a prior section.

1. Scanned systems to generate current environment reports to be used compared overtime to analyze changes.

Finally, with the user-agents in place, the server, dashboard, and indexer all running, I began to scan each system to generate reports on the current environment as detailed below.

**TESTING AND DOCUMENTATION**:

1. Verified system operability through placement on the segmented network and dashboard alerting/enrollment.

After configuring each user-agent to utilize the intended target server, I placed each user-agent system within the VLAN segmented via MAC address allow listing. While outside users may be able to change their MAC address to meet the requirements of the white-listed MAC addresses on the segmented network, they would first need to be able to read the network packets on the segmented network which may be possible, but improbable in the current situation. I verified the connectivity with the server by alternating the starting and stopping of services to detect when a user-agent becomes disconnected. This effect is very noticeable on the Overview page of the Wazuh Dashboard which lists every user-agent registered and the status whether active, disconnected, or never seen.

1. Explored the functionality and adaptability of reporting and dashboards for individual agents or overall with all agents.

Using the documentation available via the Wazuh website and through personal exploration of the Dashboard functionality, I was able to quickly determine how to use the reporting functions at an environment and individual user-agent level. It is possible to generate custom reports that are exported as PDF files and includes nicely formatted graphs and charts to identify trends and notable events that have occurred. Additionally, each user-agent can also have a baseline set with the PCI DSS, HIPAA, NIST 800- 53, or GDPR requirements with only a few clicks of selecting the user-agent from the Overview page, or selecting one of the reporting criteria, then selecting the user-agent button in the top right of the page to select specific user-agents. This reporting functionality shows the current percentage in relation to meeting all criteria and highlights those areas where criteria are missing. It allows for Indepth understanding by outside links with information relating to each requirement and steps to take to improve the environment.

Further drilldown can be commenced with the selection of either the Vulnerability Analysis or Threat Intelligence buttons on the lower right of the Overview screen. This also allows for individual user-agent selection and can be sorted by current CVE issues, including the exact issues as identified via the system log analysis that the Indexer utilizes or tracked via event ids which match common ids within systems such as Windows Login or service restarts. Further customization can be achieved with custom dashboard options to create views required by an administrator or business leader looking to track the system and its integrity. System views can be configured with the option dropdown in the center of the screen to select intervals of 15 minutes, 30 minutes, 1hr, 24hr, 7 days, 30 days, and all time. Caution should be used with lower resourced systems when selecting more than 7 days and attempting to view all events on all user-agents. This action may take significant time to process or cause the system to hang indefinitely. Of keynote is the ability to filter the events and logs based on Critical, High, Medium, and Low rule alerting. These alerting levels can be further controlled within the configuration file if the current rules are deemed too lax or restrictive.

1. Utilized monitoring to track changes in alert severity and analyze logs for all critical and high severity alerts. Alerts are generated based on MITRE attack framework and integrated into the Wazuh (Elasticstack) framework.

I ran several tests throughout this lab completion period after the baseline was conducted to check changes over time from normal use. I noted that the RaspOS had the worst base configuration with significantly low scores in the 30% range along with numerous alerts for potentially vulnerable files on the system. These files were included with the default installation and research did not prove that these files were of particular issue. This could be a false positive from the interactions between differing open-source systems. MITRE ATT&CK noted that the key-gen files may be trojanized which was likely due to them being packed and obscured from analysis. The MITRE ATT&CK framework provides excellent intelligence on potential abuse and TTPs as evidenced in the Presentation screenshot section below. Integration was trivial as the option to utilize this framework is a simple configuration file change and is turned on by default. The default setups, fully patched, of the Windows systems faired only slightly better with scores in the 50% though many issues can be alleviated with changes in the local group administration or via gpedit.msc or secpol.msc. These will be further configured as the project is intended to be ongoing for a home lab environment.

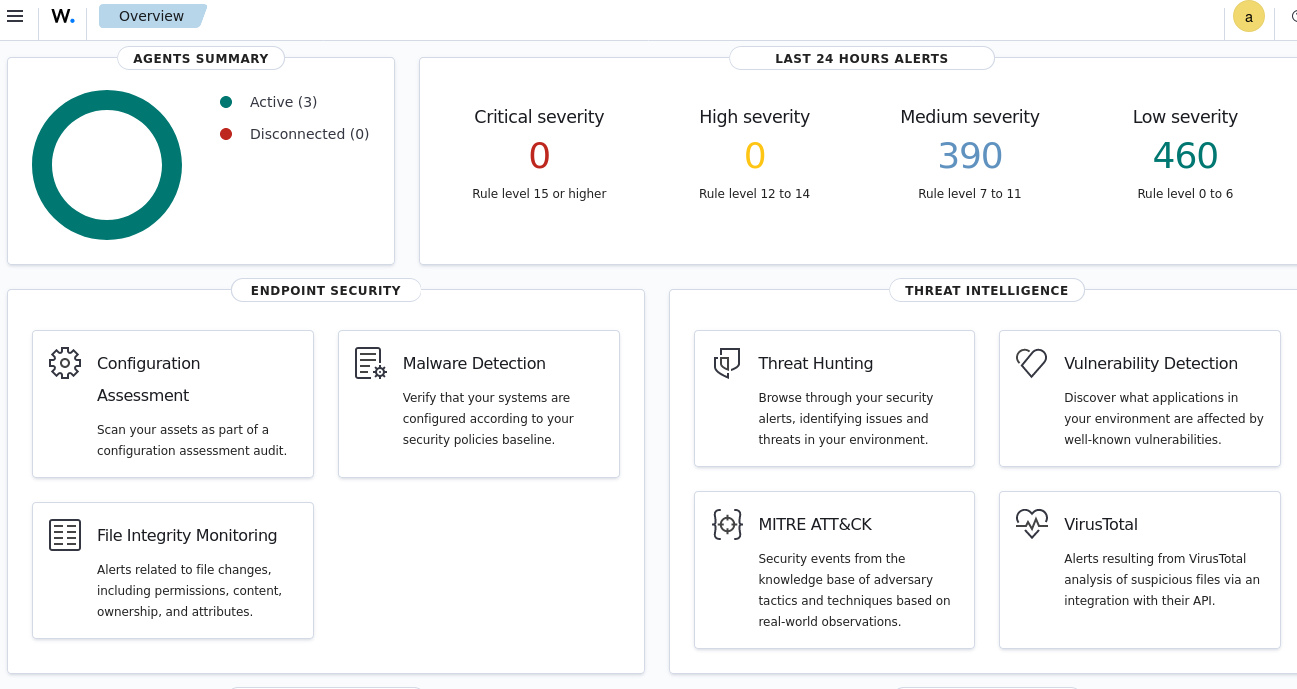
1. Ran system configuration for agents against current PCIDSS, GDPR, NIST 800-53.

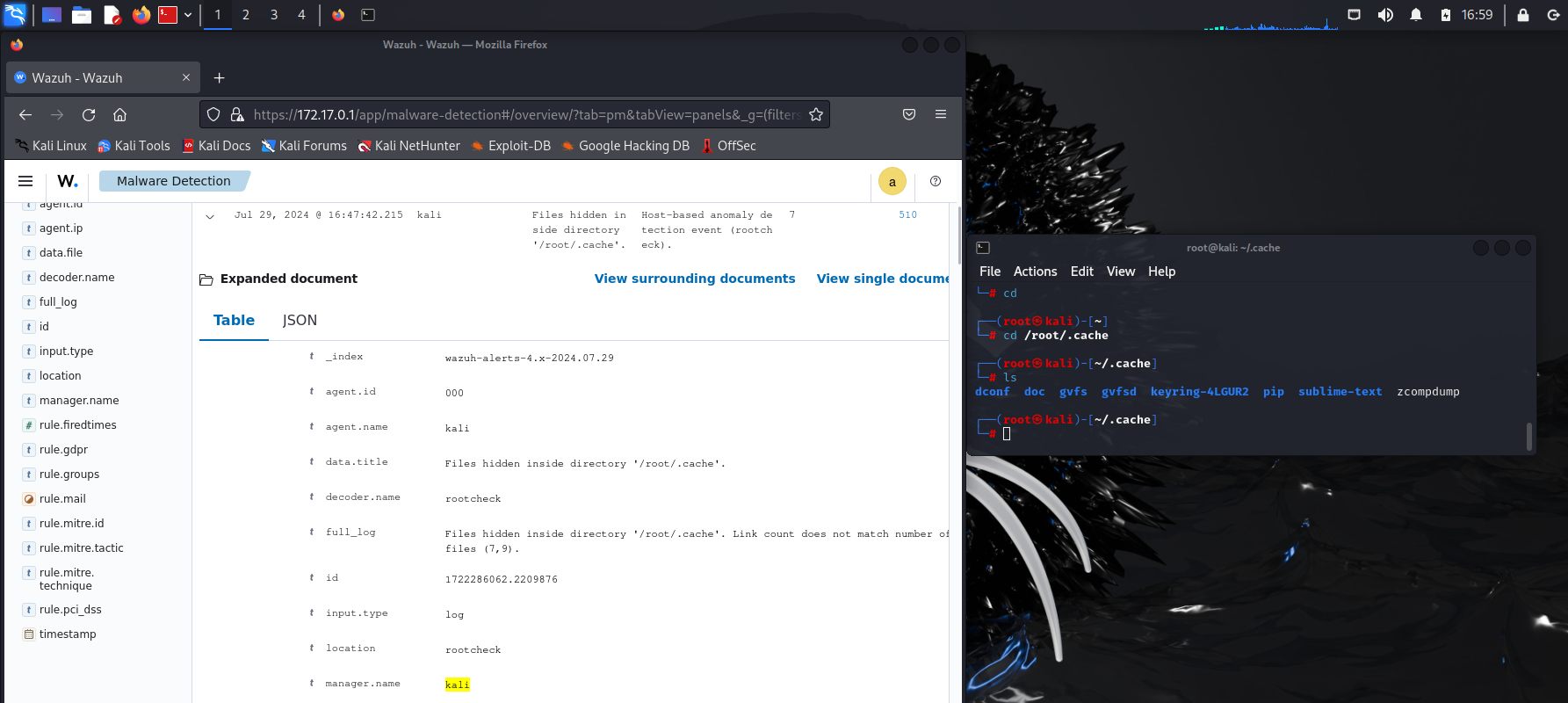
As noted above, I utilized the requirements analyzers for each of the included frameworks, standards, and legislation to address and discover insecure options and perform vulnerability analysis. I opted to remove only high and critical issues that were occurring which will be different for each individual’s setup, but in this case were remediated with clicking the link on each to learn what was the requirement and how to implement them such as not running in root user or flagging high for running out of resources. These were easily remedied.

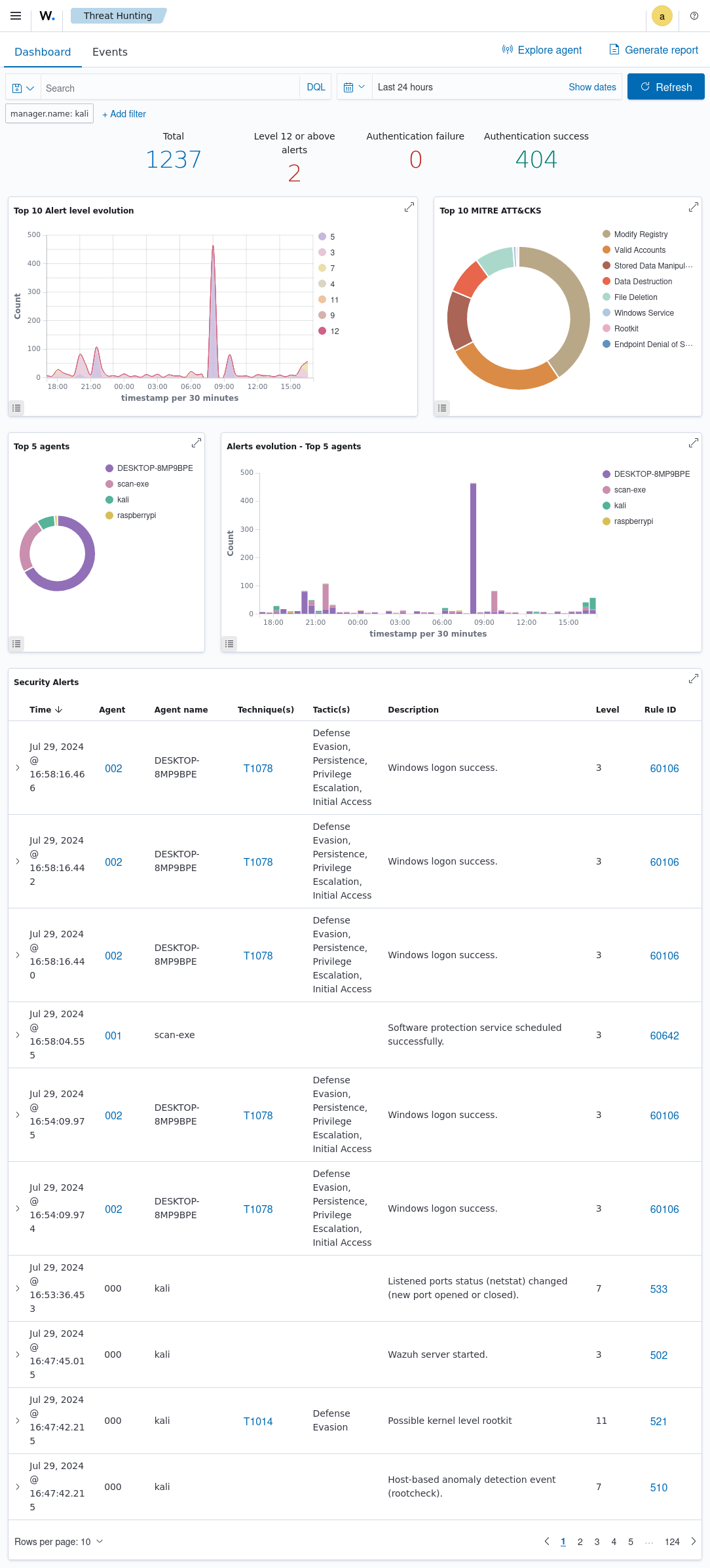
* 1. The noted agents were mostly silent, with the exception of software changes as implemented by myself, such as installation of VirtualBox on the DESKTOP agent and file changes on the Kali system which can all be further tuned as required.
  2. Utilized changes to generate alerts via the DESKTOP system agent (Login attempts, file changes).
     1. These changes included setting up an email alert via the Wazuh Dashboard to generate an alert when a critical or high event was discovered. As I control this environment and it is segmented with little traffic going in or out, I can quickly deduce if a change occurred due to my interaction, automation such as updates, or attempts by an outside threat (perhaps even myself when testing tools such as Atomic Red Team, nmap scans, and other tools available via Kali).

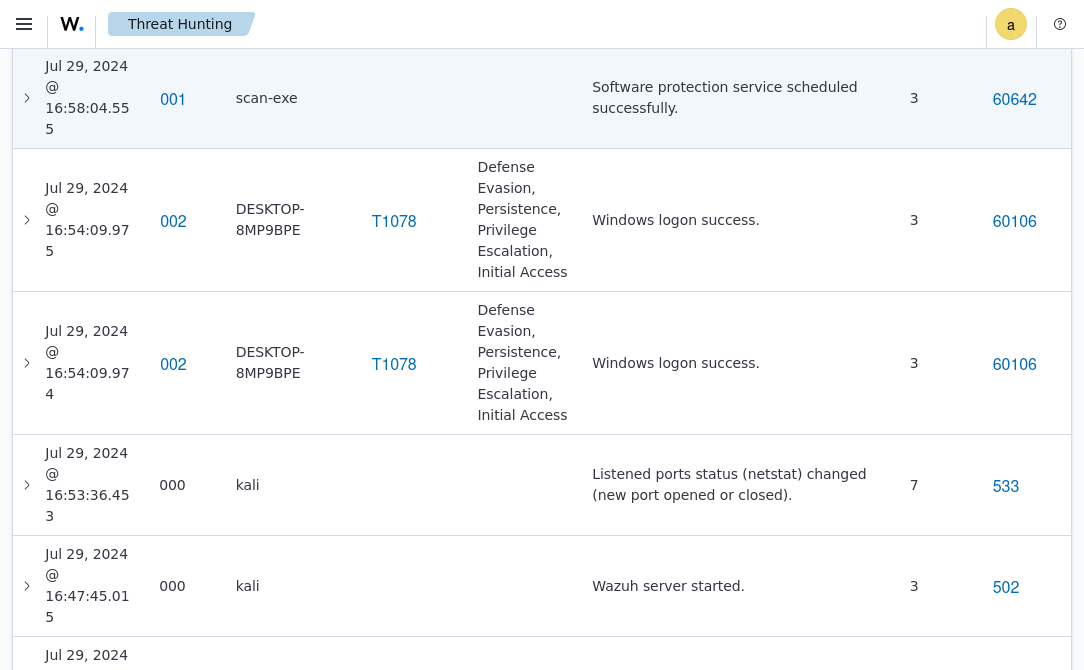
**PRESENTATION:**

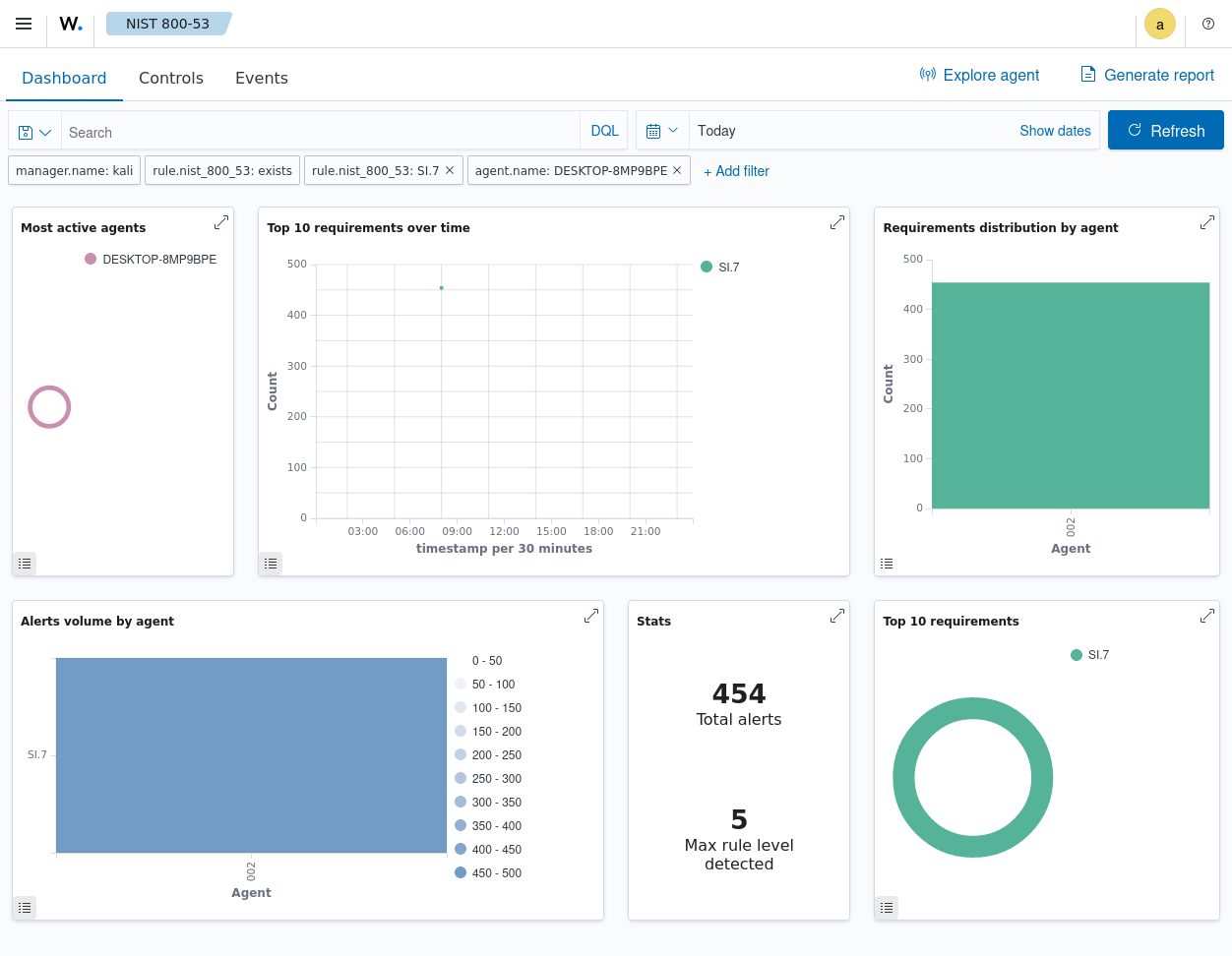
Included below are screenshots to demonstrate some of the activity that I encountered including user-agent scans against PCI DSS and how the MITRE ATT&CK system will map event ID’s to TTPs. This still requires evaluation as events that occur may be deliberate or manual as seen in my service restarts and system logons below.



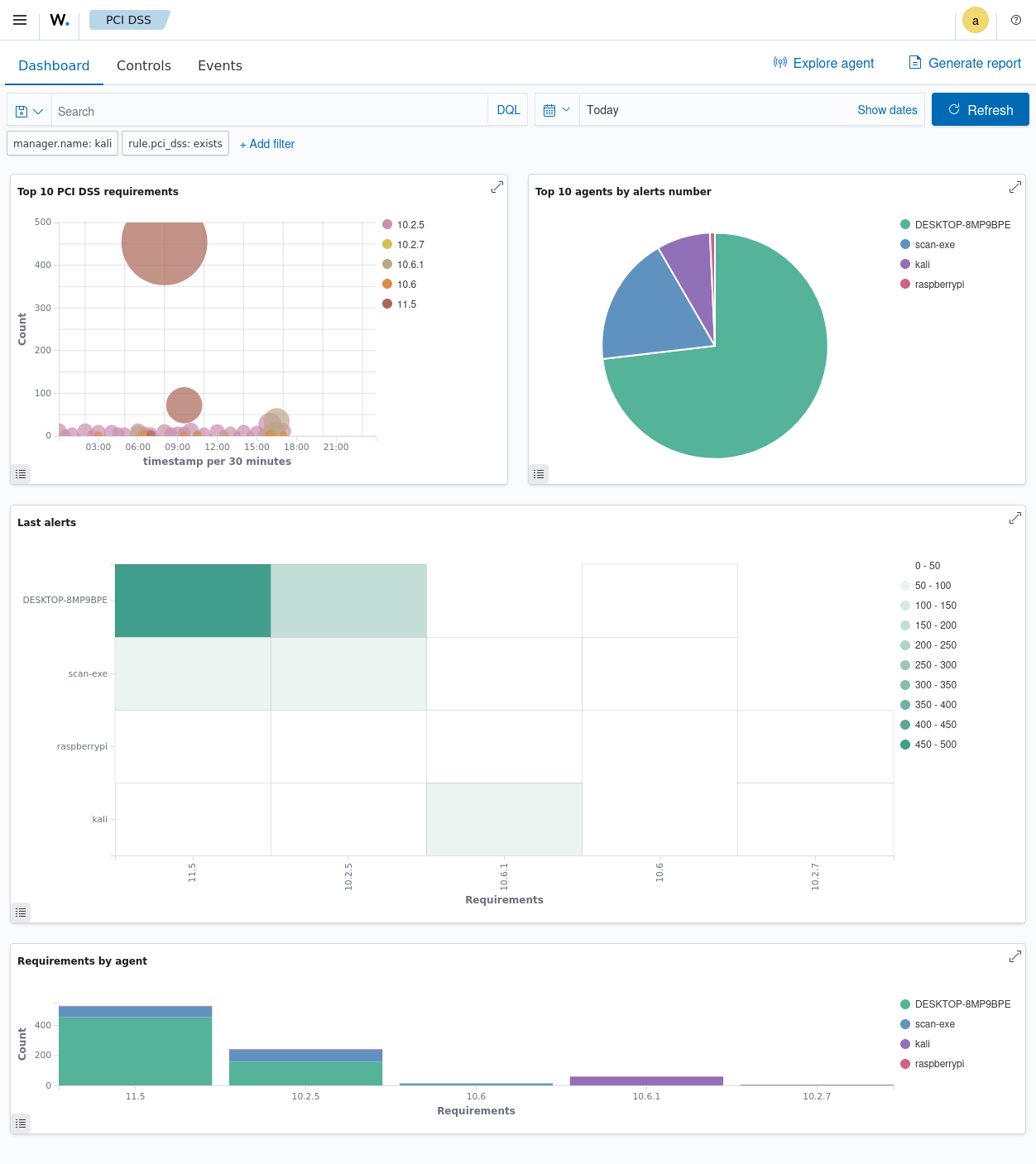








PCI DSS



Wazuh System Diagram – documentation.wazuh.com

