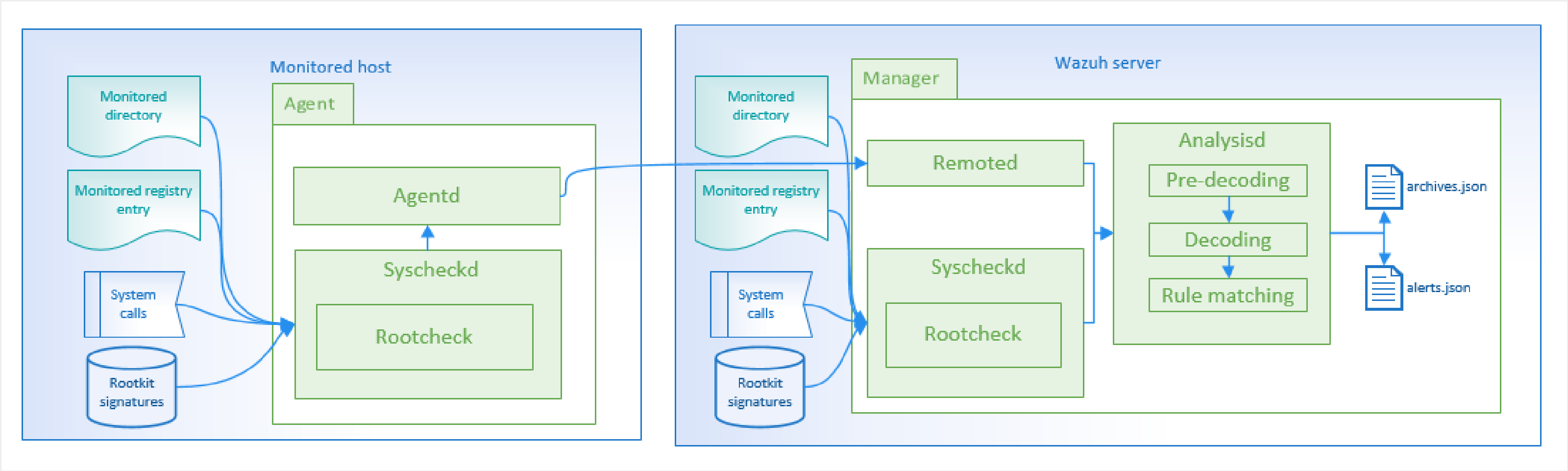
How it works

This section describes the checks performed by Wazuh to find the anomalies caused by an intruder or malware.



File integrity monitoring

Malware replaces files, directories and commands, so performing file integrity checking on the main directories allows us to detect these actions. More info File Integrity Monitoring Section

Example:

```
** Alert 1460948255.25442: mail  - ossec,syscheck,pci_dss_11.5,
2016 Apr 17 19:57:35 (ubuntu) 10.0.0.144->syscheck
Rule: 550 (level 7) -> 'Integrity checksum changed.'
Integrity checksum changed for: '/test/hello'
Size changed from '12' to '17'
Old md5sum was: 'e59ff97941044f85df5297e1c302d260'
New md5sum is : '7947eba5d9cc58d440fb06912e302949'
Old sha1sum was: '648a6a6ffffdaa0badb23b8baf90b6168dd16b3a'
New sha1sum is : '379b74ac9b2d2b09ff6ad7fa876c79f914a755e1'
```

Check running processes

A malicious process can prevent itself from being seen in a system's list of processes (trojan version of *ps* command). Rootcheck inspects all process IDs (PID) looking for discrepancies with different system calls (getsid, getpgid).

Example:

Diamorphine is a kernel-mode rootkit able to hide itself from *ps* and also able to hide other processes. If we install this package and hide a process, we will get an alert like this:

```
** Alert 1460225922.841535: mail  - ossec,rootcheck
2017 Feb 15 10:00:42 (localhost) 192.168.1.240->rootcheck
Rule: 510 (level 7) -> 'Host-based anomaly detection event (rootcheck).'
Process '495' hidden from /proc. Possible kernel level rootkit.
```

Check hidden ports

Malware can use use hidden ports to communicate with the attacker. Rootcheck checks every port in the system using *bind()* and if it is not possible to bind to a port and it is not in the *netstat* output, a malware could be using that port.

Check unusual files and permissions

Scan the entire file system looking for unusual files and permissions. Files owned by root with write permissions for other user accounts, suid files, hidden directories, and files are all inspected.

Check hidden files using system calls

Scan the entire system comparing the differences between the *stat size* and the file size when using the *fopen* + *read* calls. The number of nodes in each directory is also compared with the output of *opendir* + *readdir*. If any results do not match, you might have malware installed.

Alert Example:

```
** Alert 1460225922.51190: mail  - ossec,rootcheck
2017 Feb 15 10:30:42 (localhost) 192.168.1.240->rootcheck
Rule: 510 (level 7) -> 'Host-based anomaly detection event (rootcheck).'
Files hidden inside directory '/etc'. Link count does not match number of files (128,129)
```

Scan the */dev* directory

The */dev* directory should only contain device-specific files. Any additional file should be inspected because malware uses this partition to hide files.

Example:

If you create a hidden file on /dev, Wazuh should alert because there is a hidden file in a directory that should only contain device-specific files. This is the alert generated in that case:

```
** Alert 1487182293.37491: - ossec,rootcheck,
2017 Feb 15 10:11:33 localhost->rootcheck
Rule: 510 (level 7) -> 'Host-based anomaly detection event (rootcheck).'
File '/dev/.hiddenfile' present on /dev. Possible hidden file.
title: File present on /dev.
file: /dev/.hiddenfile
```

Scan network interfaces

Scan for any network interfaces on the system with *promiscuous mode* enabled. If the interface is in *promiscuous mode*, the output of the *ifconfig* command will show that. If not, we might have a malware installed.

Rootkit checks

Rootcheck performs several checks using its own database of rootkit signatures: *rootkit_files.txt*, *rootkit_trojans.txt* and *win_malware_rcl.txt*. Unfortunately, the signatures are out of date.