all info pulled from https://documentation.wazuh.com/current/getting-started/index.html

Table Of Contents

Wazuh-Agent

Configuration File

(More Info)

Linux	**/var/ossec/etc/ossec.conf**
Windows	C:\Program Files (x86)\ossec-agent\ossec.conf

Add to Settings between the <ossec_config> tags to monitor file.log

```
<localfile>
  <location>/<FILE_PATH>/file.log</location>
  <log_format>syslog</log_format>
</localfile>
```

Log Formats

(More Info)

syslog	Used for plain text files in a syslog-like format.	
json	Used for single-line JSON files and allows for customized labels to be added to JSON events.	

	See also the tag label for more information.	
snort-full	Used for Snort's full-output format.	
squid	Used for squid logs.	
eventlog	Used for the classic Microsoft Windows event log format.	
eventchannel	Used for Microsoft Windows event logs, gets the events in JSON format. Monitors every channel specified in the configuration file and shows every field included in it. This can be used to monitor standard "Windows" event logs and "Application and Services" logs.	
macos	Used for macOS ULS logs, gets the logs in syslog format. Monitors all the logs that match the query filter. See How to collect macOS ULS logs.	
journald	Required to monitor systemd-journal events. Events are collected in syslog format.	
audit	Used for events from Auditd. This format chains consecutive logs with the same ID into a single event.	
mysql_log	Used for `MySQL` logs, however, this value does not support multi-line logs.	
postgresql_log	Used for `PostgreSQL` logs, however, this value does not support multi-line logs.	
nmapg	Used for monitoring files conforming to the grep-able output from `nmap`.	
iis	Used for `iis` (Windows Web Server) logs.	
command	Used to read the output from the command (as run by root) specified by the command tag. Each line of output is treated as a separate log.	
full_command	Used to read the output from the command (as run by root) specified by the command tag. The entire output will be treated as a single log item.	
djb-multilog	Used to read files in the format produced by the multi-log service logger in daemon tools.	

multi-line	Used to monitor applications that log multiple lines per event. The number of lines must be consistent in order to use this value. The number of lines in each log entry must be specified following the `multi-line:` value. Each line will be combined with the previous lines until all lines are gathered which means there may be multiple timestamps in the final event. The format for this value is: <log_format>multi-line: NUMBER</log_format>	
multi-line-regex	Used to monitor applications that log variable amount lines with variable length per event. The behavior depends on multiline_regex option.	

Warning

- Agents will ignore `command` and `full_command` log sources unless they have `logcollector.remote_commands=1` set in their **/var/ossec/etc/internal_options.conf** or **/var/ossec/etc/local_internal_options.conf** file. This is a security precaution to prevent the Wazuh manager from running arbitrary commands on agents in their root security context.

Location

(More Info)
The `location` field specifies where the log data comes from. It includes the following options

- A path to a log file
- A Windows event channel
- The macOS ULS
- The `journald` system

Default Value	**N/A**
Allowed Values File Path, Event Channel. macos, journald .	

- To collect logs from the `journald` system, you must set both `location` and `log_format` to `journald`.

Query

(More Info)

This label can be used to filter *Windows* `eventchannel` events or *macOS* ULS logs (`macos`) that Wazuh will process.

To filter *Windows* `eventchannel` events, *XPATH* format is used to make the queries following the event schema.

Command

(More Info)

Given a command output, it will be read as one or more log messages depending on *command* or *full_command* is used.

Default Value	**N/A**
Allowed Values	Any command line, optionally including arguments

Example Config Snippets

(More Info)

Linux

```
<!-- For monitoring log files -->
<localfile>
  <log_format>syslog</log_format>
  <location>/var/log/syslog</location>
</localfile>

<!-- For monitoring command output -->
<localfile>
  <log_format>command</log_format>
  <command>df -P</command>
  <frequency>360</frequency>
</localfile>

<!-- To use a custom target or format -->
<localfile>
```

```
  <log_format>syslog</log_format>
  <location>/var/log/auth.log</location>
  <target>agent,custom_socket</target>
  <out_format target="custom_socket">$(timestamp %Y-%m-%d %H:%M:%S): $(log)
</out_format>
</localfile>
```

Windows

```
<!-- For monitoring Windows eventchannel -->
<localfile>
  <location>Security</location>
  <log_format>eventchannel</log_format>
  <only-future-events>yes</only-future-events>
  <query>Event/System[EventID != 5145 and EventID != 5156]</query>
  <reconnect_time>10s</reconnect_time>
</localfile>
```