# Linux Security Monitoring with Splunk



## The Current Landscape

### Sysmon for Linux

Add-on for Linux Sysmon | Splunkbase

The Splunk Add-on for Linux Sysmon extract fields from syslog data. Add-On map events for CIM data models: Endpoint, Network Resolution (DNS), Network Traffic, Change. The Splunk Add-on for Linux Sysmon provides the parsing and CIM-compatible knowledge to use with other Splunk apps, such as Splunk Enterprise Security and the Splunk App for PCI Compliance.

https://splunkbase.splunk.com/app/6176/

### Application Focused

Monitoring Linux - Metrics and Logs Forwarding | Splunkbase

Outcold Solutions provide solutions for monitoring Kubernetes, OpenShift, Docker and Linux clusters in Splunk Enterprise and Splunk Cloud. We offer Splunk applications, which give you insights across all environments. We are helping businesses to reduce complexity related to logging and monitoring by

https://splunkbase.splunk.com/app/4768/

## Observability Oriented

Splunk Add-on for Linux | Splunkbase

The Splunk Add-on for Linux allows a Splunk software administrator to collect Linux performance metrics using HTTP Event Collector (HEC) or TCP. The Splunk Add-on for Linux collects data includes: * CPU metrics.* Memory metrics.* Swap metrics.* Mountpoint usage/FS usage.* Network interface traffic.* Disk utilization.* System load.* Process information.* Network protocols information.* IRQ metrics.* TCP connections information.* Thermal information.* System uptime statistics.

https://splunkbase.splunk.com/app/3412/

## Observability & Security

Splunk Add-on for Unix and Linux | Splunkbase

Important: Read upgrade Instructions and test add-on update before deploying to production ***There are changes to default indexes and .conf changes in version 6.0 of Splunk Add-on for Unix and Linux that can break an existing installation if upgrade instructions are not followed in detail.

https://splunkbase.splunk.com/app/833/

## Security Focused

Linux Secure Technology Add-On | Splunkbase

This app provides field extractions and normalisation to the Common Information Model for /var/log/secure and /var/log/auth.log (linux_secure sourcetype). It is intended to replace the security-relevant aspects of the Splunk Add-on for Unix and Linux (Splunk_TA_nix) and as such it's strongly recommended that the Splunk_TA_nix app be removed from your search head before installing this app as they may conflict.

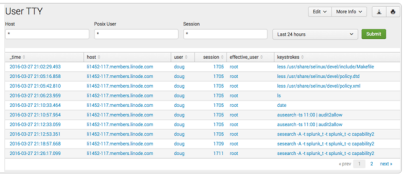
https://splunkbase.splunk.com/app/3476/

## Linux Auditd

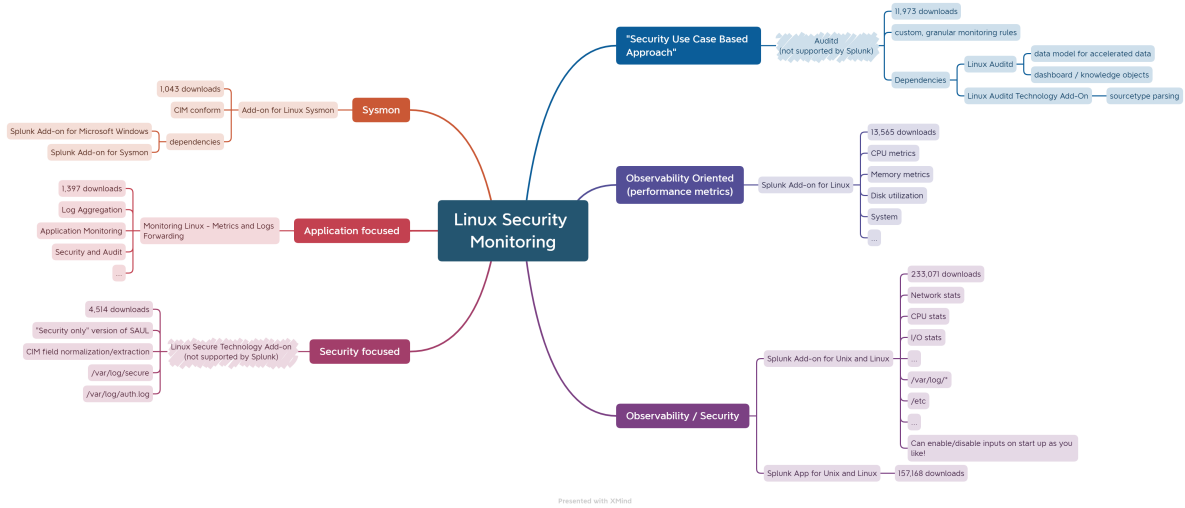
Linux Auditd | Splunkbase

Splunk .conf 2018 Edition!

https://splunkbase.splunk.com/app/2642/




## Feature Comparison




*Which solution should we use?*

# Demo Use Cases

## T1016 - System Network Configuration Discovery

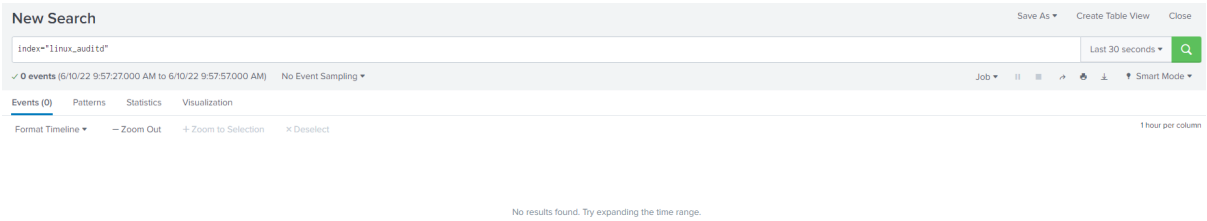
Adversaries my use certain utilities to discover network configurations and settings. This may include IP and MAC addresses, interfaces configurations, open ports, etc.

For this use case we will use the following script to conduct network configuration discovery and trigger alerts in Splunk:

```
# .sh script
if [ -x "$(command -v arp)" ]; then arp -a; else echo "arp is missing from the machine. skipping..."; fi;
if [ -x "$(command -v ifconfig)" ]; then ifconfig; else echo "ifconfig is missing from the machine. skipping..."; fi;
if [ -x "$(command -v ip)" ]; then ip addr; else echo "ip is missing from the machine. skipping..."; fi;
if [ -x "$(command -v netstat)" ]; then netstat -ant | awk '{print $NF}' | grep -v '[a-z]' | sort | uniq -c; else echo "netstat is mis
```

### Auditd Monitoring

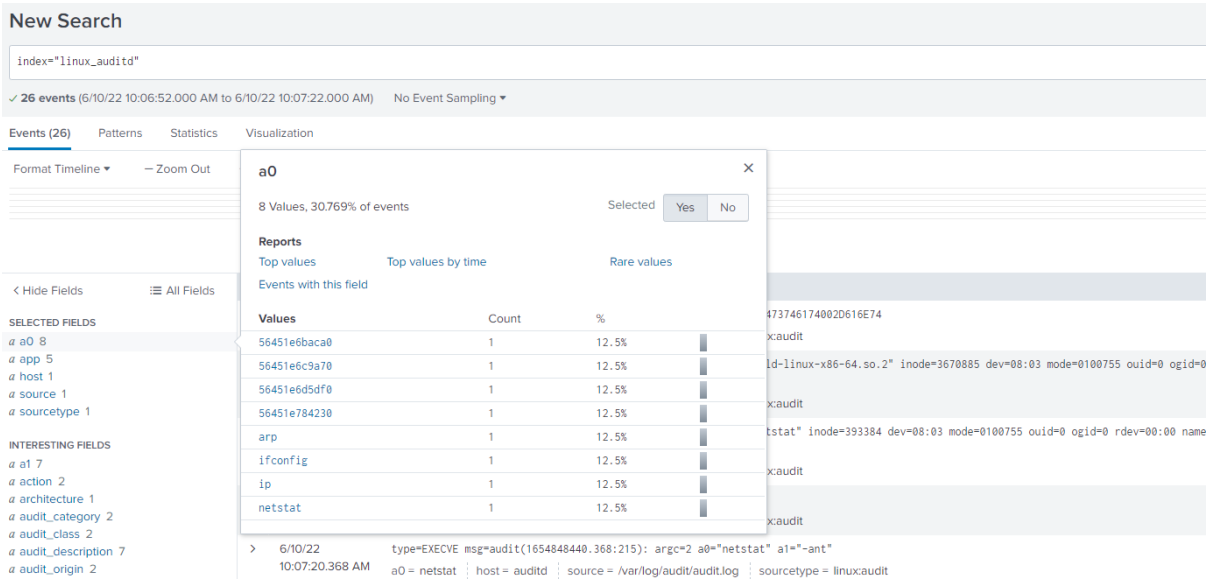
After executing the script, we see that there are no log events generated:



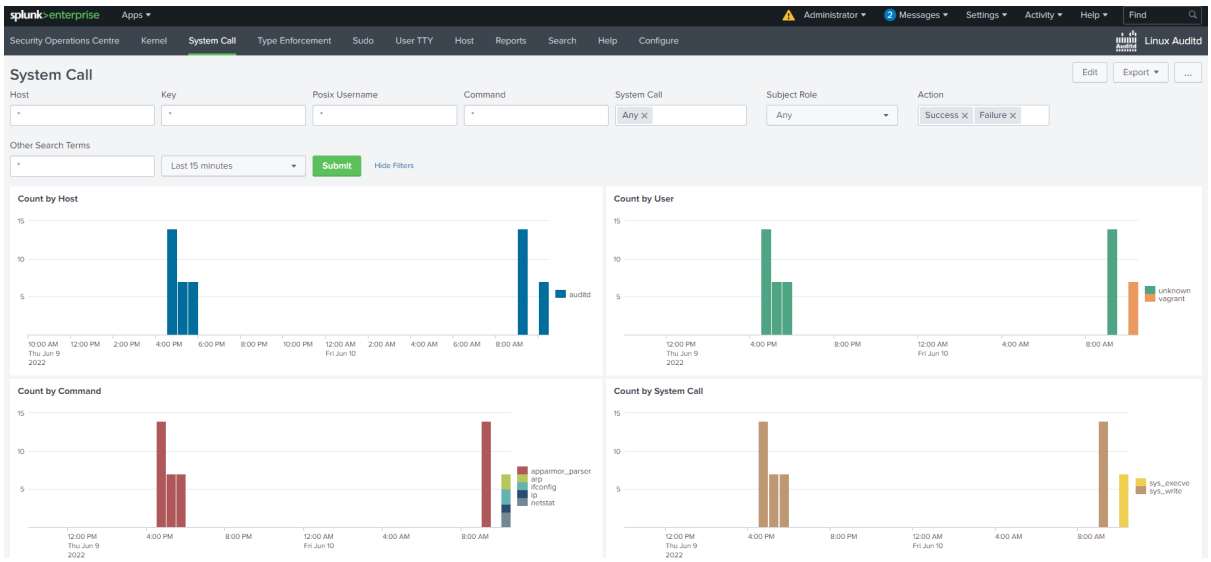

This is because we have not specified in the audit rules any conditions for auditd to actively monitor. Opening the file `/etc/audit/rules.d/audit.rules`, we can add the following:

```
-w /usr/sbin/arp -p x -k T1016_network_discovery
# -w /sbin/ip has symlink: /sbin/ip -> /bin/ip*
-w /bin/ip -p x -k T1016_network_discovery
-w /bin/netstat -p x -k T1016_network_discovery
-w /sbin/ifconfig -p x -k T1016_network_discovery
```

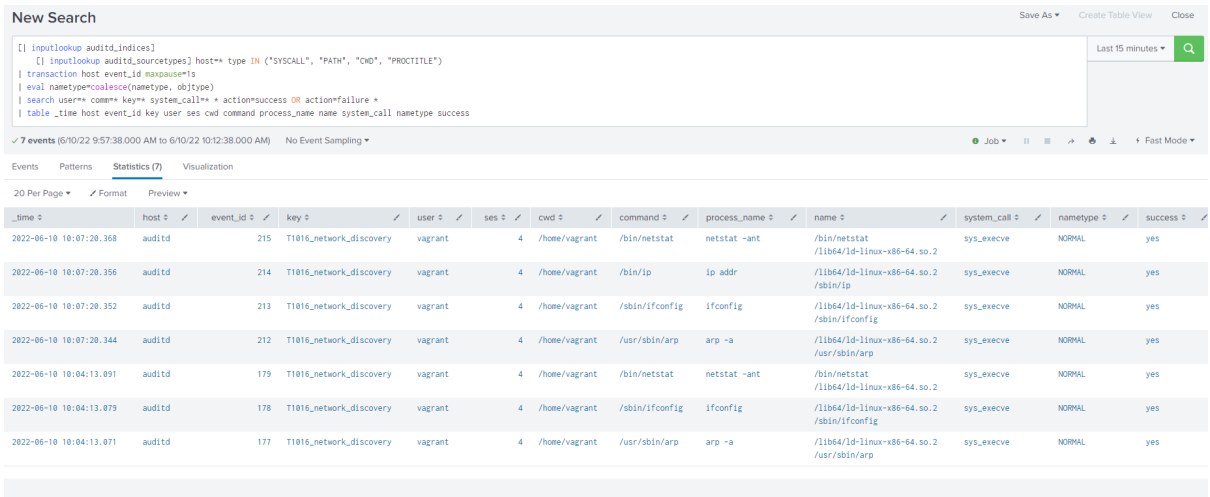
Note here that `/sbin/ip` has a symlink to another location, although the `which` command specified this binary. The final binary must be given in the `audit.rules`. Executing the script again, we now see many Splunk events and the name of each command given the `a0` field:



Heading over to the Auditd App we can examine some interesting dashboards for these events. For example, in the System Call dashboard we see some visualizations to these recent events:




When using a different SIEM app, such as Splunk Enterprise Security, the Correlation Events table is the most important visualization. It not only gives a great overview of what has happened on system which Auditd has successfully logged, but it also provides a basis for the correlation search logic which will be built in the Splunk ES context:




Further audit rules for this use case that could be helpful:

```
-w /usr/bin/ping -p x -k recon
-w /usr/bin/curl -p x -k recon
-w /usr/sbin/dhclient -p x -k recon
-w /usr/sbin/ethtool -p x -k recon
-w /usr/sbin/ifstat -p x -k recon
-w /usr/sbin/ifconfig -p x -k recon
-w /usr/sbin/ifup -p x -k recon
-w /usr/sbin/ifdown -p x -k recon
-w /usr/bin/ip -p x -k recon
# /sbin/ip -> /bin/ip
-w /usr/bin/routel -p x -k recon
-w /usr/bin/routef -p x -k recon
-w /usr/bin/ss -p x -k recon
# ss in /usr/bin/
-w /usr/bin/ssh -p x -k recon
-w /usr/bin/tracepath -p x -k recon
-w /usr/bin/netstat -p x -k recon
-w /usr/sbin/traceroute -p x -k recon
# traceroute -> /etc/alternatives/traceroute.sbin
-w /etc/alternatives/traceroute.sbin -p x -k recon
# /etc/alternatvies/traceroute.sbin -> /usr/bin/traceroute.db
```

```
-w /usr/bin/traceroute.db -p x -k recon
-w /usr/bin/mtr -p x -k recon
-w /usr/sbin/arp -p x -k recon
```

atomic-red-team/T1016.md at master · redcanaryco/atomic-red-team

Adversaries may look for details about the network configuration and settings, such as IP and/or MAC addresses, of systems they access or through information discovery of remote systems. Several operating system administration utilities exist that can be used to gather this information.

https://github.com/redcanaryco/atomic-red-team/blob/master/atomics/T1016/T1016.md#atomic-test-3---system-network-configuration-discovery

redcanaryco/**atomic-red-team**

Small and highly portable detection tests based on MITRE's ATT&CK.

246 Contributors · 10 Issues · 6k Stars · 2k Forks

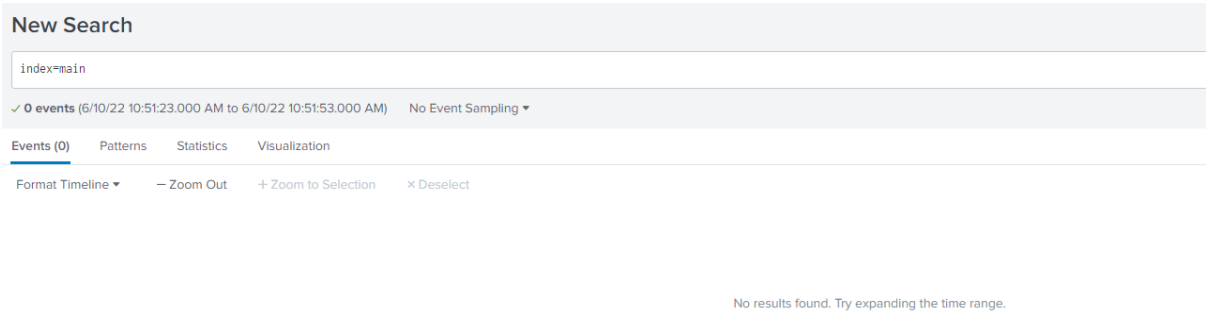
System Network Configuration Discovery

Adversaries may look for details about the network configuration and settings, such as IP and/or MAC addresses, of systems they access or through information discovery of remote systems. Several operating system administration utilities exist that can be used to gather this information. Examples include Arp,

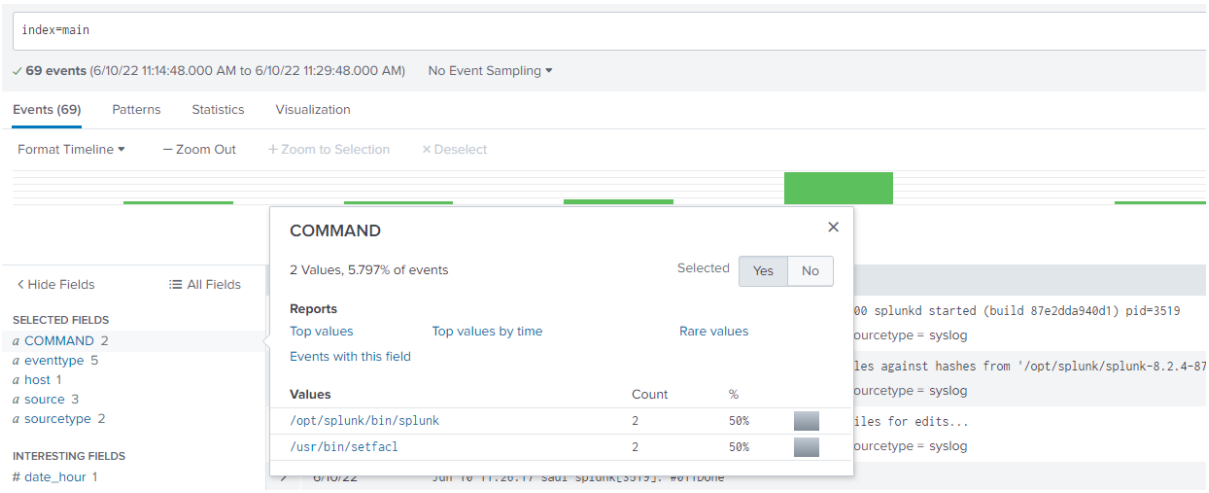
https://attack.mitre.org/techniques/T1016/

### SAUL Monitoring

When the above script is executed now on a system running SAUL and not Auditd, there are results for the following search:




Instead in the last fifteen minutes, only using setfacl and restarting Splunk have been logged, but nothing to do with the discovery script that was executed:




Thus, if we want to detect system network configuration discovery, it seems this TA alone will not help us.

## T1003.008 - /etc/passwd and /etc/shadow

Dumping the contents of these files can allow for password cracking. In the case of hashed passwords, one could could a rainbow table to discover the password by finding the corresponding hash from a known base of clear text passwords. John the Ripper also contains the capability for password cracking, using `unshadow` with `/etc/passwd` and `/etc/shadow`.

OS Credential Dumping: /etc/passwd and /etc/shadow

Adversaries may attempt to dump the contents of /etc/passwd and /etc/shadow to enable offline password cracking. Most modern Linux operating systems use a combination of /etc/passwd and /etc/shadow to store user account information including password hashes in /etc/shadow. By default, /etc/shadow is only

https://attack.mitre.org/techniques/T1003/008/

atomic-red-team/T1003.008.md at master · redcanaryco/atomic-red-team

Adversaries may attempt to dump the contents of /etc/passwd and /etc/shadow to enable offline password cracking. Most modern Linux operating systems use a combination of /etc/passwd and /etc/shadow to store user account information including password hashes in /etc/shadow.

https://github.com/redcanaryco/atomic-red-team/blob/master/atomics/T1003.008/T1003.008.md

redcanaryco/**atomic-red-team**

Small and highly portable detection tests based on MITRE's ATT&CK.

246 Contributors · 10 Issues · 6k Stars · 2k Forks

## Auditd Monitoring

In order to detect read or write to the these files, the following audit rules can be added:

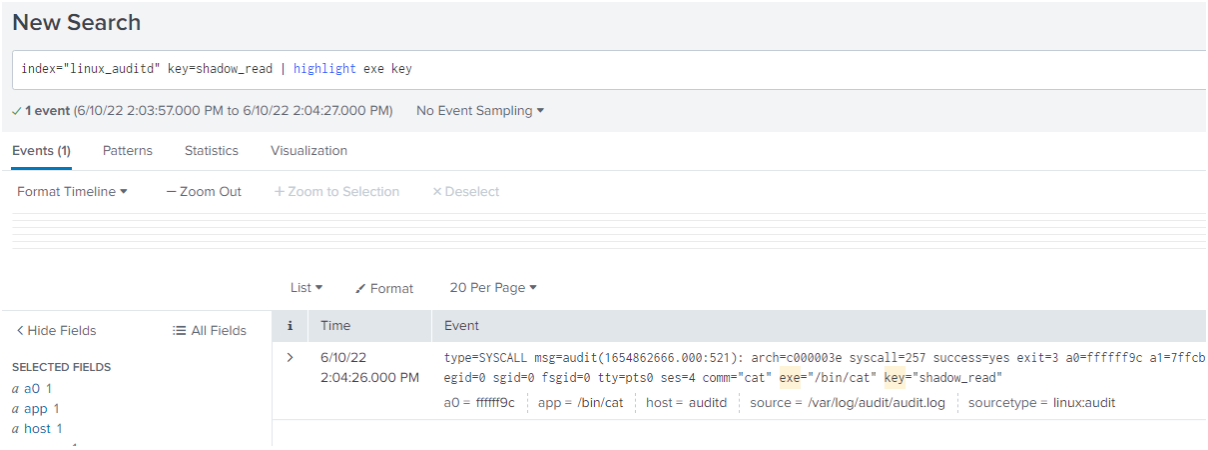
```
-w /etc/passwd -p r -k passwd_read
-w /etc/passwd -p wa -k passwd_write
-w /etc/shadow -p r -k shadow_read
-w /etc/shadow -p wa -k shadow_write
```

We can monitor the events generated and ingested in Splunk for the following tests:

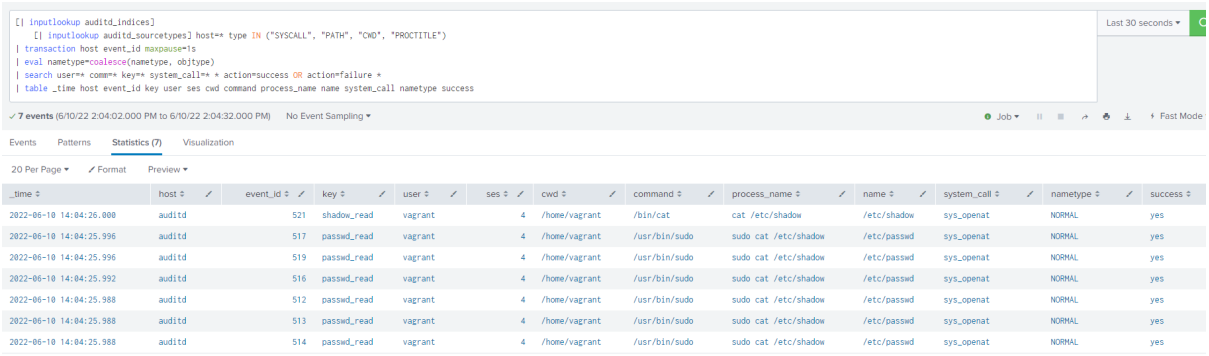
```
# Test 1
sudo cat /etc/shadow > sys_pwds

# Test 2
cat /etc/passwd > sys_accs

# Test 3: bash script
function testcat(){ echo "$(< $1)"; }
testcat /etc/passwd > credential_dump
testcat /etc/shadow >> credential_dump
# execute with
sudo bash credential_access.sh
```

**Test 1**

Searching for the key, we see the logged credential dump:

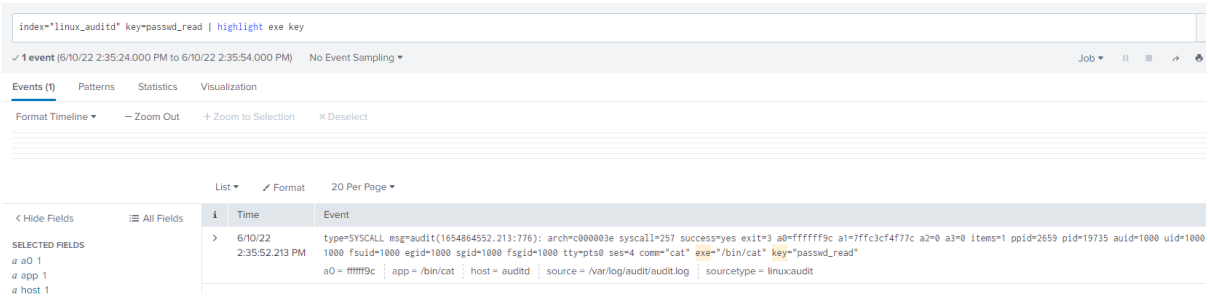



In the App we get a view of the transaction:

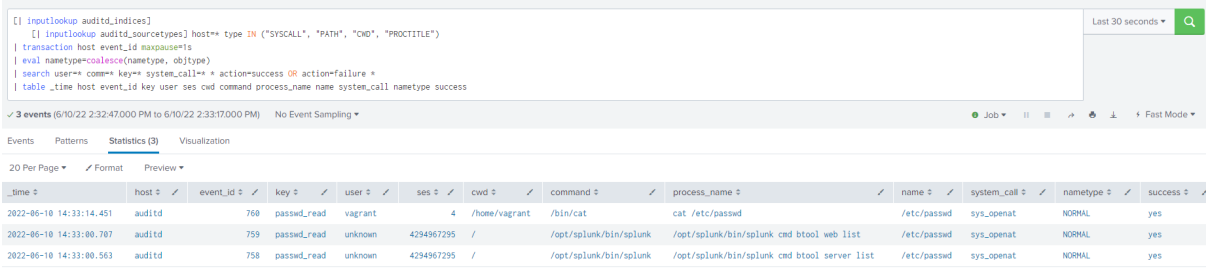



**Test 2**:

Monitored events:

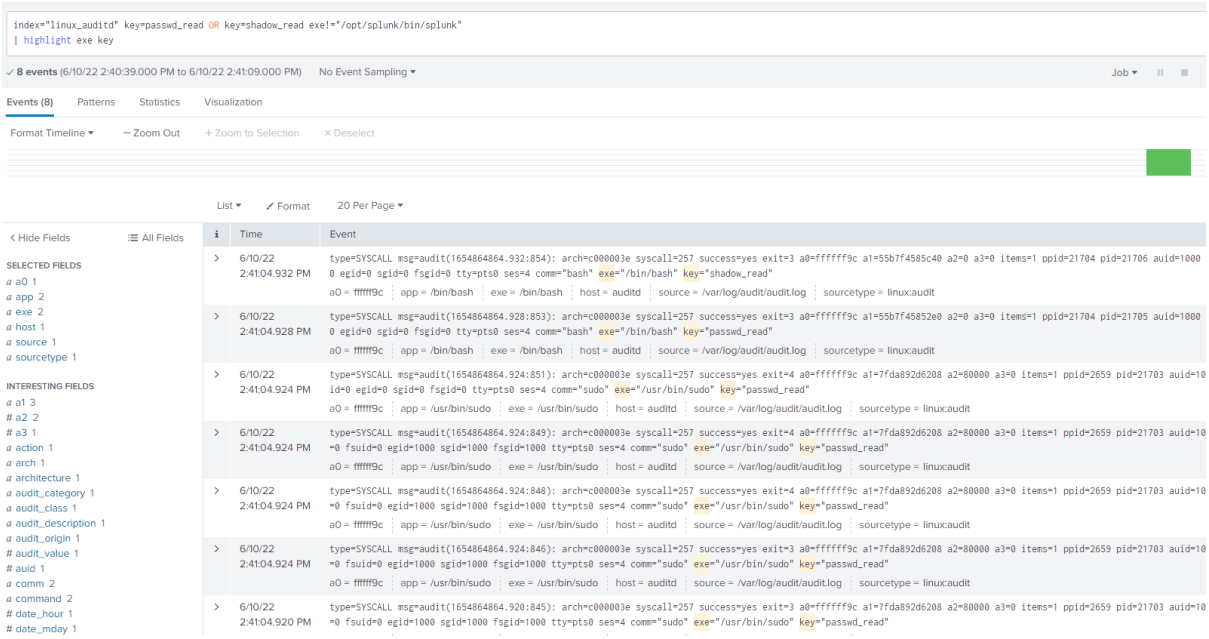



False positives with running Splunk process:

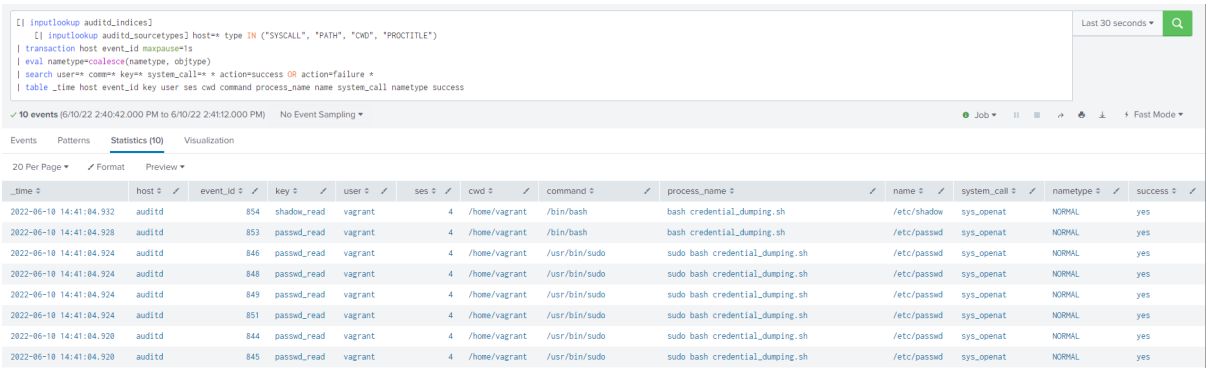



**Test 3**:

After running the bash script, we get the following events:




Transaction:

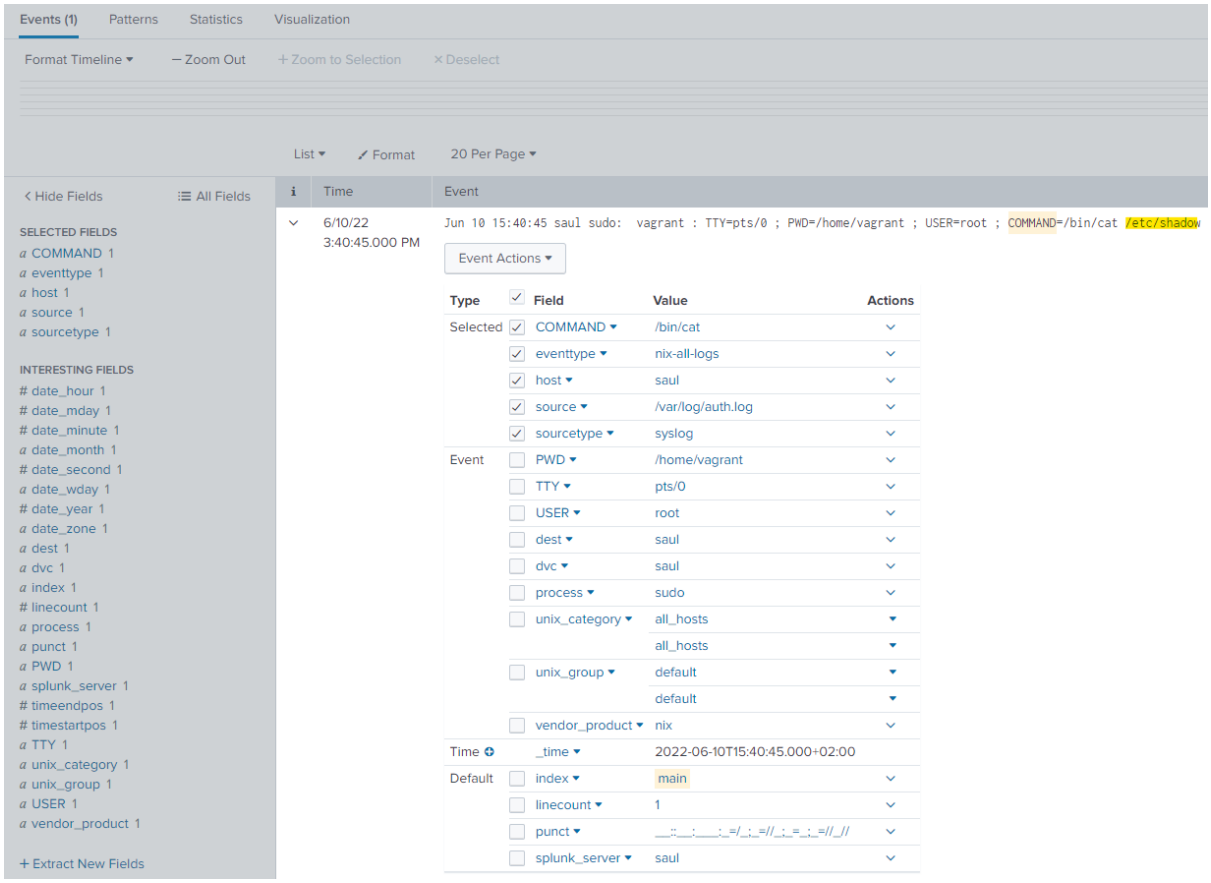



Similar false positives as before and with cron jobs.

## SAUL Monitoring

**Test 1**:

Here we do see an event from credential dumping:



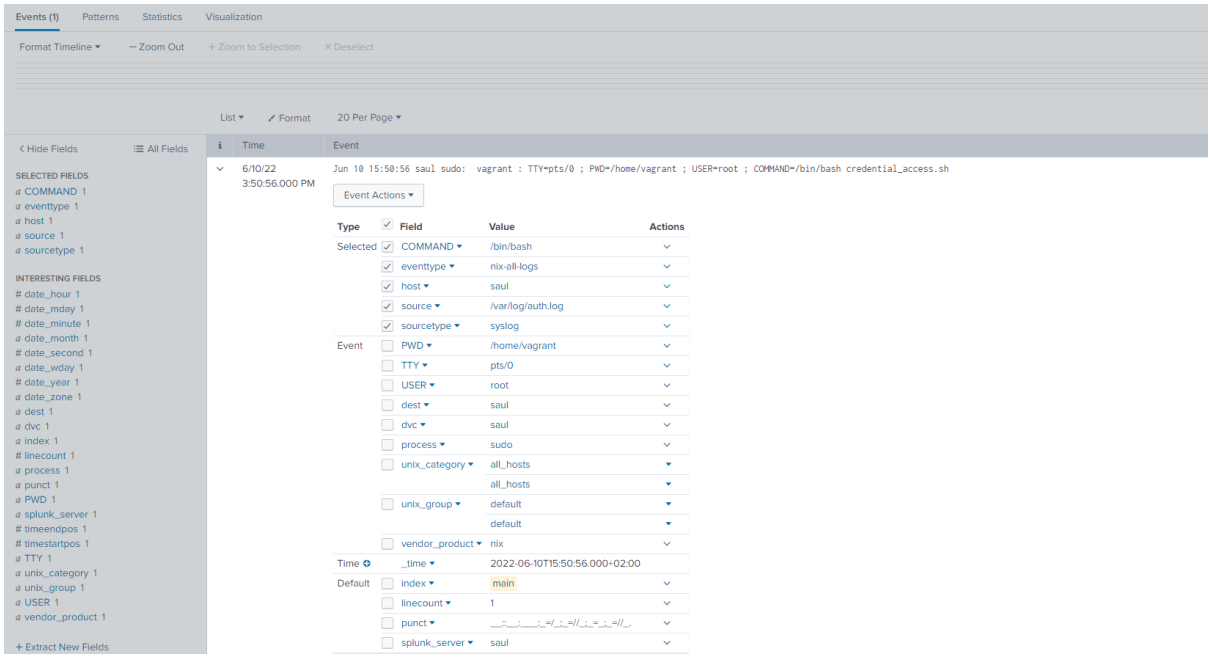

From the field COMMAND it is clear that someone had read the shadow file. The reason this event is logged is because `sudo` was necessary in order to read the hashed passwords. This writes an entry in the `auth.log`.

**Test 2**:

Nothing is loggged and read from the monitored inputs.

**Test 3**:

Just as in Test 1, this event is logged because super user privileges were required to read `shadow`:



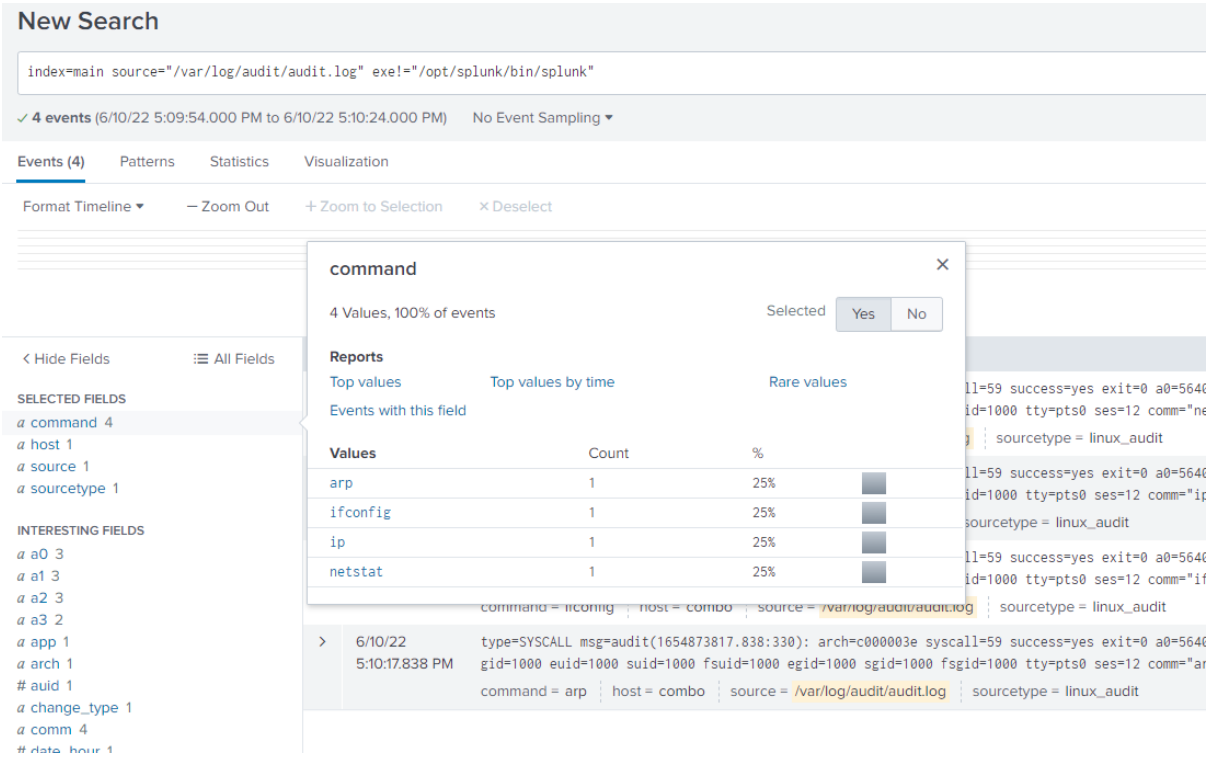
## SAUL + Auditd Consideration

Here we will use an additional instance to setup Splunk with Splunk Add-On for Unix and Linux (SAUL) just as before, but we will install and configure auditd as well. Since SAUL has a monitoring input on `/var/log`, the `audit.log` will be monitored. In fact, we need not install the Auditd TA. SAUL has the following stanza in the `props.conf` already for handling the log parsing:

```
[linux_audit]
REPORT-command = command_for_linux_audit
EVAL-status = if('res'=="failed","failure",'res')
FIELDALIAS-object = id as object
FIELDALIAS-dvc = hostname as dvc
FIELDALIAS-dest = hostname as dest
FIELDALIAS-object_id = id as object_id
EVAL-op = if(op=="PAM:authentication", res, op)
EVAL-vendor_product = if(isnull(vendor_product), "nix", vendor_product)
LOOKUP-action = nix_linux_audit_action_lookup op OUTPUT action,object_category
EVAL-object_attrs= case(type=="ADD_USER" OR type=="USER_MGMT" OR type=="DEL_USER",grp)
EVAL-app = "nix"
EVAL-change_type = "AAA"
EVAL-object = if((type="GRP_MGMT" OR type="DEL_GROUP" or type=="ADD_GROUP") AND isnotnull('grp'),'grp','object')
EVAL-user = case((type=="ADD_USER" OR type=="USER_MGMT" OR type=="DEL_USER" OR type=="USER_CMD") AND isnull('user'),'id',(type=="GRP_M
EVAL-user_name = case((type=="ADD_USER" OR type=="USER_MGMT" OR type=="DEL_USER" OR type=="USER_CMD") AND isnull('user'),'id',(type=="
EVAL-user_id = if(type=="GRP_MGMT" OR type=="DEL_GROUP" or type=="ADD_GROUP" ,'uid','id')
EVAL-src_user = case((type=="ADD_USER" OR type=="USER_MGMT" OR type=="DEL_USER" OR type=="USER_AUTH" ) AND uid=="0" ,"root",type=="ADD
EVAL-src_user_name = case((type=="ADD_USER" OR type=="USER_MGMT" OR type=="DEL_USER" OR type=="USER_AUTH" ) AND uid=="0" ,"root",type=
EVAL-src_user_id = if(type=="ADD_USER" OR type=="USER_MGMT" OR type=="DEL_USER" OR type=="USER_AUTH" ,'uid','src_user_id')
EVAL-reason = if(type="USER_AUTH" AND (res=="failed" OR res=="failure"),"other",'reason')
```

### Use Case 1: System Network Configuration Discovery

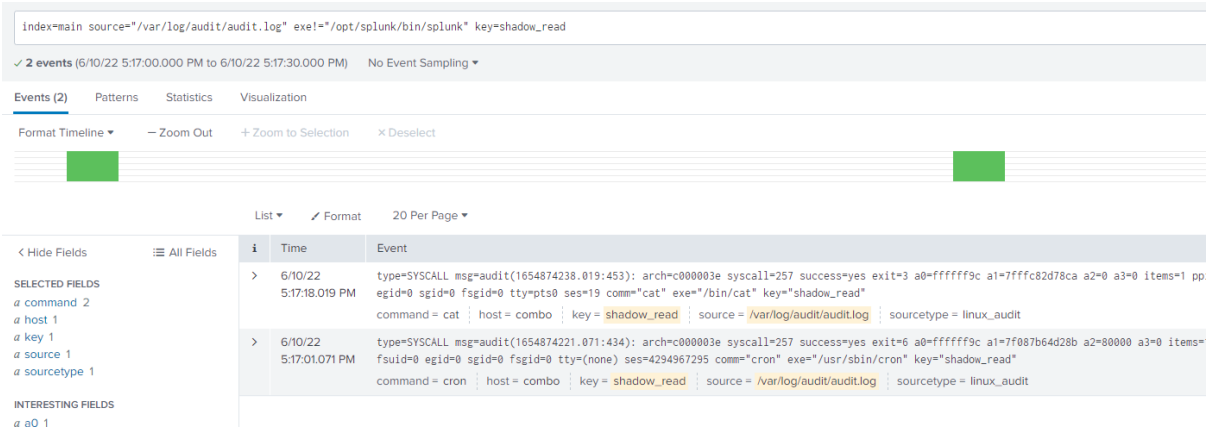
Execution of the network_discovery script has been successfully detected:



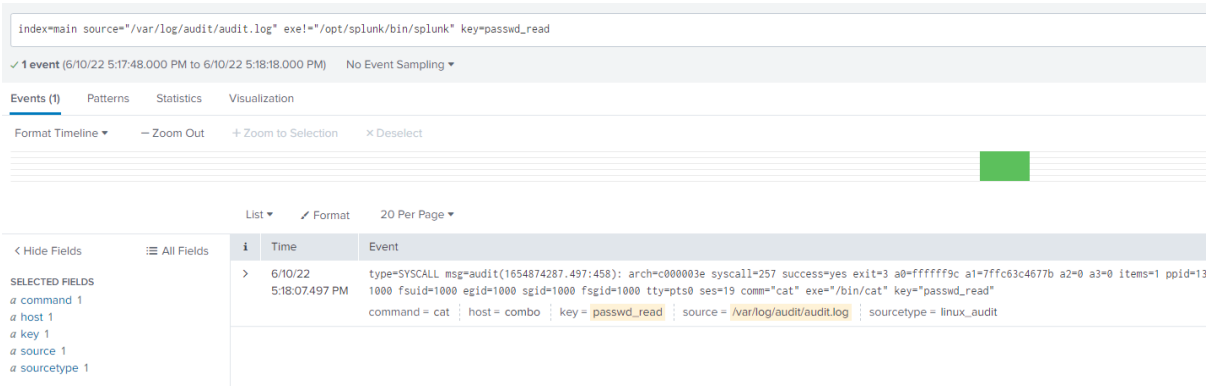

Note: similar false positives as before.

### Use Case 2: Credential Dumping

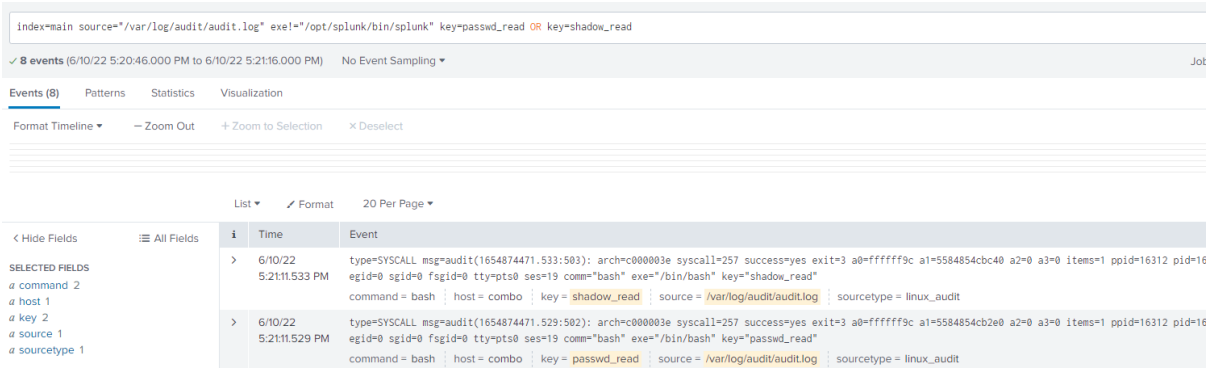
**Test 1**:




**Test 2**:




**Test 3**:




# Pros and Cons

## Auditd

**Pros**

- Easy to install
- Helpful App for SOC overview and building correlation searches
- Granular, “use case based” configuration
- Logs only what you need for specific use cases

**Cons**

- For more complex use cases, a deeper understanding of Linux needed and is more time-consuming to construct and test the rules
- No “out-of-the-box” logging: risk of missing something you weren’t looking for

**Pro/Cons?**

- No way (as far as I know) to get observability data out of it
- Not supported by Splunk

## SAUL

**Pros**

- Can ingest audit.log and parse the events without needing the Auditd TA
- Observability with elegant dashboards in App
- Can work with auditd with built in linux_audit sourcetype in props.conf

**Cons**

- No granular logging, i.e., not a “use case based” approach
- Default monitoring inputs potentially misses A LOT of things we need to detect