


Exploration, Monitoring and Security with osquery

KOLIDE

Zach Wasserman

Cofounder & Principal Engineer

Preparing for Interactivity

- Please download and install osquery so that you can follow along with the examples in this presentation.

- osquery.io/downloads (Mac pkg is ~12MB)

- Note Homebrew support is limited. Please download and install the package.

- You are set up correctly if you can open a terminal and start the program **osqueryi**.

Who Am I?

- Core osquery contributor since its inception in 2014.
- Cofounder & Principal Engineer at Kolide.
- Currently building open-source (Kolide Fleet, Kolide Launcher) and commercial (Kolide Cloud) software in the osquery ecosystem

The Problem

The Problem

- Sysadmins and security folks have a huge number of sources for the data relevant to their operations and decision-making.

- How can we reliably access this data to get an understanding of the system state in the present moment, and as it changes over time?

Introducing Osquery

KOLIDE

Introducing Osquery

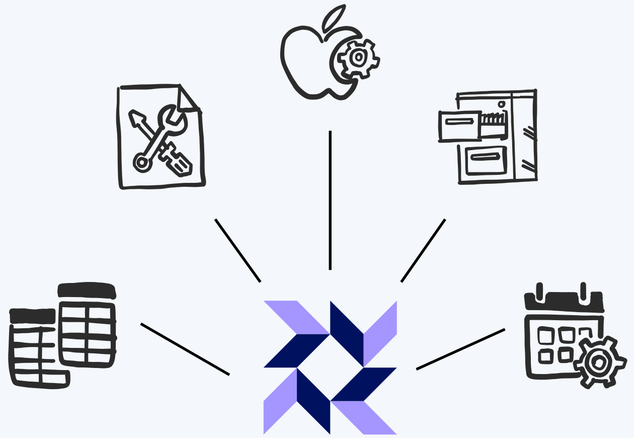
- Open-sourced by Facebook in 2014. Still supported by a core team at FB.
- 4,486+ commits, 231+ contributors, 12,707+ stars on Github
- Apache 2.0 License
- osquery.io

Osquery Goals

- First class support for macOS/Linux
- Enable non-developers to access and aggregate data across disparate sources
- Performance/reliability to deploy across corporate and production infrastructure

Unify disparate sources of information

- Flat files (/etc/hosts, /etc/crontab, ~/.ssh/known_hosts, etc.)
- SQLite files (/var/db/SystemPolicy [GateKeeper configuration], etc.)
- System APIs (Apple System Log, Keychain, SMC, CoreFoundation, etc.)
- Application APIs (Docker, Carbon Black, etc.)
- Event-based APIs (FSEvents, OpenBSM, etc.)
- Filesystem (Shared folders, file hashes, permissions, etc.)
- Plists (/Library/Managed\ Installs/* [Munki data], etc.)
- … And more …



The Power of SQL

KOLIDE

osquery> SELECT * FROM...

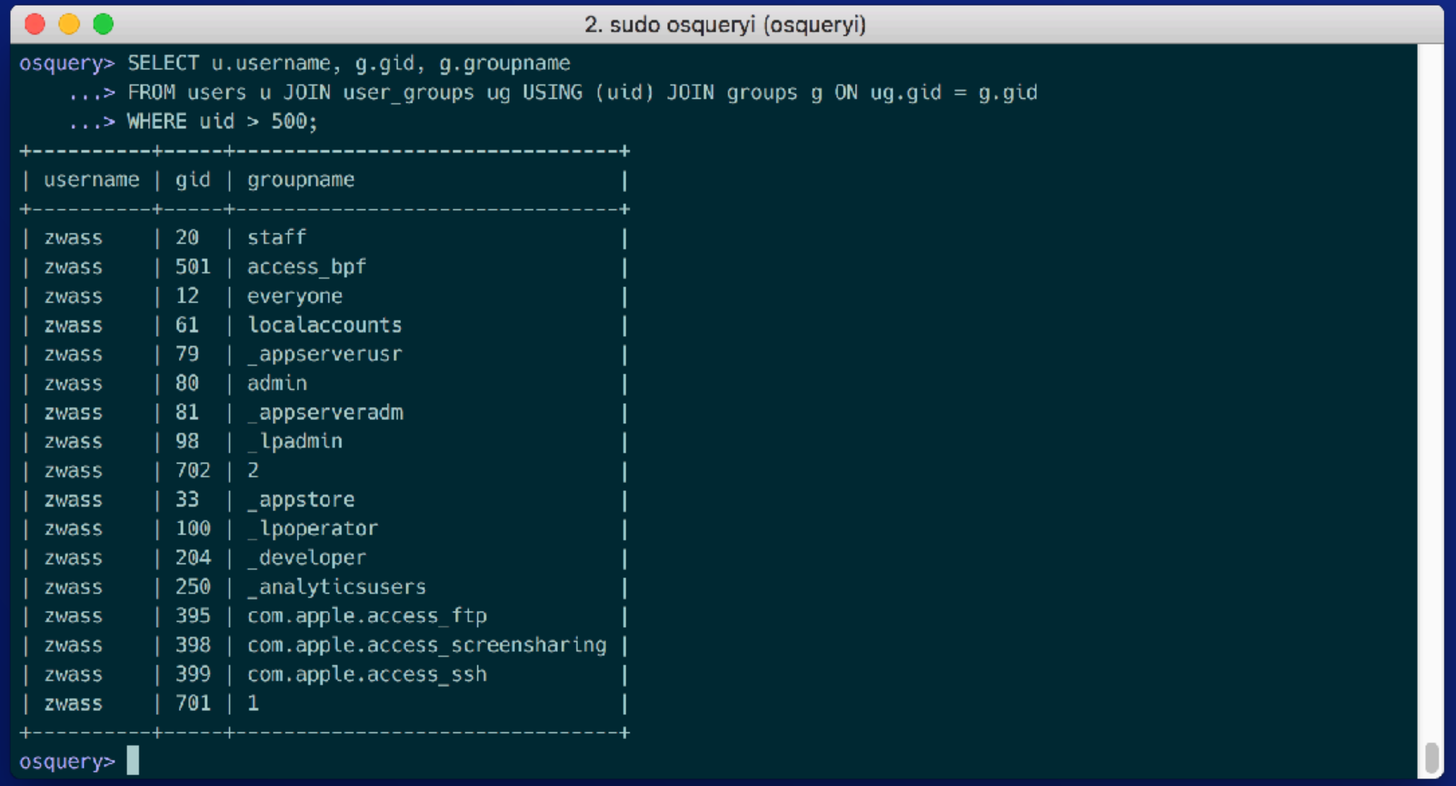
account_policy_data
acpi_tables
ad_config
alf
alf_exceptions
alf_explicit_auths
alf_services
app_schemes
apps
apt_sources
arp_cache
asl
augeas
authorization_mechanisms
authorizations
authorized_keys
block_devices
browser_plugins
carbon_black_info
carves
certificates
chrome_extensions
cpu_time
cpuid
crashes
crontab
cups_destinations
cups_jobs
curl
curl_certificate
device_file
device_firmware
device_hash
device_partitions
disk_encryption
disk_events
dns_resolvers
docker_container_labels
docker_container_mounts
docker_container_networks
docker_container_ports
docker_container_processes
docker_container_stats
docker_containers
docker_image_labels
docker_images
docker_info
docker_network_labels
docker_networks
docker_version
docker_volume_labels
docker_volumes
etc_hosts
etc_protocols
etc_services
event_taps
extended_attributes
fan_speed_sensors
file
file_events
firefox_addons
gatekeeper
gatekeeper_approved_apps
groups
hardware_events
hash
homebrew_packages
intel_me_info
interface_addresses
interface_details
iokit_devicetree
iokit_registry
kernel_extensions
kernel_info
kernel_panics
keychain_acls
keychain_items
known_hosts
last
launchd
launchd_overrides
listening_ports
load_average
logged_in_users
magic
managed_policies
mdfind
memory_devices
mounts
nfs_shares
nvram
opera_extensions
os_version
osquery_events
osquery_extensions
osquery_flags
osquery_info
osquery_packs
osquery_registry
osquery_schedule
package_bom
package_install_history
package_receipts
pci_devices
platform_info
plist
power_sensors
preferences
process_envs
process_events
process_memory_map
process_open_files
process_open_sockets
processes
prometheus_metrics
python_packages
quicklook_cache
routes
safari_extensions
sandboxes
shared_folders
sharing_preferences
shell_history
signature
sip_config
smbios_tables
smc_keys
startup_items
sudoers
suid_bin
system_controls
system_info
temperature_sensors
time
time_machine_backups
time_machine_destinations
uptime
usb_devices
user_events
user_groups
user_interaction_events
user_ssh_keys
users
virtual_memory_info
wifi_networks
wifi_status
wifi_survey
xprotect_entries
xprotect_meta
xprotect_reports
yara
yara_events

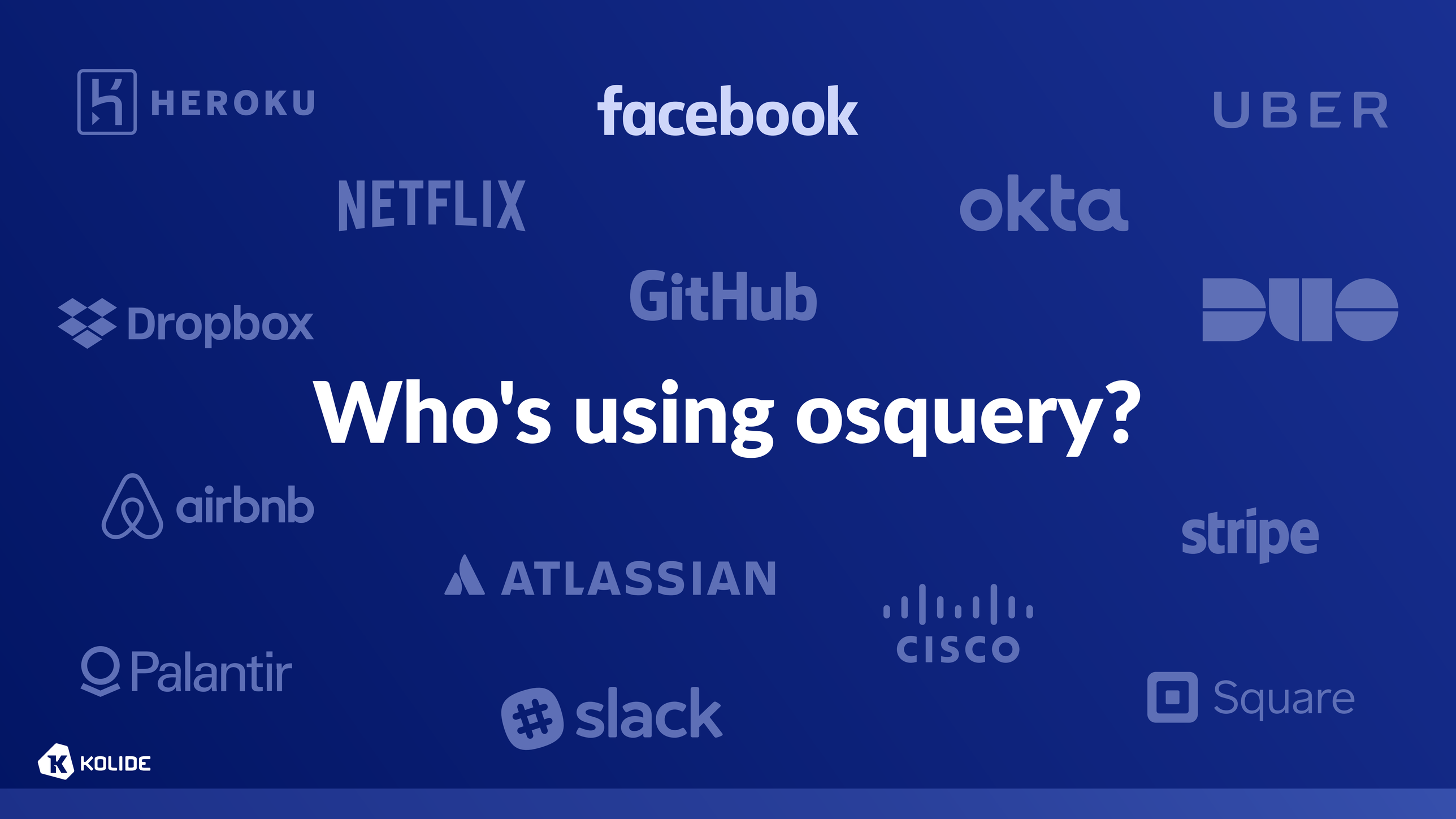
KOLIDE

The Power of SQL

- `select * from hosts; -- /etc/hosts`
- `select * from smc_keys; -- SMC`
- `select * from keychain_items; -- Keychain`
- `select * from file_events; -- FSEvents`
- `select * from hash where path = ‘/bin/bash'; -- File hashes`

```
osquery> SELECT u.username, g.gid, g.groupname FROM users u JOIN user_groups
ug USING (uid) JOIN groups g ON ug.gid = g.gid  WHERE uid > 500;
```




HEROKU
facebook
UBER
NETFLIX
okta
GitHub
Dropbox
DUO
Who's using osquery?
airbnb
stripe
ATLASSIAN
CISCO
Palantir
slack
Square
KOLIDE

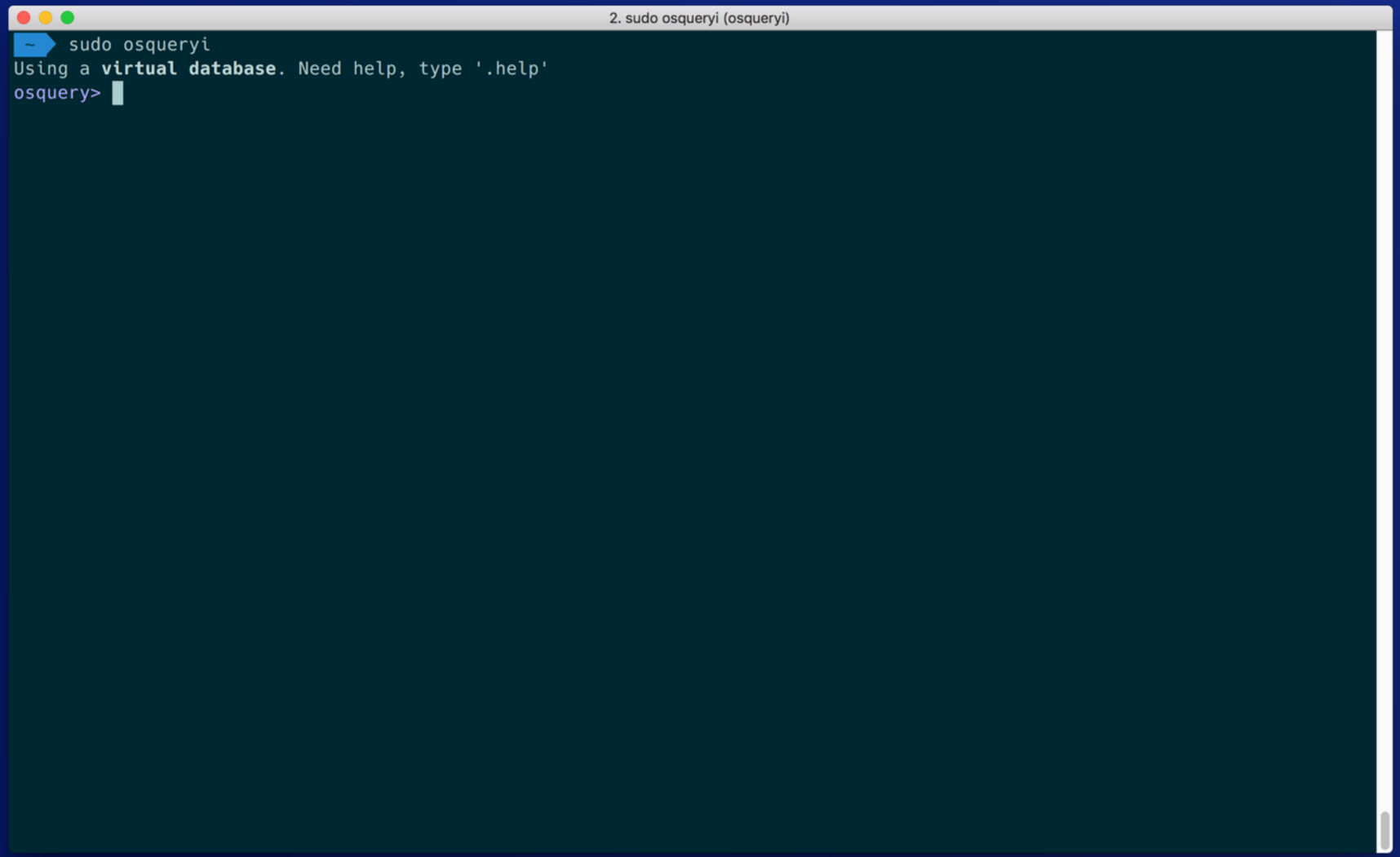
Digging In

KOLIDE

osqueryi

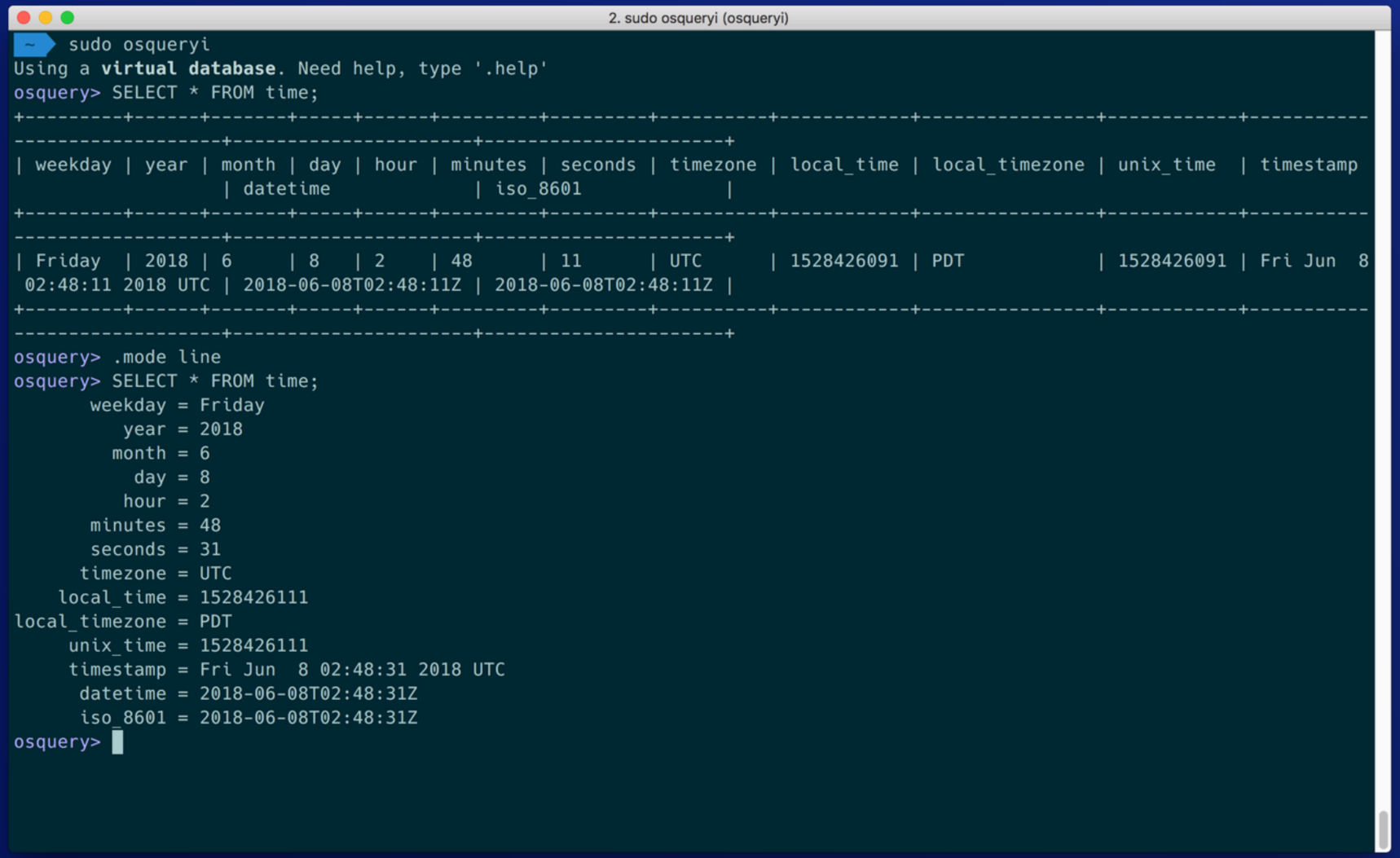
- CLI and interactive shell for executing queries and viewing results
- Use this as a part of scripts, or for manual exploration
- After iterating on and understanding queries in osqueryi, evolve them to create monitoring via osqueryd (more later)

osqueryi



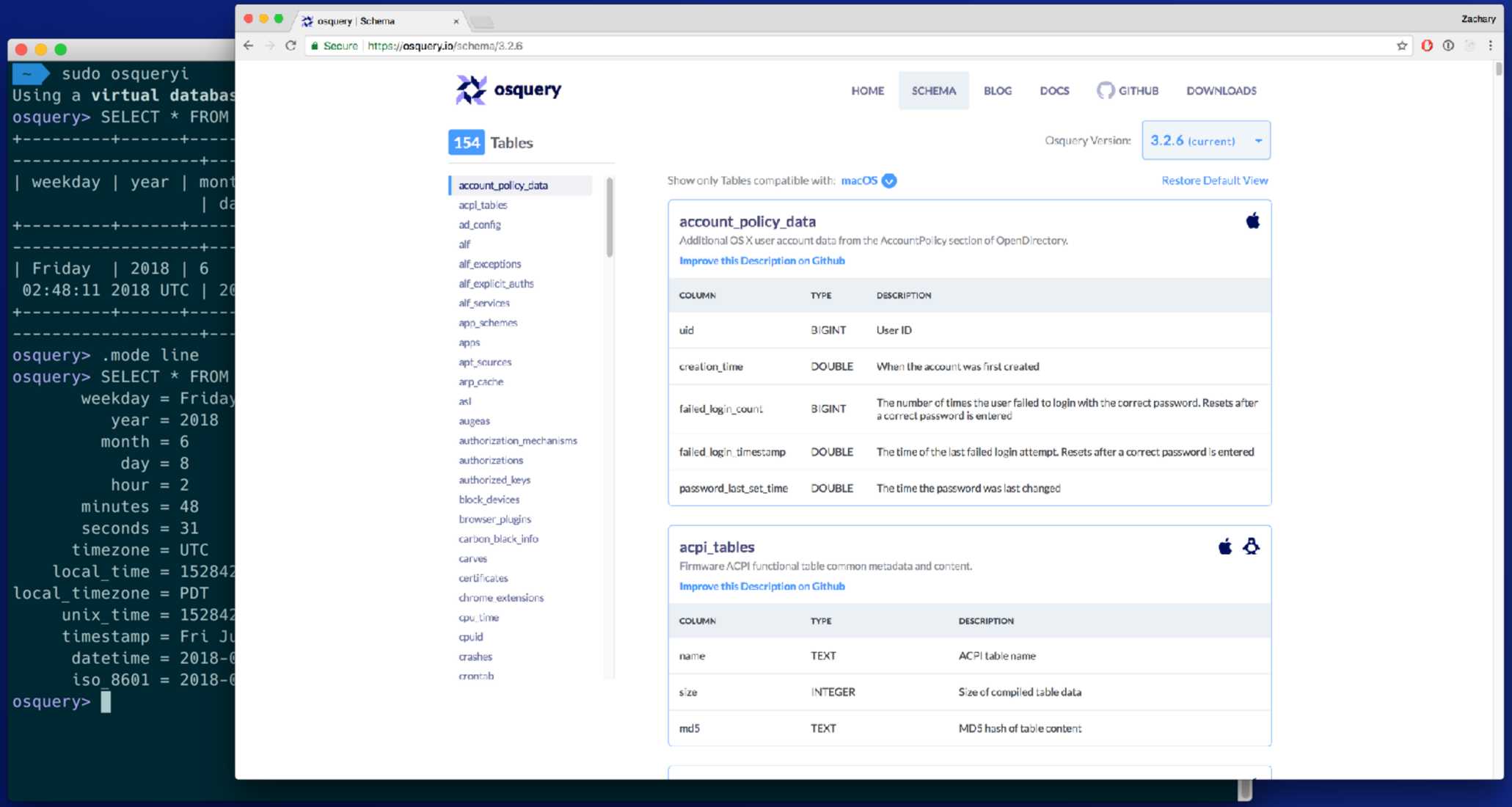
KOLIDE

osqueryi



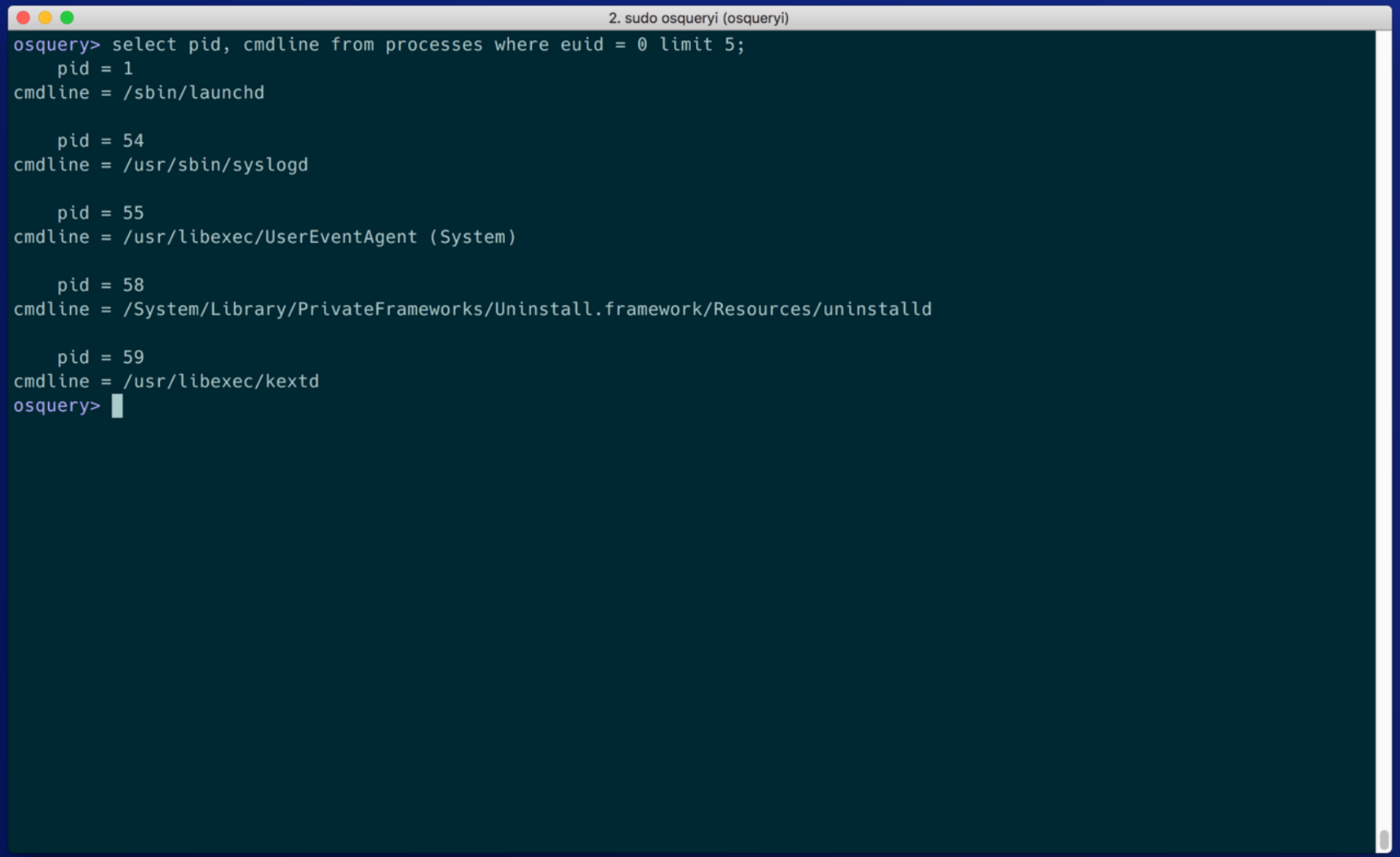
KOLIDE

osqueryi



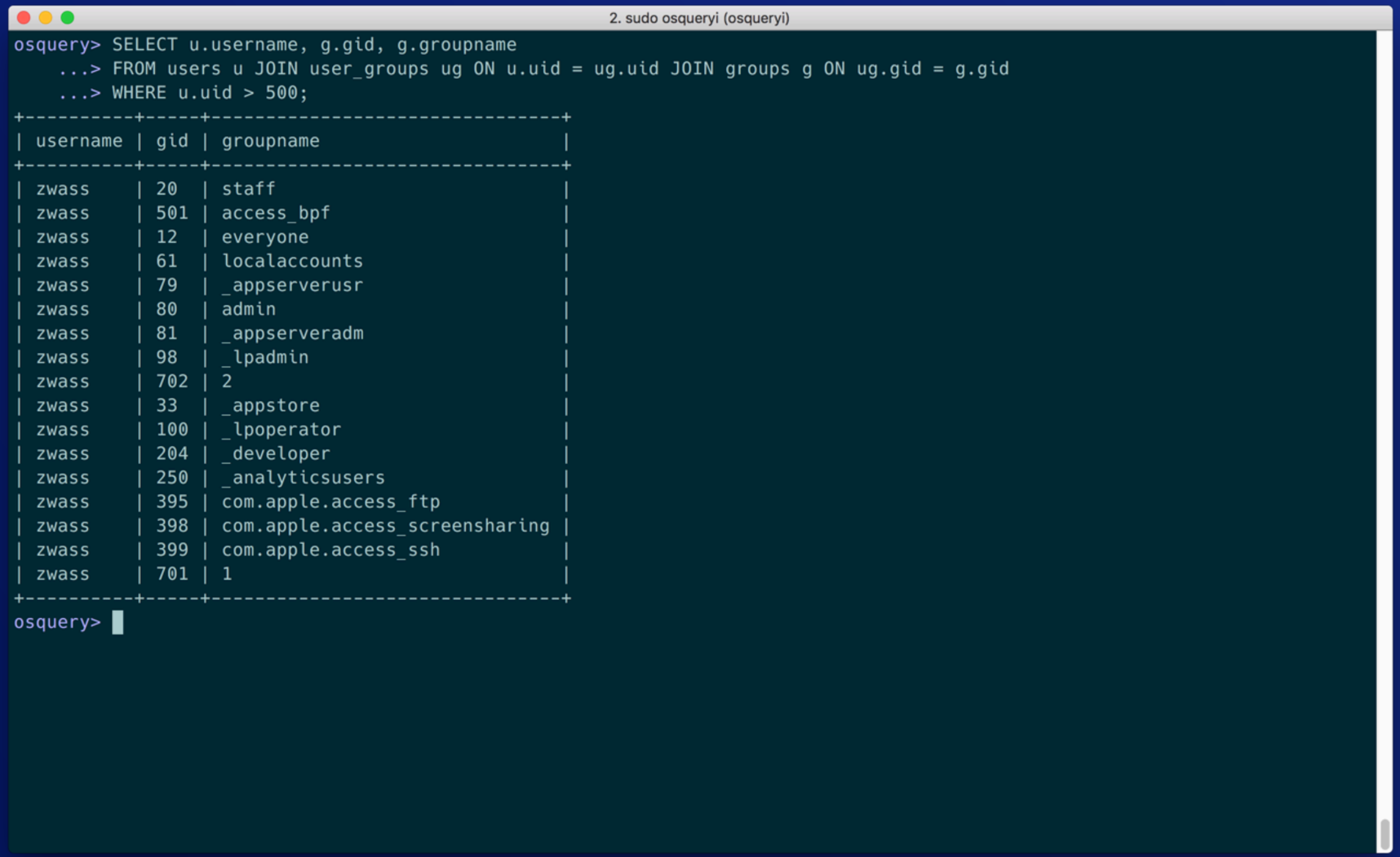
KOLIDE

osqueryi



KOLIDE

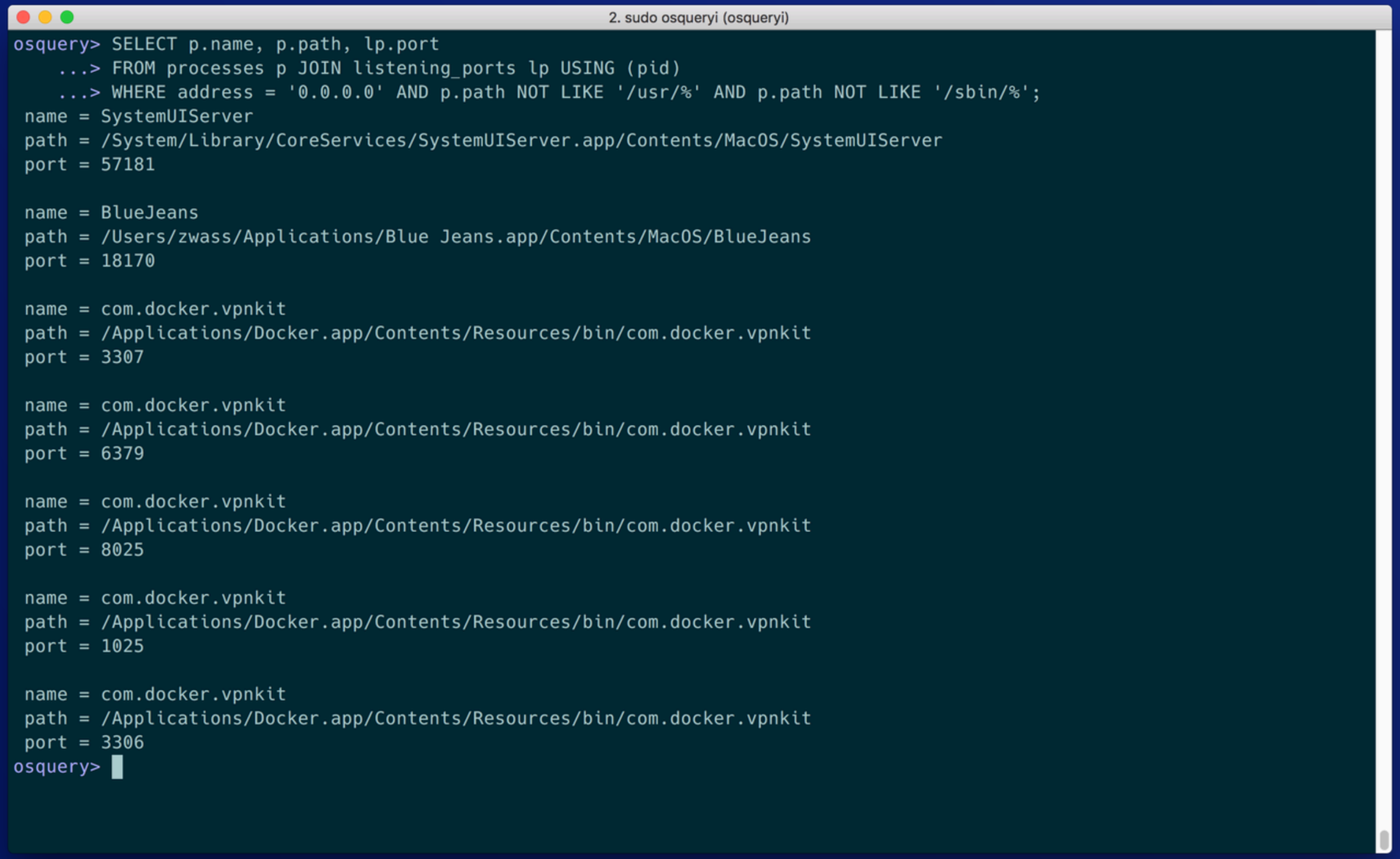
osqueryi



KOLIDE

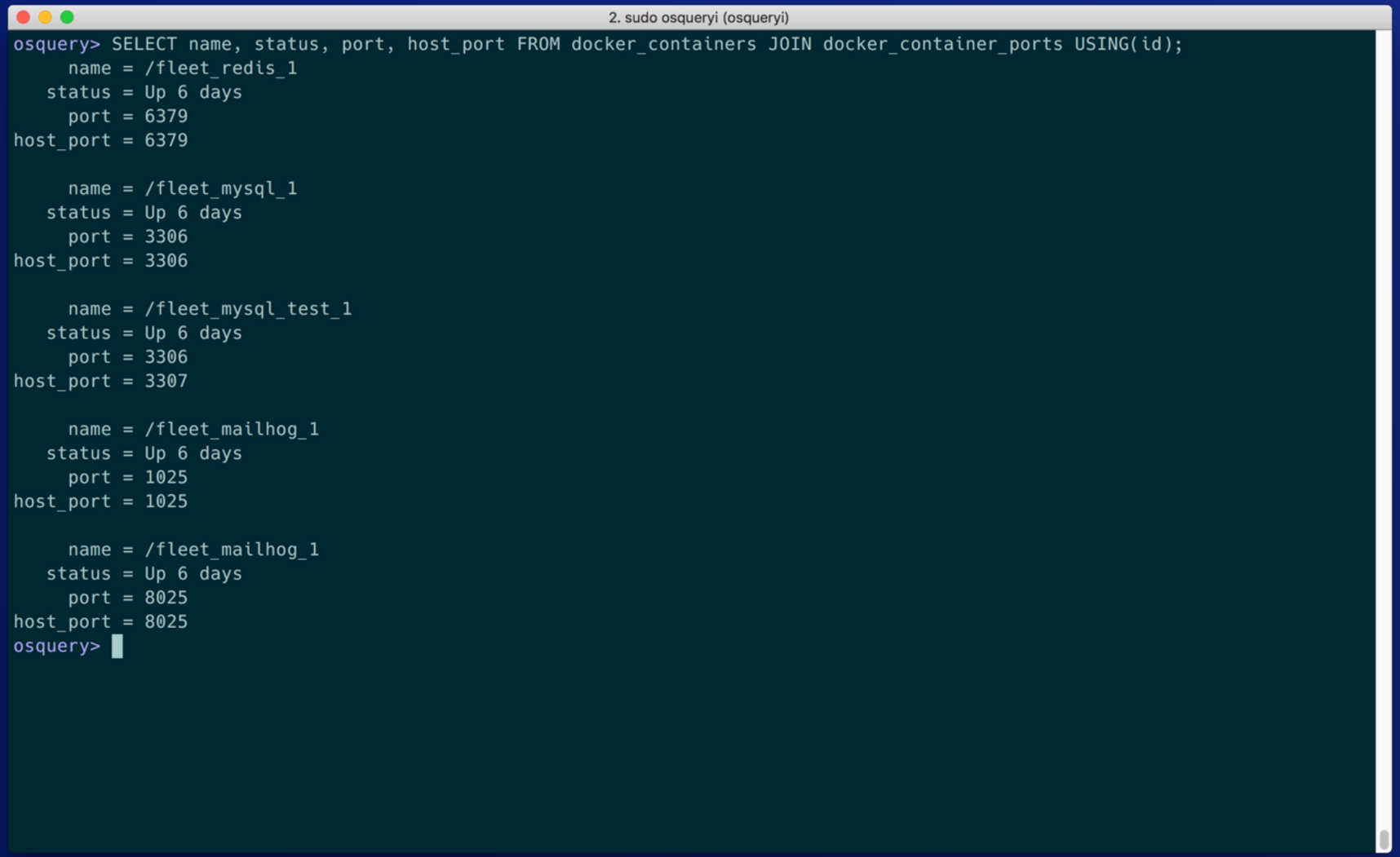
How can we interactively investigate system activity using osqueryi?

osqueryi



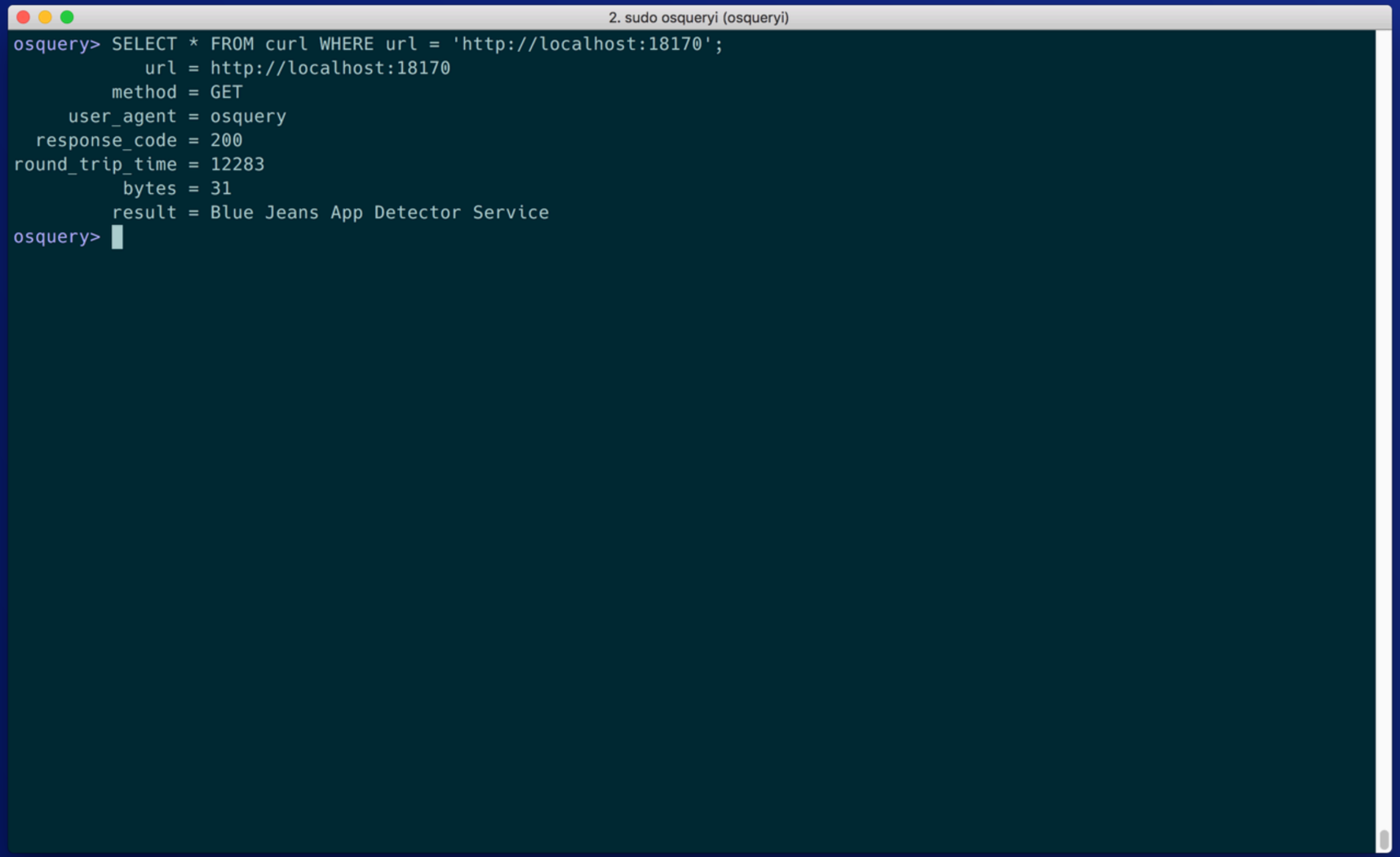
KOLIDE

osqueryi



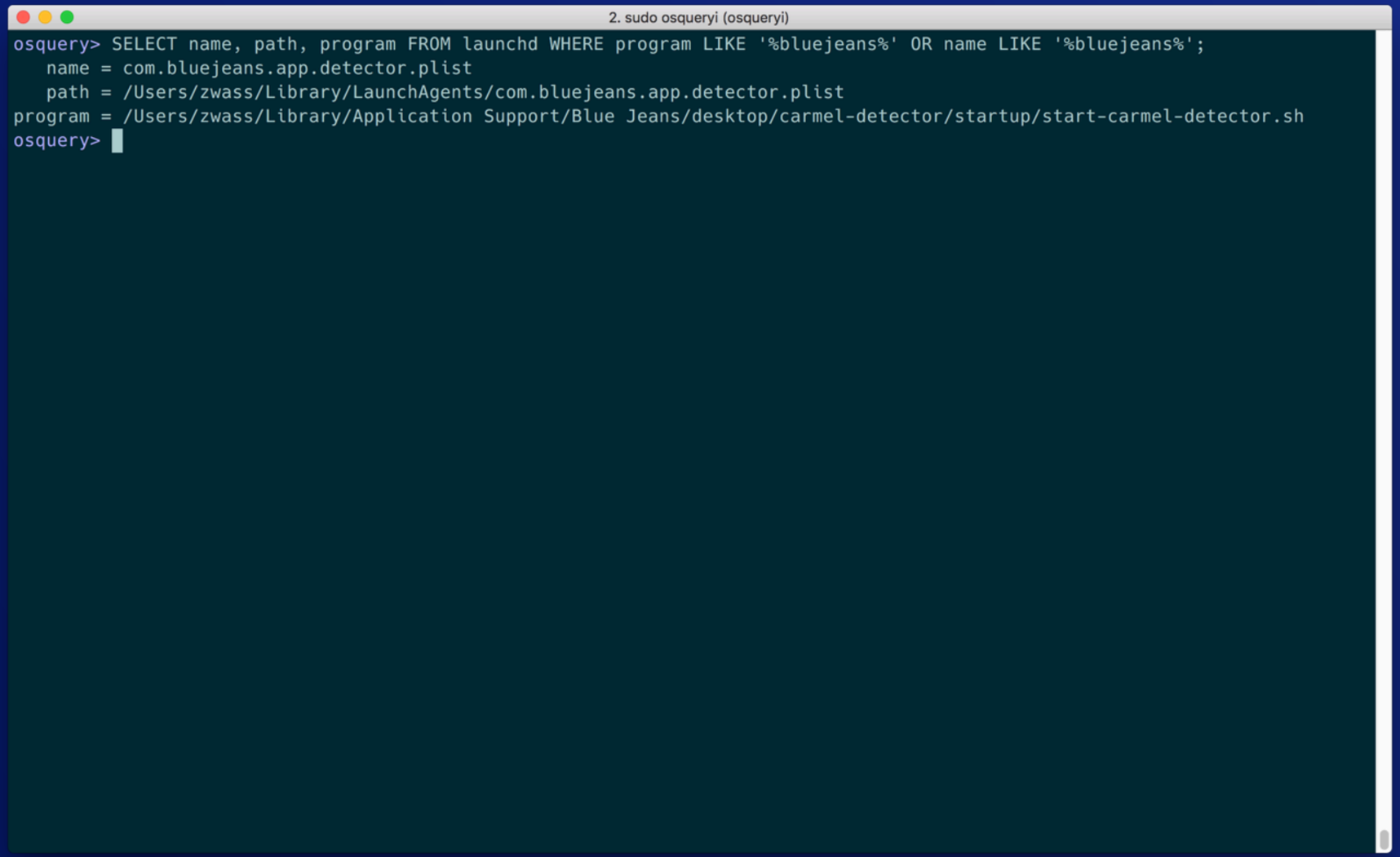
KOLIDE

osqueryi



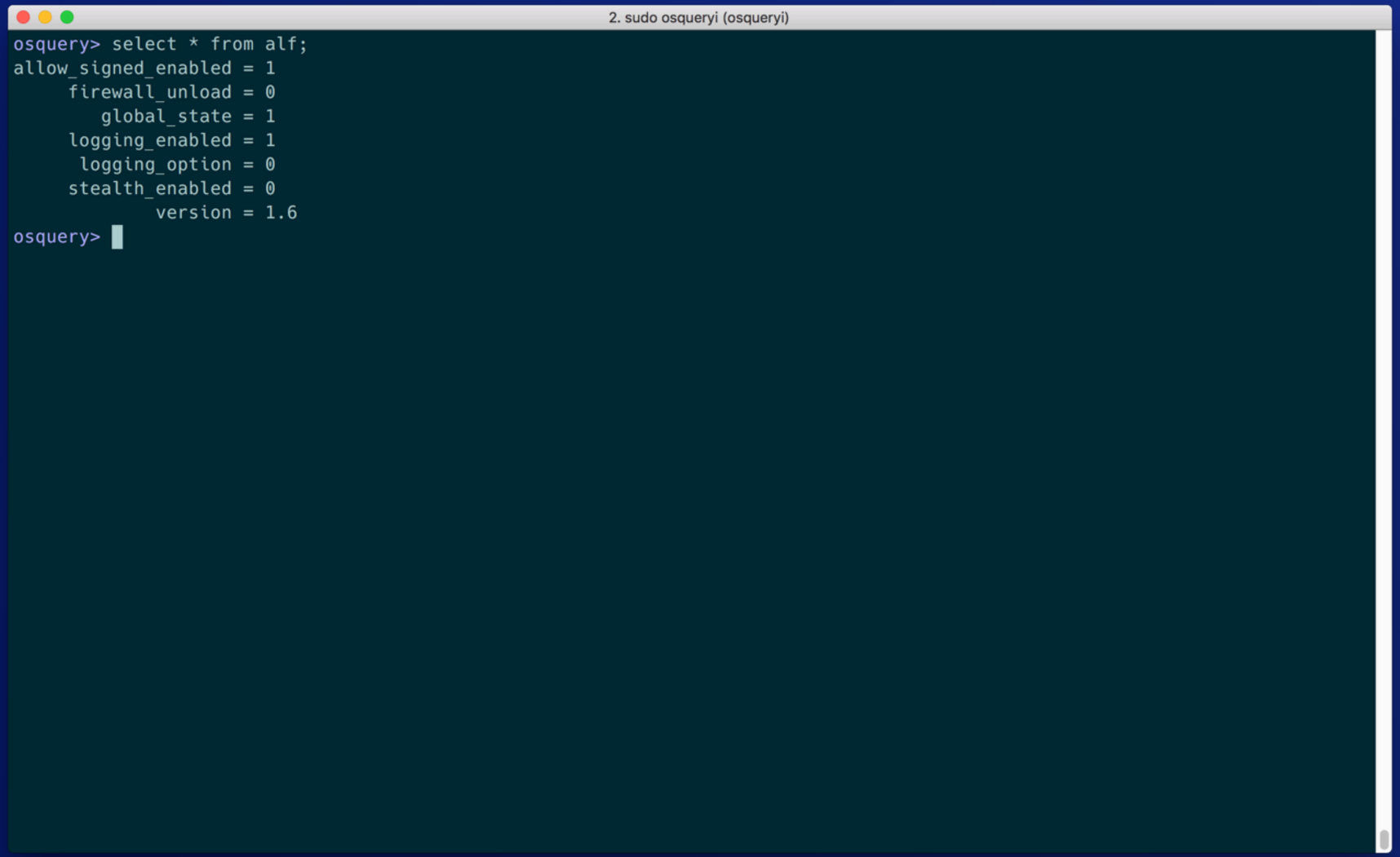
KOLIDE

osqueryi



KOLIDE

osqueryi

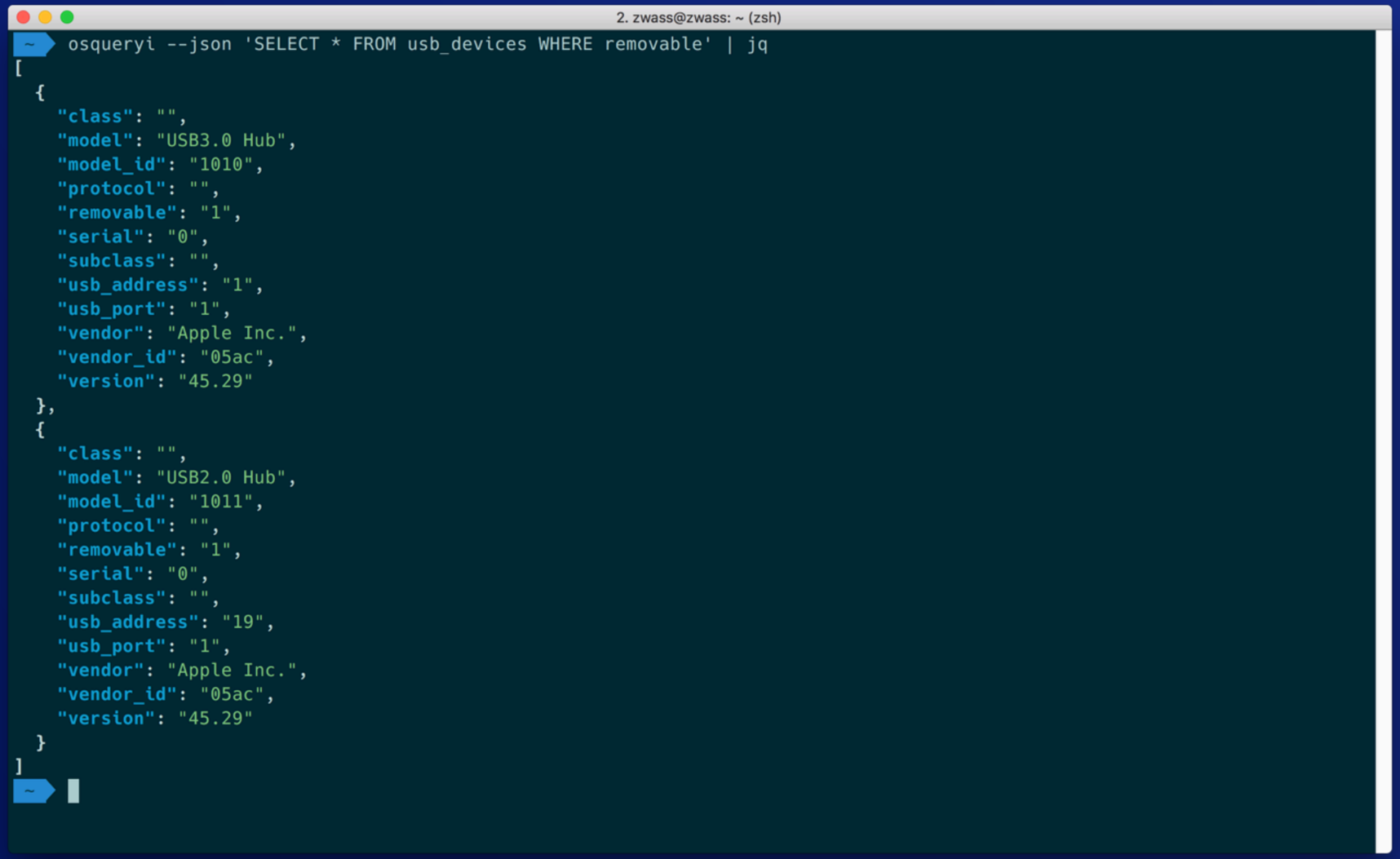


KOLIDE

Get structured output for scripting

KOLIDE

osqueryi



KOLIDE

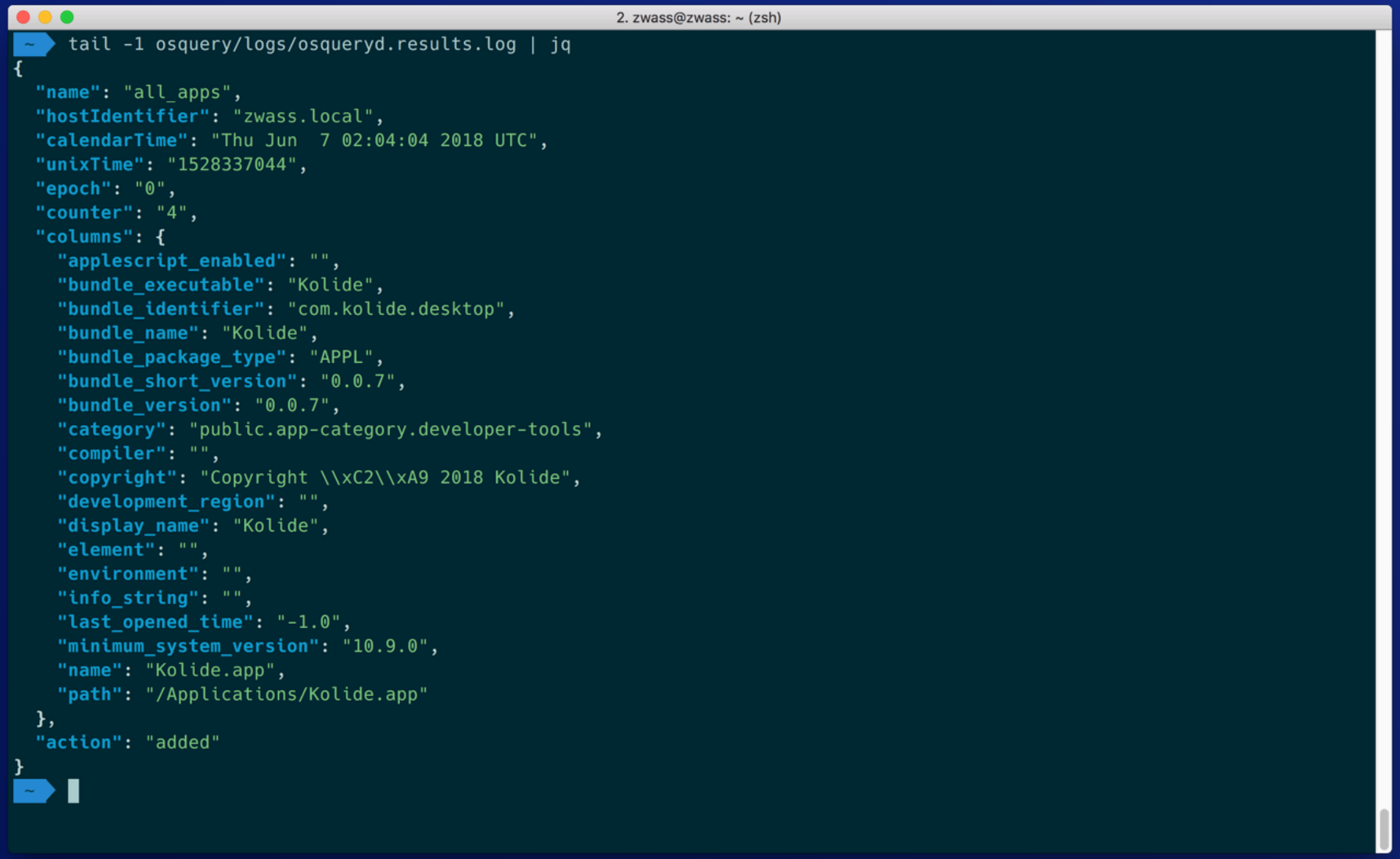
osqueryd

- Schedule queries for continuous results
- Differential engine to see how state changes over time
- Event-based tables ensure that data is not lost even when queries run on an interval

osqueryd

```
{
  "schedule": {
    "all_apps": {
      "query": "SELECT * FROM apps",
      "interval": 60
    }
  }
}
```

osqueryd

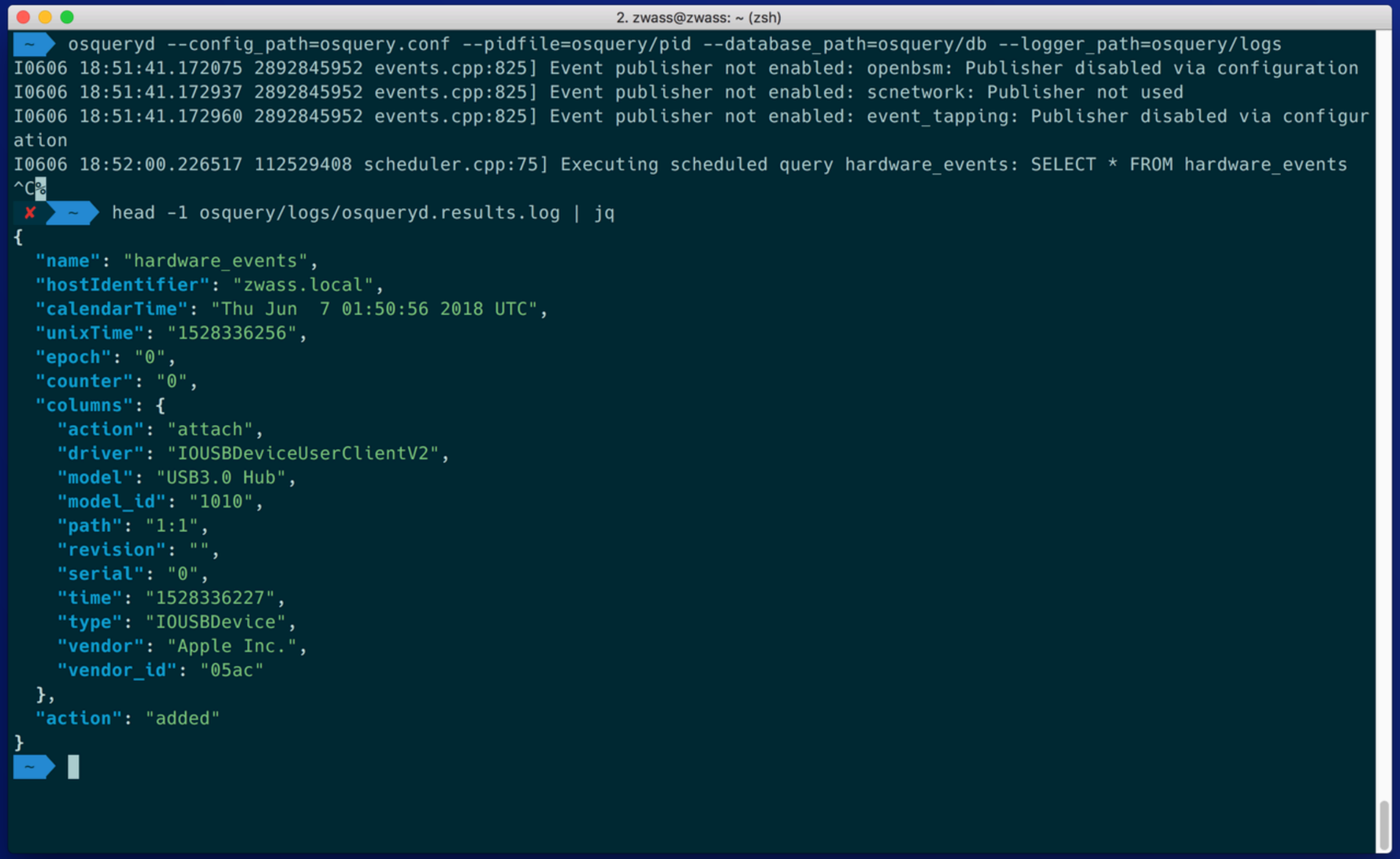


KOLIDE

osqueryd

```
{
  "schedule": {
    "hardware_events": {
      "query": "SELECT * FROM hardware_events",
      "interval": 60
    }
  }
}
```

osqueryd



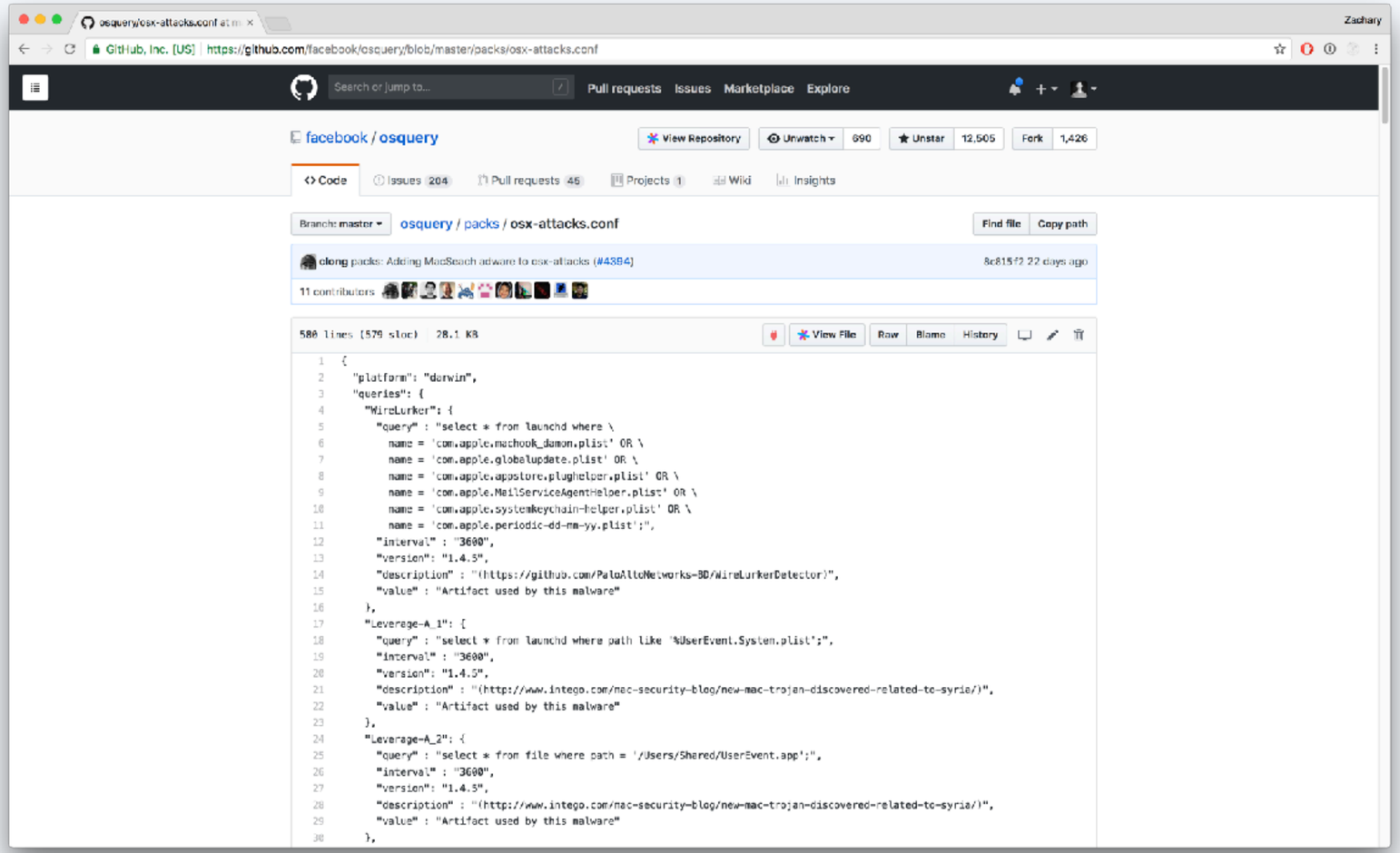
KOLIDE

What to do with all this power?

KOLIDE

What to do with all this power?

Check out the community-sourced query packs



KOLIDE

http://bit.ly/osx_attacks_pack

What to do with all this power?

Build custom tables for your organization

```go
package main

import (
	"context"
	"log"
	"os"

	"github.com/kolide/osquery-go"
	"github.com/kolide/osquery-go/plugin/table"
)

func main() {
	if len(os.Args) != 2 {
		log.Fatalf(`Usage: %s SOCKET_PATH`, os.Args[0])
	}

	server, err := osquery.NewExtensionManagerServer("foobar", os.Args[1])
	if err != nil {
		log.Fatalf("Error creating extension: %s\n", err)
	}

	server.RegisterPlugin(table.NewPlugin("foobar", FoobarColumns(), FoobarGenerate))
	if err := server.Run(); err != nil {
		log.Fatalln(err)
	}
}

func FoobarColumns() []table.ColumnDefinition {
	return []table.ColumnDefinition{
		table.TextColumn("foo"),
		table.TextColumn("baz"),
	}
}

func FoobarGenerate(ctx context.Context, queryContext table.QueryContext) ([]map[string]string, error) {
	return []map[string]string{
		{
			"foo": "bar",
			"baz": "baz",
		},
		{
			"foo": "bar",
			"baz": "baz",
		},
	}, nil
}
```

```python
#!/usr/bin/env python

import osquery

@osquery.register_plugin
class MyTablePlugin(osquery.TablePlugin):
    def name(self):
        return "foobar"

    def columns(self):
        return [
            osquery.TableColumn(name="foo", type=osquery.STRING),
            osquery.TableColumn(name="baz", type=osquery.STRING),
        ]

    def generate(self, context):
        query_data = []

        for _ in range(2):
            row = {}
            row["foo"] = "bar"
            row["baz"] = "baz"
            query_data.append(row)

        return query_data

if __name__ == "__main__":
    osquery.start_extension(name="my_awesome_extension", version="1.0.0")
```

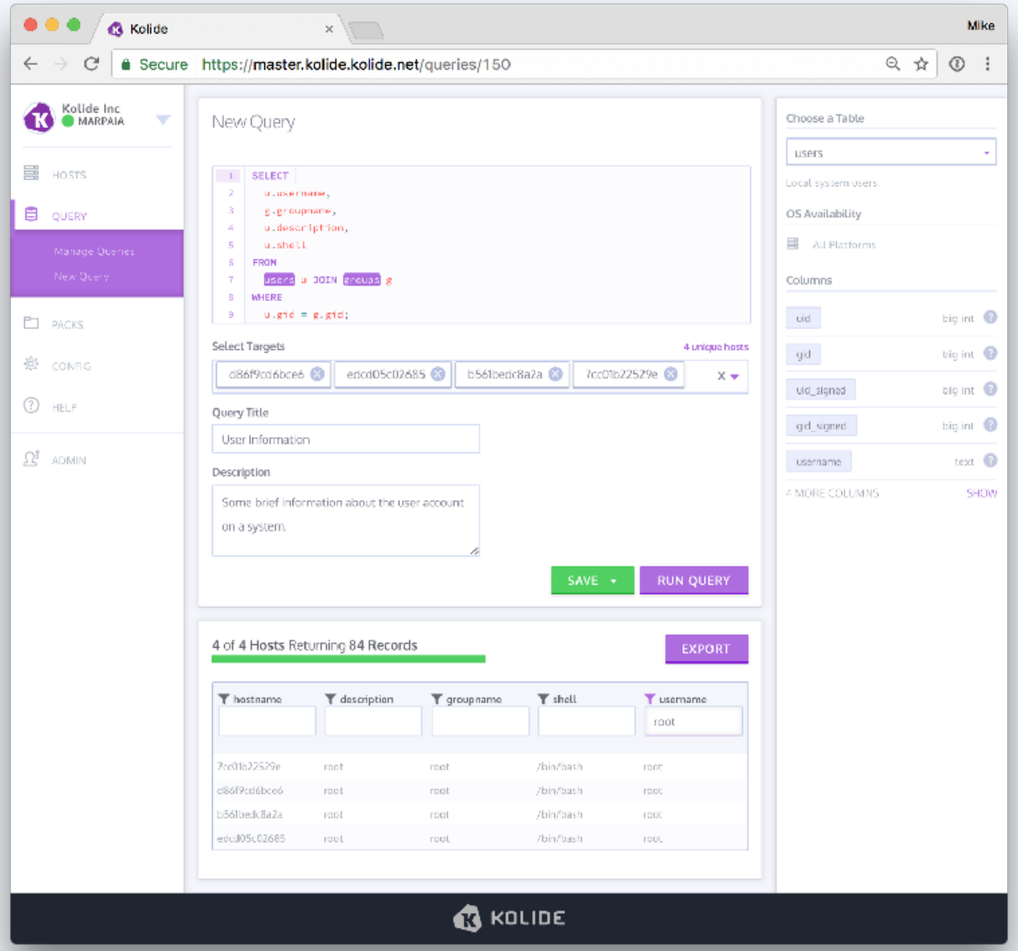
KOLIDE

https://github.com/kolide/osquery-go

https://github.com/osquery/osquery-python

What to do with all this power?

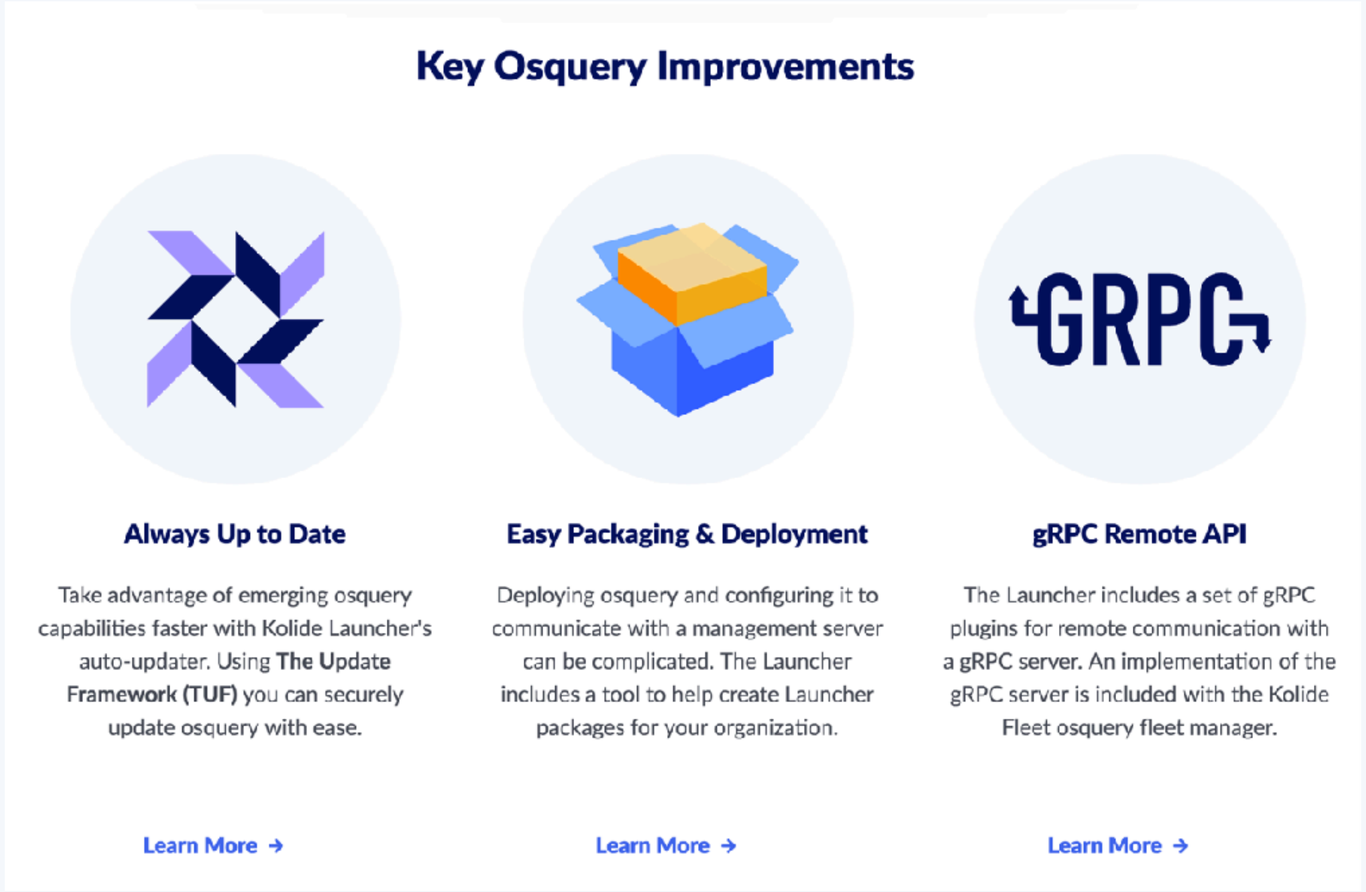
Implement a central management server



https://kolide.com/fleet

What to do with all this power?

Check out Kolide Launcher

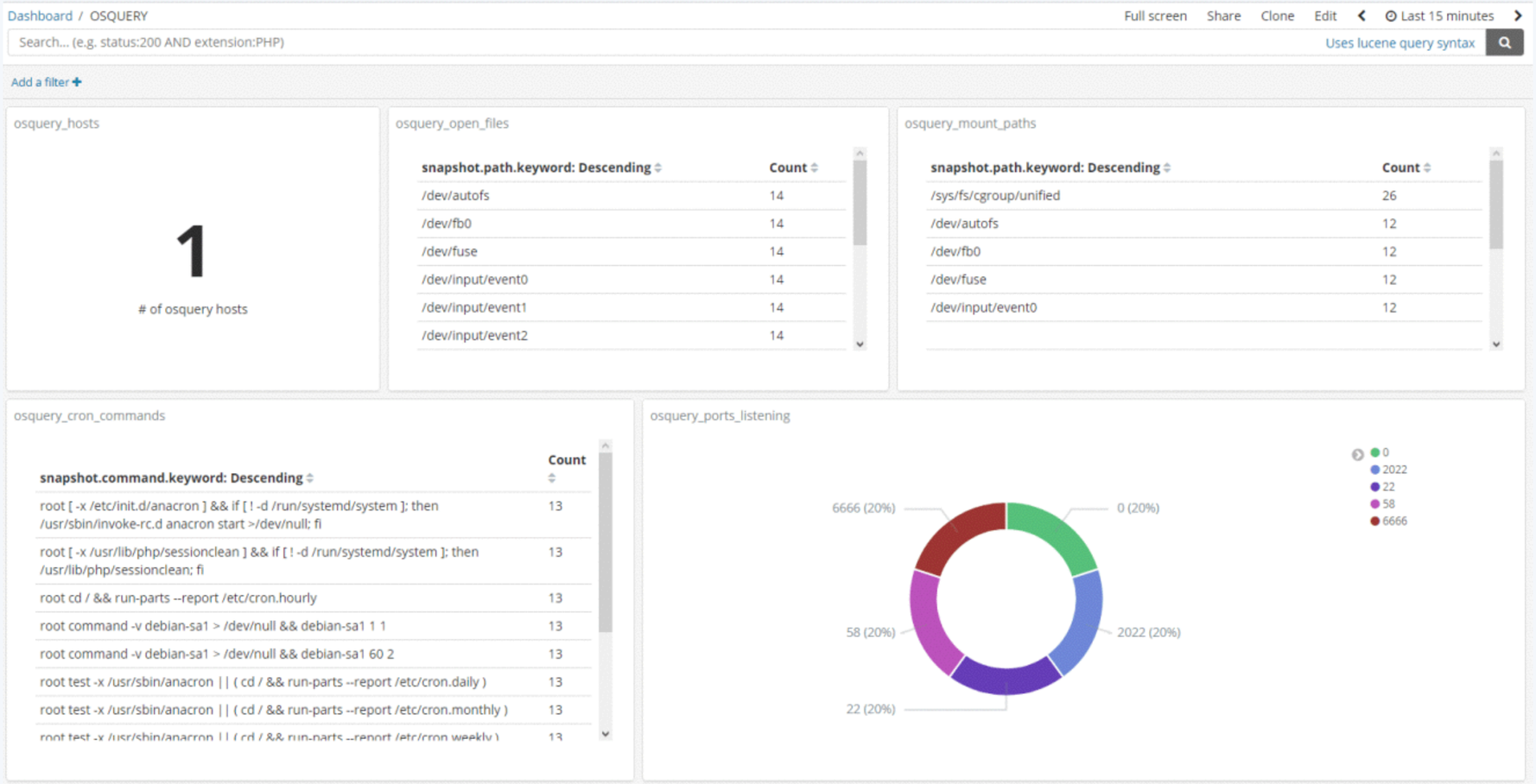


KOLIDE

https://kolide.com/launcher

What to do with all this power?

Push logs to ELK stack for dashboards, alerting and archiving



KOLIDE

http://bit.ly/elk_osquery_potti

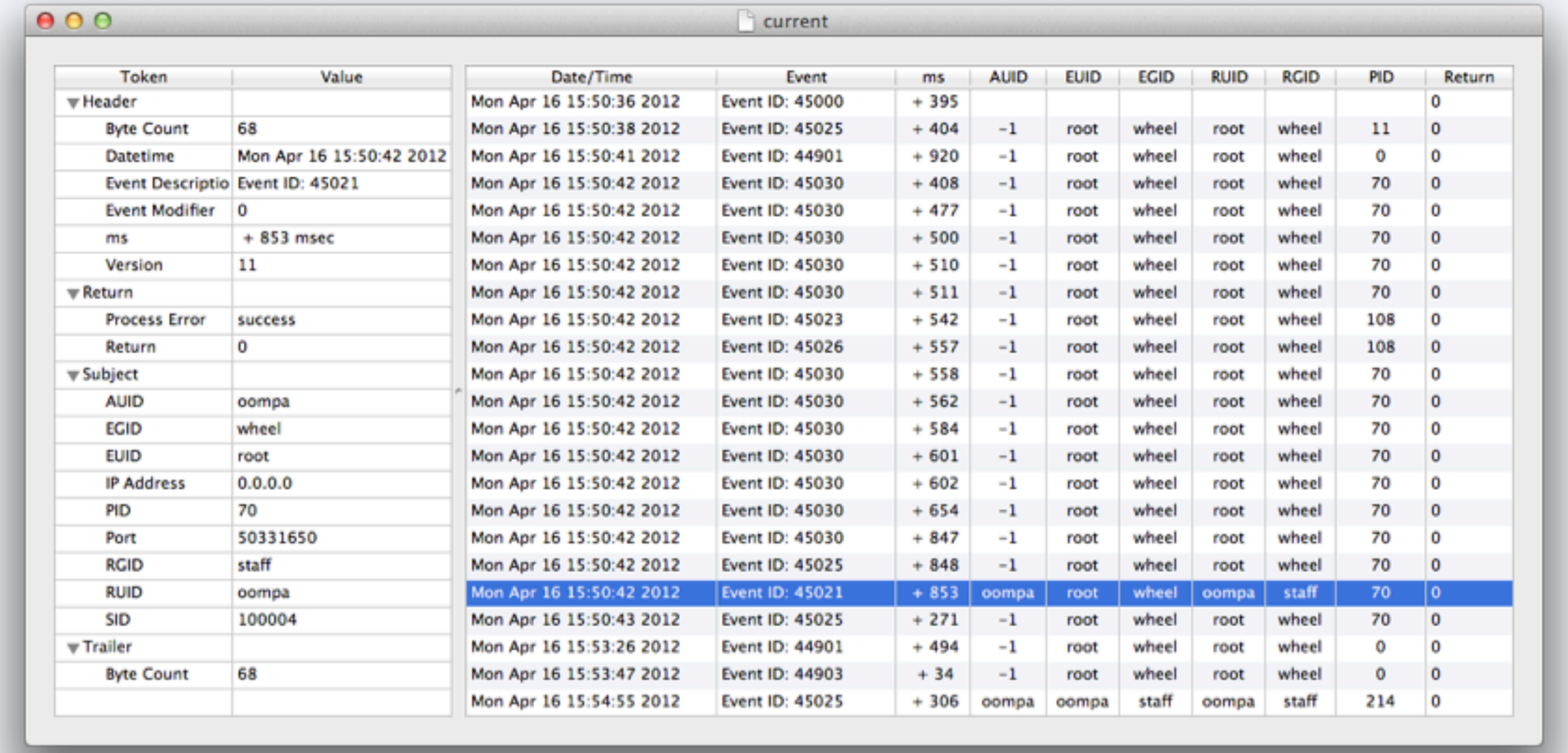
What to do with all this power?

Conditionally install software using Munki

http://bit.ly/osquery_munki_groob

What to do with all this power?

Process/Socket Auditing, File Integrity Monitoring



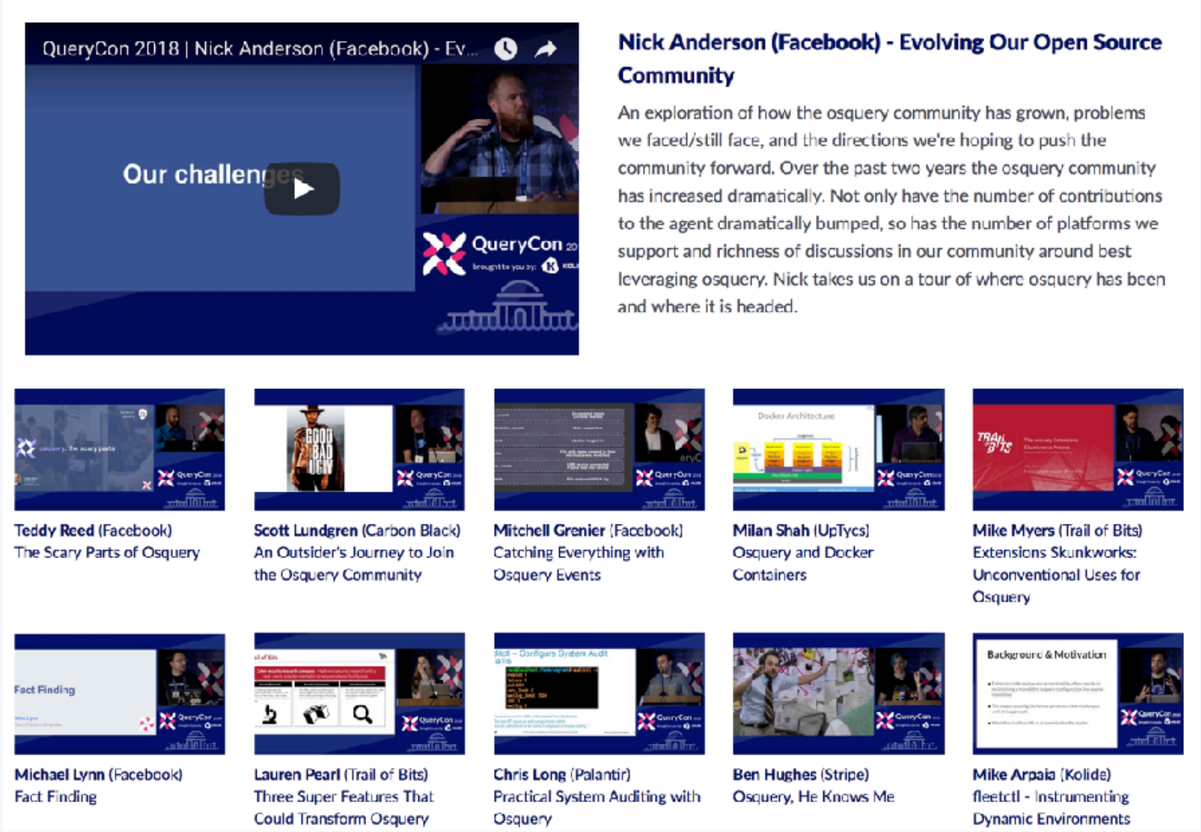
Token	Value
▼Header	
Byte Count	68
Datetime	Mon Apr 16 15:50:42 2012
Event Descriptio	Event ID: 45021
Event Modifier	0
ms	+ 853 msec
Version	11
▼Return	
Process Error	success
Return	0
▼Subject	
AUID	oompa
EGID	wheel
EUID	root
IP Address	0.0.0.0
PID	70
Port	50331650
RGID	staff
RUID	oompa
SID	100004
▼Trailer	
Byte Count	68

Date/Time	Event	ms	AUID	EUID	EGID	RUID	RGID	PID	Return
Mon Apr 16 15:50:36 2012	Event ID: 45000	+ 395							0
Mon Apr 16 15:50:38 2012	Event ID: 45025	+ 404	-1	root	wheel	root	wheel	11	0
Mon Apr 16 15:50:41 2012	Event ID: 44901	+ 920	-1	root	wheel	root	wheel	0	0
Mon Apr 16 15:50:42 2012	Event ID: 45030	+ 408	-1	root	wheel	root	wheel	70	0
Mon Apr 16 15:50:42 2012	Event ID: 45030	+ 477	-1	root	wheel	root	wheel	70	0
Mon Apr 16 15:50:42 2012	Event ID: 45030	+ 500	-1	root	wheel	root	wheel	70	0
Mon Apr 16 15:50:42 2012	Event ID: 45030	+ 510	-1	root	wheel	root	wheel	70	0
Mon Apr 16 15:50:42 2012	Event ID: 45030	+ 511	-1	root	wheel	root	wheel	70	0
Mon Apr 16 15:50:42 2012	Event ID: 45023	+ 542	-1	root	wheel	root	wheel	108	0
Mon Apr 16 15:50:42 2012	Event ID: 45026	+ 557	-1	root	wheel	root	wheel	108	0
Mon Apr 16 15:50:42 2012	Event ID: 45030	+ 558	-1	root	wheel	root	wheel	70	0
Mon Apr 16 15:50:42 2012	Event ID: 45030	+ 562	-1	root	wheel	root	wheel	70	0
Mon Apr 16 15:50:42 2012	Event ID: 45030	+ 584	-1	root	wheel	root	wheel	70	0
Mon Apr 16 15:50:42 2012	Event ID: 45030	+ 601	-1	root	wheel	root	wheel	70	0
Mon Apr 16 15:50:42 2012	Event ID: 45030	+ 602	-1	root	wheel	root	wheel	70	0
Mon Apr 16 15:50:42 2012	Event ID: 45030	+ 654	-1	root	wheel	root	wheel	70	0
Mon Apr 16 15:50:42 2012	Event ID: 45030	+ 847	-1	root	wheel	root	wheel	70	0
Mon Apr 16 15:50:42 2012	Event ID: 45025	+ 848	-1	root	wheel	root	wheel	70	0
Mon Apr 16 15:50:42 2012	Event ID: 45021	+ 853	oompa	root	wheel	oompa	staff	70	0
Mon Apr 16 15:50:43 2012	Event ID: 45025	+ 271	-1	root	wheel	root	wheel	70	0
Mon Apr 16 15:53:26 2012	Event ID: 44901	+ 494	-1	root	wheel	root	wheel	0	0
Mon Apr 16 15:53:47 2012	Event ID: 44903	+ 34	-1	root	wheel	root	wheel	0	0
Mon Apr 16 15:54:55 2012	Event ID: 45025	+ 306	oompa	oompa	staff	oompa	staff	214	0

http://bit.ly/advanced_osquery_clong

What to do with all this power?

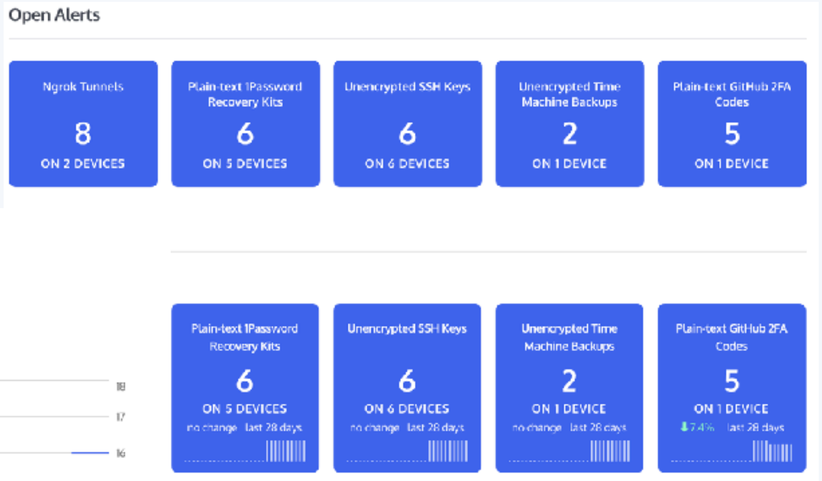
Get Inspired with ideas from QueryCon

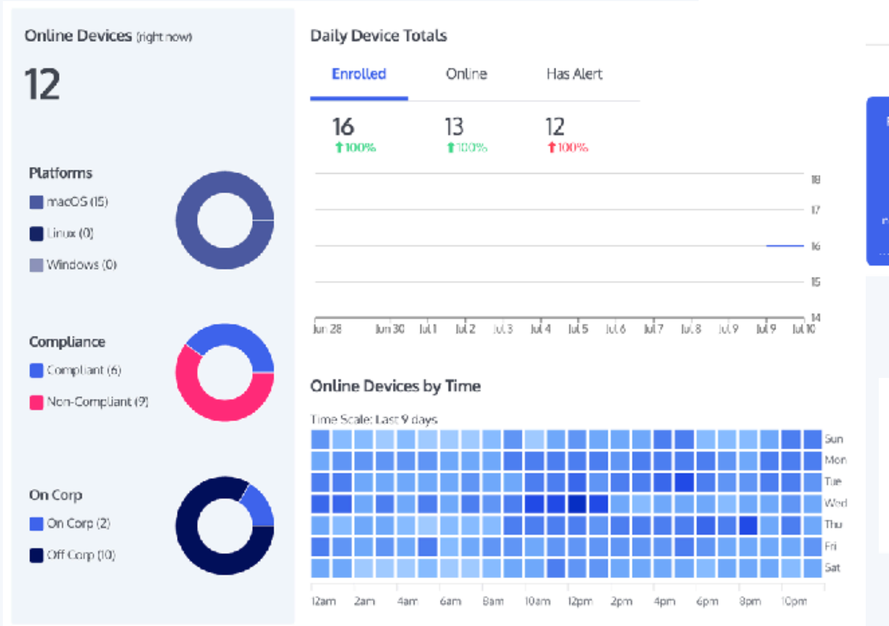


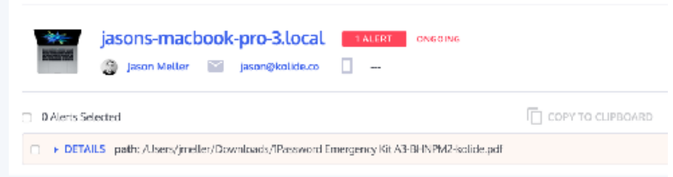
https://querycon.io/#videos

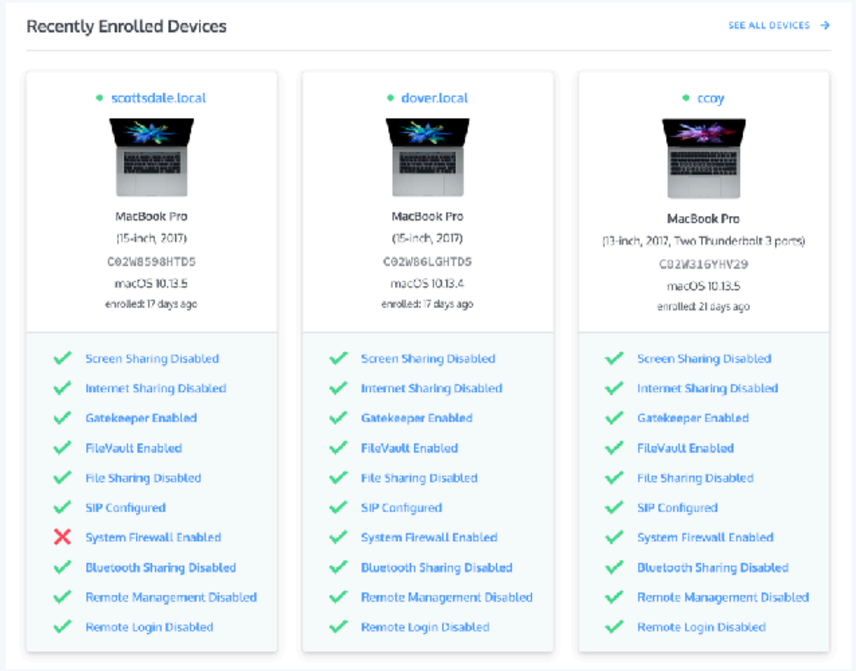
What to do with all this power?

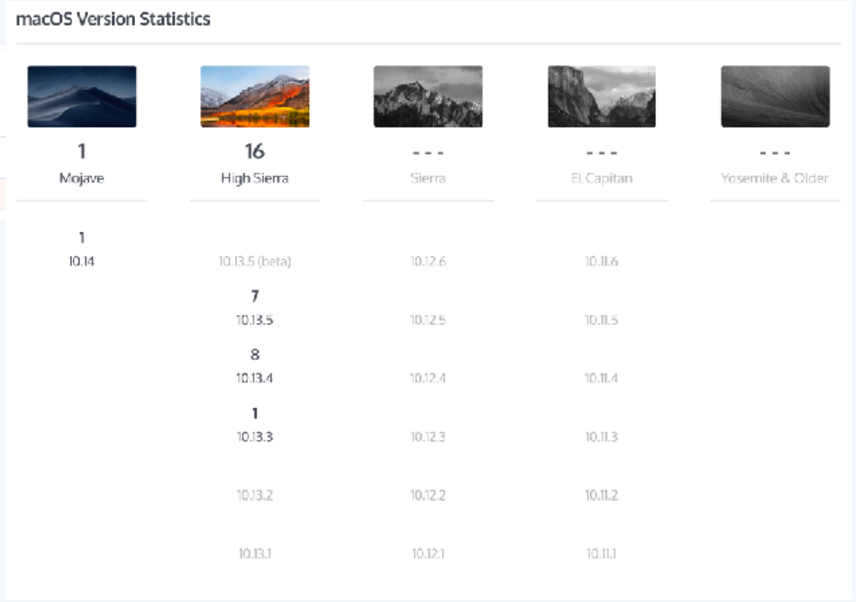
Kolide Cloud











https://kolide.com

KOLIDE

Two paths with osquery

Configuration management,
Log aggregation,
Alerting,
etc.

Direct to insights with commercial tools (Kolide Cloud)

Build it Yourself with OSS tools (Kolide Fleet, Zentral)

Two paths with osquery @ PSUMacAdmins

- The commercial straight-to-insights approach
- “How Software Engineers Ruin Macs and Could Ruin You” - Jason Meller (Kolide)
- Thursday July 12, 9:00-10:15am

- The DIY open-source approach with Kolide’s OSS tools
- “Using osquery via Fleet for client/server visibility” - Lucas Hall (Saturna Capital)
- Thursday July 12, 3:15-4:30pm

Join us in osquery Slack

bit.ly/osquery_slack

StackOverflow: #osquery

KOLIDE

Thank you!

 **zach** @ kolide.com

 github.com / **zwass**

 **zwass** @ osquery Slack

 twitter.com / **thezachw**