



USER SESSION RECORDING FOR THE ENTERPRISE

An Open-Source Effort by Red Hat

Nikolai Kondrashov
Software Engineer
FrOSCon 2017

INTRODUCTION

Nikolai “spbnick” Kondrashov, a software engineer

- Working at Red Hat Common Logging team
- Focusing on the User Session Recording project
- Maintaining FreeRADIUS packages
- Founder and (still) maintainer of the DIGImend project
- Flirting with embedded as a hobby

WHY?

THERE IS A DEMAND

Customers have been telling us for a long time:

- We need to comply with government regulations
- We need to track what contractors do on our systems
- We need to know who broke our server and how

AND A DREAM

What people and governments want:

- Record everything users do
- Store that somewhere safe
- Let us find who did that thing
- Show us how they did it

THERE IS A SUPPLY

A great number of commercial offerings:

- From application-level proxies on dedicated hardware
- To user-space processes on the target system
- Recording keystrokes, display, commands, apps, URLs, etc.
- Integrated with identity management, and access control
- With central storage, searching, and playback

BUT NOT GOOD ENOUGH

Still people are not satisfied:

- Expensive
- Sometimes very expensive
- Can't fix it yourself
- Can't improve it yourself

WHAT CAN BE BETTER?

The customers want:

- Free (as in beer)
- Open-Source, so they can fix, or at least understand it better
- Yet still has support

WAIT, WE HAVE IT ALREADY!

Nope, not really:

- script(1) plus duct tape
 - popular, but not security-oriented, needs lots of DIY
- sudo(8) I/O logging
 - security-oriented, has searching, but not centralized
- TTY audit with auditd(8)
 - security-oriented, can be centralized, but only for input

WHAT?

SO, WHAT DO WE NEED?

Hottest features requested:

- Record what the user enters, sees on the screen, executes, accesses
- Get it off the machine *ASAP*, and store centrally and securely
- Search, analyze, and correlate with other events
- Playback in real time, or later
- Control centrally

SOUNDS FAMILIAR!

Let's do it with logs!

- Audit system records processes executed, files accessed
- Logging servers know how to deliver
- There's a whole zoo of storing/searching/analyzing solutions

LEAN AND MEAN

Why it's better:

- Reuses log plumbing
 - No separate infrastructure needed
 - Saves resources
 - Reduces maintenance load
- Allows easy correlation with all the other logs
 - Not just an isolated “video of the terminal”
 - Lets you see what was behind the scenes

FIRST...

What to take out of the store/search/analyze zoo?

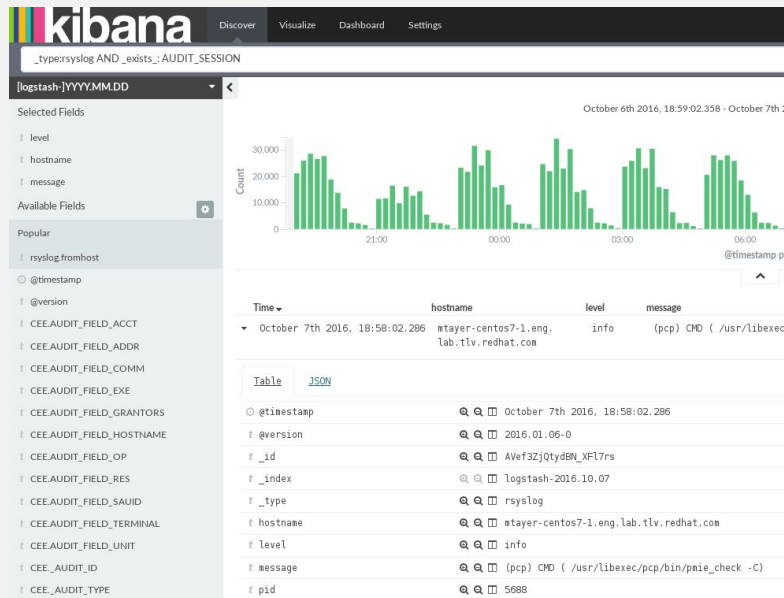
- Open-Source
- Scalable
- Hip

YES, ELASTICSEARCH AND KIBANA!

Our **ViaQ** project is bringing them to Red Hat product portfolio:

<https://github.com/ViaQ>

- Normalize all the logs
- Put them into Elasticsearch
- Provide dashboards and analytics
- Part of OpenShift, coming to OpenStack and other Red Hat products!



THEN...

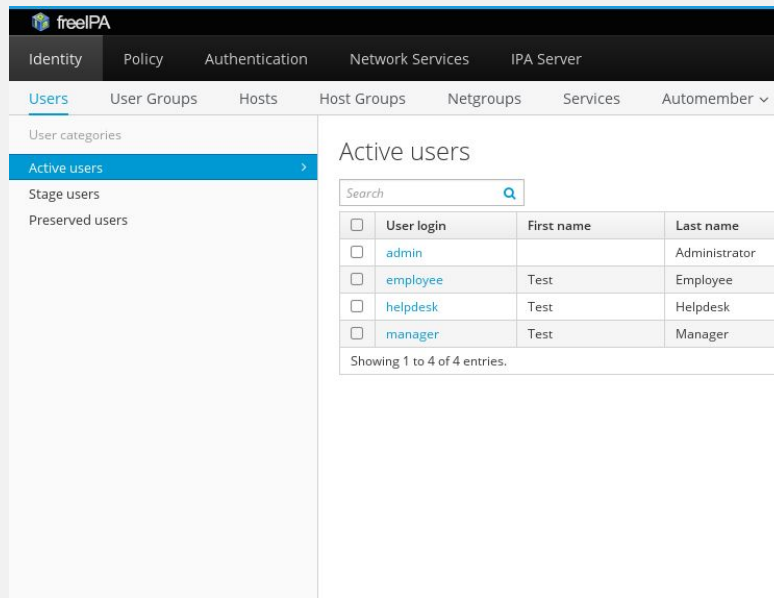
How can we:

- Control centrally what, where and whom to record?
- Log what user enters and sees?
- Make sense of audit logs?
- Deliver to Elasticsearch?
- Play everything back?

CENTRAL CONTROL?

Naturally, FreeIPA and SSSD!

- Manage domains, hosts, groups, users, and more
- Cache credentials and authenticate offline
- Session Recording control being designed



The screenshot displays the FreeIPA web interface. The top navigation bar includes 'Identity', 'Policy', 'Authentication', 'Network Services', and 'IPA Server'. Below this, a secondary navigation bar shows 'Users', 'User Groups', 'Hosts', 'Host Groups', 'Netgroups', 'Services', and 'Automember'. The 'Users' section is active, and the 'Active users' category is selected in the left sidebar. The main content area shows a search bar and a table of active users.

<input type="checkbox"/>	User login	First name	Last name
<input type="checkbox"/>	admin		Administrator
<input type="checkbox"/>	employee	Test	Employee
<input type="checkbox"/>	helpdesk	Test	Helpdesk
<input type="checkbox"/>	manager	Test	Manager

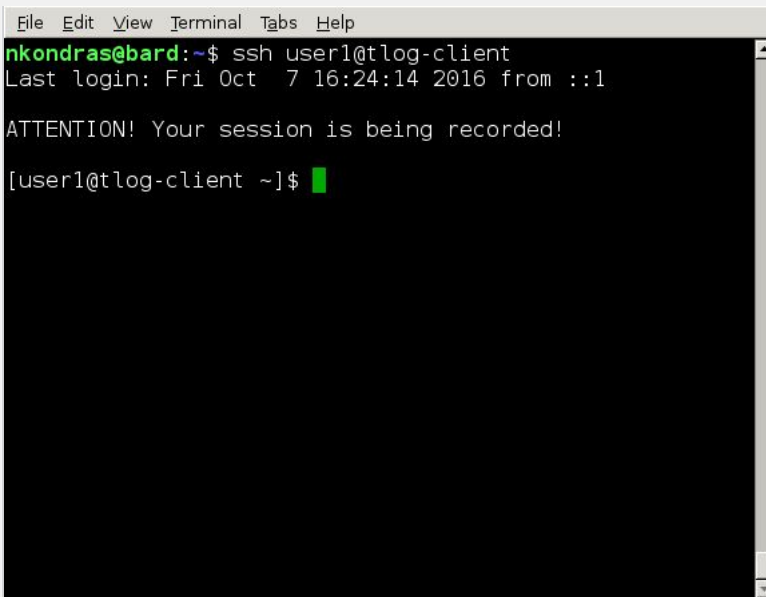
Showing 1 to 4 of 4 entries.

LOG INPUT AND OUTPUT?

We made a tool for that - **tlog**

<http://scribery.github.io/tlog>

- A shim between the terminal and the shell, started at login
- Converts what passes in between to searchable JSON
- Logs to syslog or journal
- Plays back recordings on a terminal



```
File Edit View Terminal Tabs Help
nkondras@bard:~$ ssh user1@tlog-client
Last login: Fri Oct 7 16:24:14 2016 from ::1

ATTENTION! Your session is being recorded!

[user1@tlog-client ~]$ █
```

MAKE SENSE OF AUDIT LOGS?

We made a tool for that too - **aushape**

<http://scribery.github.io/aushape/>

- Listens for audit events
- Converts them to JSON or XML
- Both have official schemas
- Logs to syslog
- Developed with the help from auditd

```
File Edit View Terminal Tabs Help
sh-4.3#
sh-4.3# pwd
/root
sh-4.3# ps cf -C auditd,audispd,aushape | grep '|au.*'
  PID TTY          STAT       TIME COMMAND
   540 ?        S<s1      0:00  auditd
   550 ?        S<s1      0:00  \_  audispd
   552 ?        S<        0:00  \_  aushape
sh-4.3#
```

DELIVER TO ELASTICSEARCH?

Any popular logging service:



RSYSLOG*



Or our coming solution:

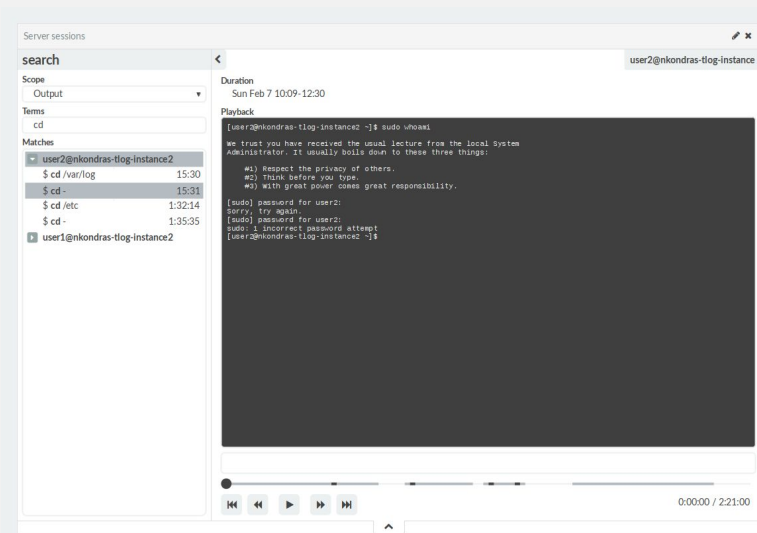
ViaQ

* Distributed by Red Hat now

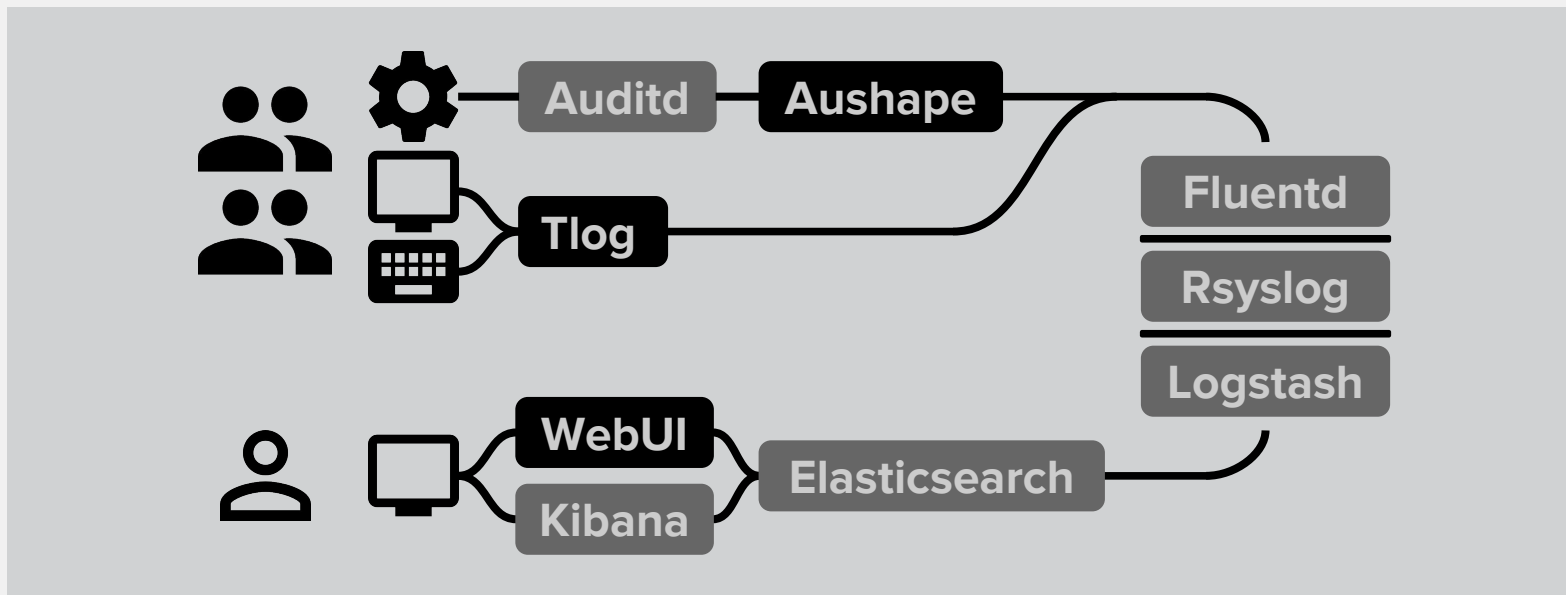
PLAY EVERYTHING BACK?

We're building a Web UI

- Playback data from Elasticsearch
- See input, output, commands executed and files accessed
- Search for input, output, commands and files
- Reuse and integrate
- Starting in [Cockpit](#) with storage in journal



ALL TOGETHER NOW!



DEMO!

IN THIS DEMO...

- A recorded user logs in
- Playback of the session is started at the same time
- Some work is done on the terminal
- Terminal I/O and converted audit logs are seen in journal
- Logs in Elasticsearch are displayed by Kibana
- Guest appearance: recordings in Cockpit

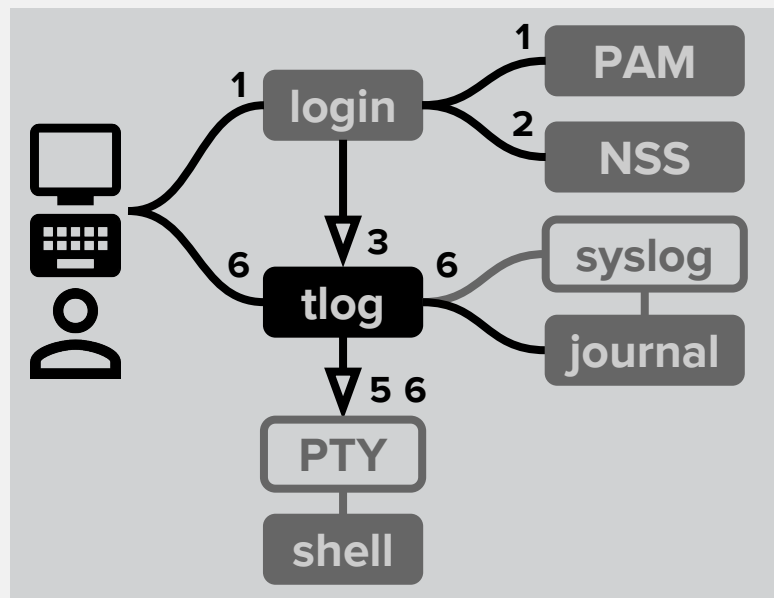
HOW?

HOW TLOG WORKS?

Console login example

Starting a console session:

1. User authenticates to **login** via **PAM**
2. **NSS** tells **login**: **tlog** is the shell
3. **login** starts **tlog**
4. Env/config tell **tlog** the actual shell
5. **tlog** starts the actual shell in a **PTY**
6. **tlog** logs everything passing between its **terminal** and the **PTY**, via **syslog(3)** or **sd-journal(3)**

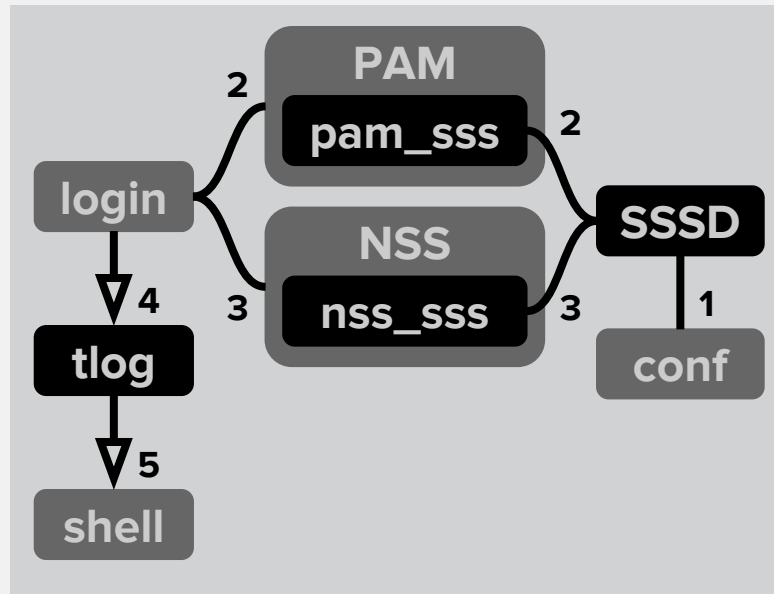


CONTROL TLOG WITH SSSD

Console login example

When a recorded user logs in:

1. **SSSD** finds a match for the user in its **configuration**
2. **pam_sss** stores the actual user **shell** in the **PAM** environment
3. **nss_sss** tells **login: tlog** is the shell
4. **login** starts **tlog** with **PAM** environment
5. **tlog** starts the actual user **shell** retrieved from environment



CONTROL TLOG WITH FREEIPA

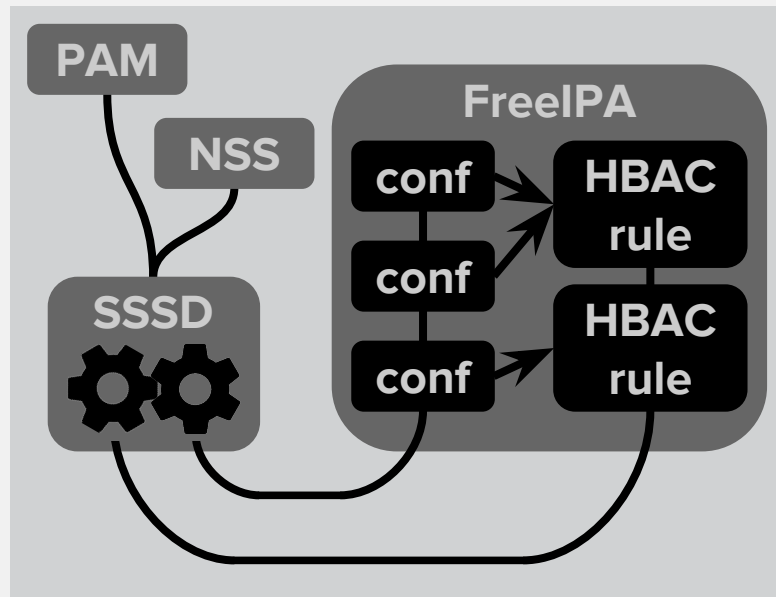
Plan so far

Which users to record on which hosts:

- Recording **configurations** are linked to **HBAC** rules, like SELinux maps

When users login:

- **SSSD** fetches applicable rules
- **SSSD** decides if recording is enabled
- Proceeds as on previous slide



EXTRA TLOG FEATURES

Also control:

- What to record: input/output/window resizes
- “You are being recorded” notice
- Where to write: sd-journal(3), syslog(3), or file
- Low latency vs. low overhead

Basic playback on the terminal:

- From Elasticsearch
- From Journal
- From file

TLOG SCHEMA

Optimized for streaming and searching:

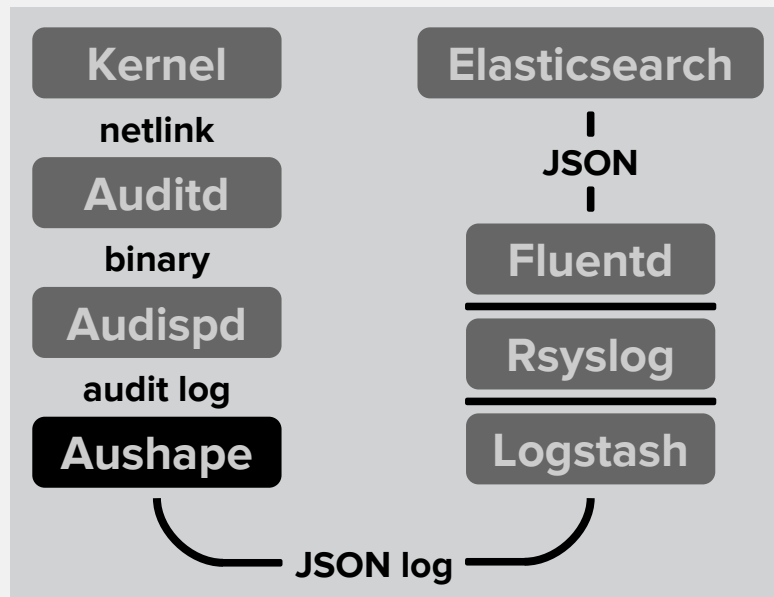
- Chopped into messages for streaming, but can be merged
- Input and output stored separately
- All I/O preserved
- Invalid UTF-8 stored separately
- Timing separate, ms precision
- Window resizes preserved

```
{
  "ver"      : "2.1",
  "host"     : "tlog-client.example.com",
  "rec"      : "c8aa248c81264f5d98d1..."
  "user"     : "user1",
  "term"     : "xterm",
  "session"  : 23,
  "id"       : 1,
  "pos"      : 0,
  "timing"    : "=56x22+98>23",
  "in_txt"   : "",
  "in_bin"   : [ ],
  "out_txt"  : "[user1@tlog-client ~]$ ",
  "out_bin"  : [ ]
}
```

HOW AUSHAPE WORKS?

From the kernel to Elasticsearch:

- **Kernel** sends messages to **auditd**
- **Auditd** passes messages to **auditpd**
- **Auditpd** distributes them to plugins, including **aushape**
- **Ausshape** formats JSON
- **Ausshape** logs it through **syslog(3)**
- **Fluentd/Rsyslog/Logstash** deliver it to **Elasticsearch**



AUSHAPE SCHEMAS

Mimicking the audit log, XML and JSON are similar, raw log can be preserved

```
<log>
  <event serial="number"
        time="timestamp">
    <text>
      <line>log message</line> ...
    </text>
    <data>
      <record>
        <field i="value" r="value"/> ...
      </record> ...
    </data>
  </event> ...
</log>
```

```
[
  {
    "serial": number,
    "time": "timestamp",
    "text": [
      "log message", ...
    ],
    "data": {
      "record": {
        "field": ["value", "value"], ...
      }, ...
    }
  }, ...
]
```


AUSHAPE EXAMPLES

A heavily-trimmed event

```
<event serial="880"
  time="2016-09-28T19:34:44.771+03:00">
  <data>
    <syscall>
      <syscall i="execve" r="59"/>
      <success i="yes"/>
    </syscall>
    <cwd>
      <cwd i="/home/user"/>
    </cwd>
    <execve>
      <a i="ps"/>
    </execve>
  </data>
</event>
```

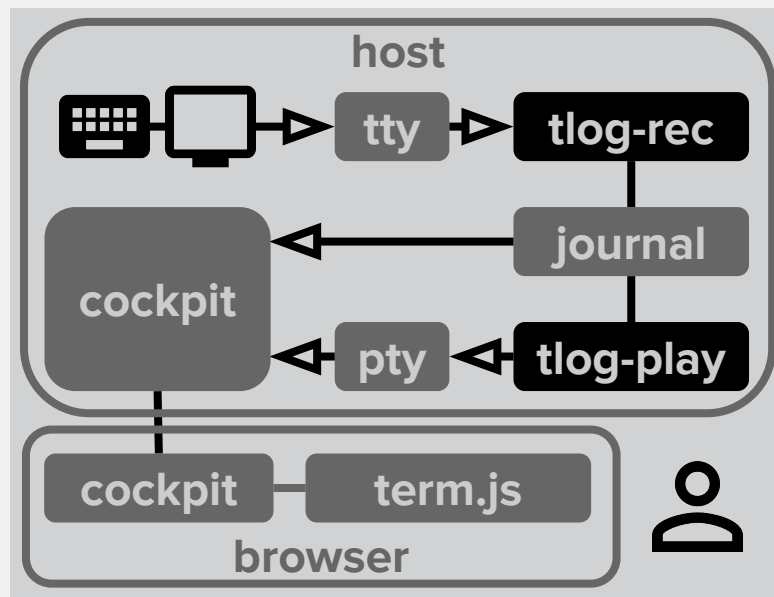
```
{
  "serial":880,
  "time":"2016-09-28T19:34:44.771+03:00",
  "data":{
    "syscall":{
      "syscall":["execve","59"],
      "success":["yes"]
    },
    "cwd":{
      "cwd":["/home/user"]
    },
    "execve":[
      "ps"
    ]
  }
}
```

HOW COCKPIT UI WORKS?

This is an early proof-of-concept

Setup for recordings in Cockpit:

- **Tlog** logs to **Journal**, adding a **recording ID** field
- To list recordings, **Cockpit** looks for **tlog** messages in journal, groups by **recording ID**
- To play a recording, **Cockpit** asks **tlog** on the host to play from **Journal** with **recording ID**, shows it in a JS-based terminal emulator

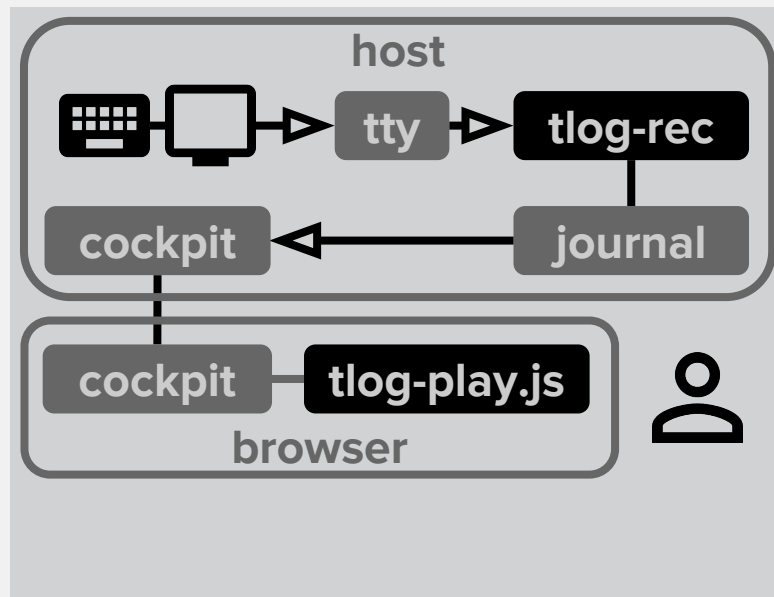


HOW COCKPIT UI WILL WORK?

Getting rid of playback on host

Setup for recordings in Cockpit:

- Logging and listing recordings works the same
- Playback is done fully in the browser, in a customized JS-based terminal emulator



CHALLENGES!

TLOG CHALLENGES

Lots of fun problems:

- How not to record passwords
 - Detect “echo off” mode, or cooperate with TTY audit
- Detect graphical sessions and don't record under them
 - Perhaps look at environment variables
- Support charset conversion
 - Use iconv, and keep original text
- Playback controls
 - Play/pause, fast-forward, rewind

AUSHAPE CHALLENGES

Some more fun (and not so fun) problems:

- Audit log is a mess
 - Can't fix. Track all the cases, use what auditd knows
- Somehow generate coherent schemas
 - Keep schema simple, use auditd record/field dictionaries
- Convert character encodings
 - Iconv, and keep invalid text in base64 or discard

WEB UI CHALLENGES

We're taking them to Cockpit Hackfest!

On the road to first release for Cockpit:

- Journal as a storage
 - Risky
- On-host playback control
 - Interesting, but difficult task
- Correlation with audit logs
 - It's about time

TRY IT!

TRY TLOG!

<https://github.com/Scribery/tlog>

- Download and install a release RPM, or
- Build from source, dependencies:
 - `json-c-devel / libjson-c-dev`
 - `libcurl-devel / libcurl4-* -dev`
 - `systemd-devel/libsystemd-journal-dev`
- Log to and playback from file
 - Easiest, good for testing
- Log to and playback from Elasticsearch
- Instructions in README.md!
- Submit issues, suggestions and pull requests!

TRY AUSHAPE!

<https://github.com/Scribery/aushape>

- Download and install a release RPM, or
- Build from source
 - Only `audit-libs-devel` / `libauparse-dev` is required
- Convert your own `/var/log/audit/audit.log` single-shot
 - Try both JSON and XML
- Set up live forwarding to Elasticsearch
- Instructions in README.md!
- Submit issues, suggestions and pull requests!

TRY COCKPIT UI!

https://github.com/Scribery/cockpit/tree/scribery_poc

- Checkout our [scribery_poc](#) branch
- Build and run from source
 - Read [HACKING.md](#)
- Install [tlog](#)
- Set writer to “journal” in `/etc/tlog/tlog-rec-session.conf`
- Create a user with shell set to `/usr/bin/tlog-rec-session`
- Login as that user and do some stuff
- Checkout “Session Recording” page at <http://localhost:9090>



THANK YOU



User Session Recording Project
<http://scribery.github.io/>



THANK YOU



plus.google.com/+RedHat



facebook.com/redhatinc



linkedin.com/company/red-hat



twitter.com/RedHatNews



youtube.com/user/RedHatVideos