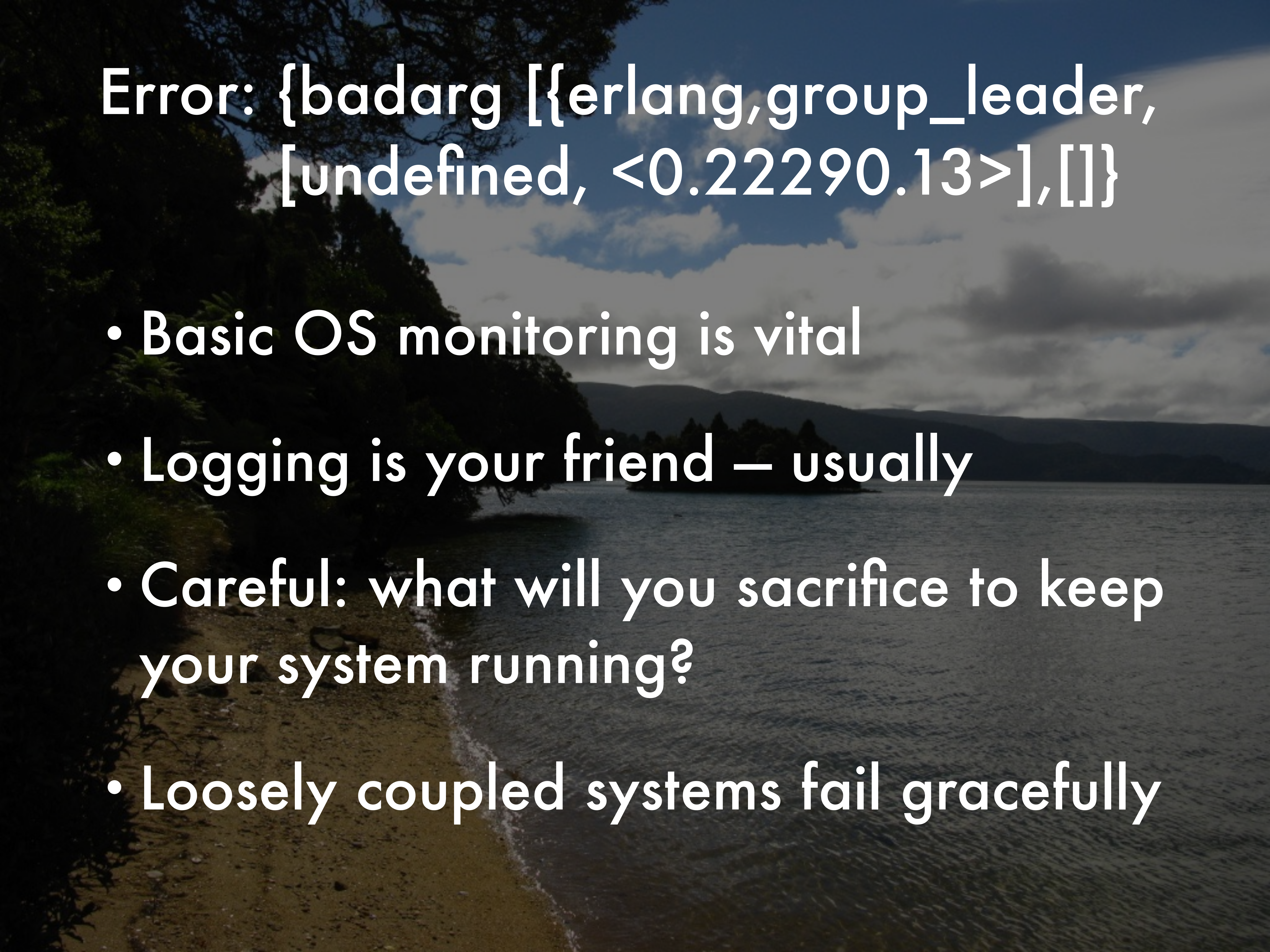
iwantmyname

[Bringing You Artisanal Domain Names since 2007]

Fluffy
Zombie

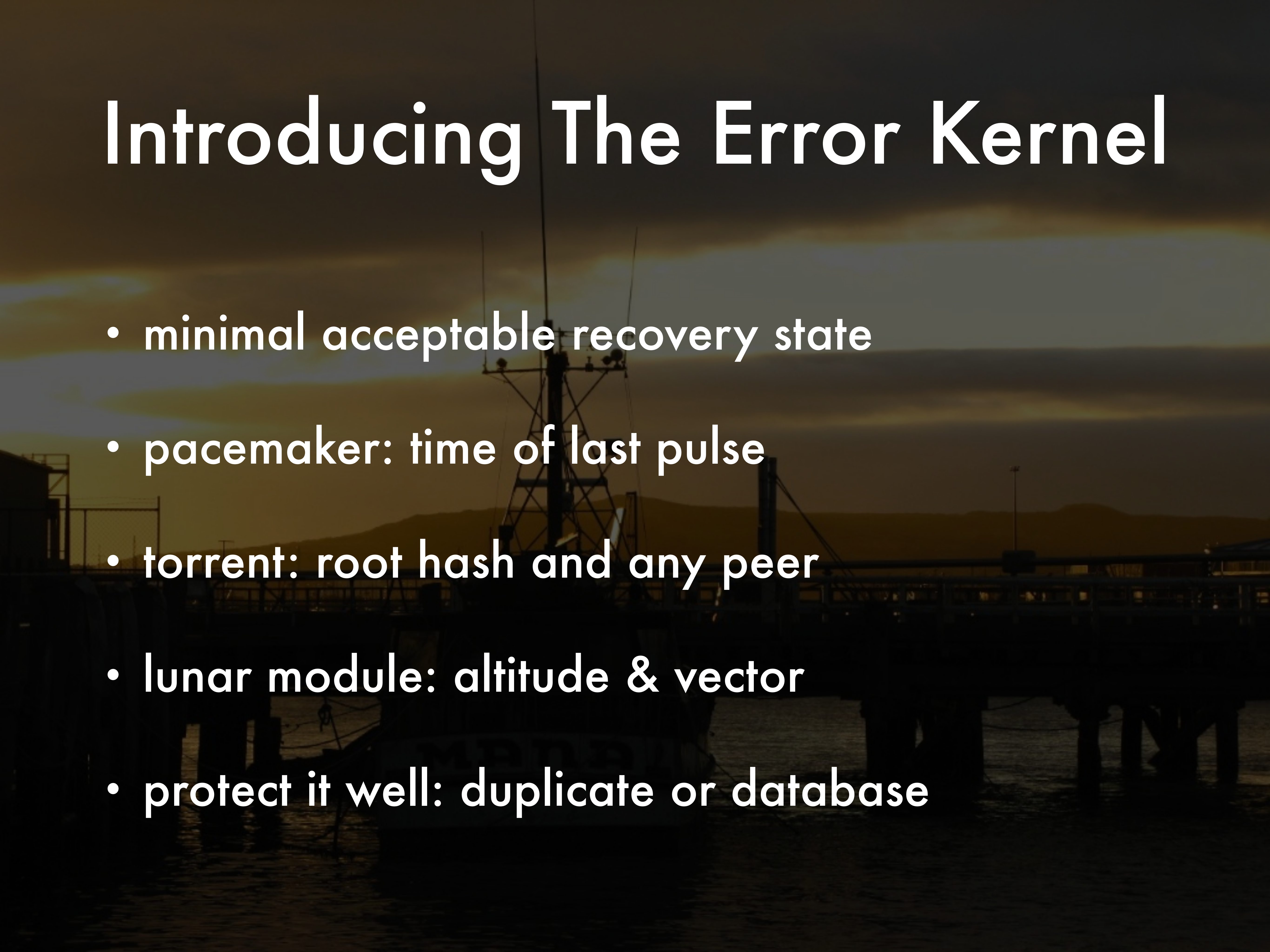


Error: {badarg [{erlang,group_leader, [undefined, <0.22290.13>],[]}

- Basic OS monitoring is vital
- Logging is your friend – usually
- Careful: what will you sacrifice to keep your system running?
- Loosely coupled systems fail gracefully

Designing OTP systems

- Error Kernels
- State Machines
- Queues & Predictable Modes of Failure
- The Happy Path – Dealing with Reality

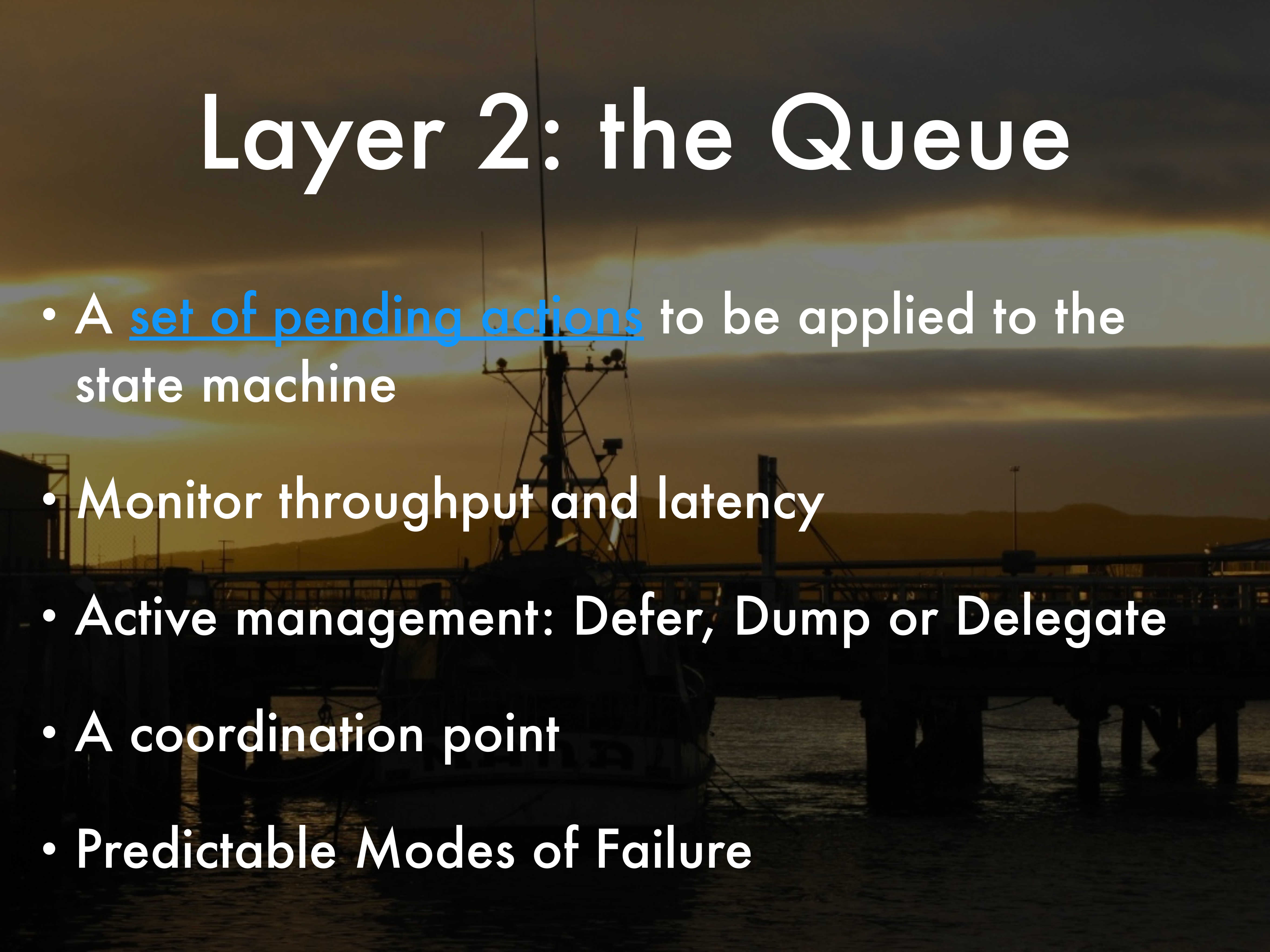


Introducing The Error Kernel

- minimal acceptable recovery state
- pacemaker: time of last pulse
- torrent: root hash and any peer
- lunar module: altitude & vector
- protect it well: duplicate or database

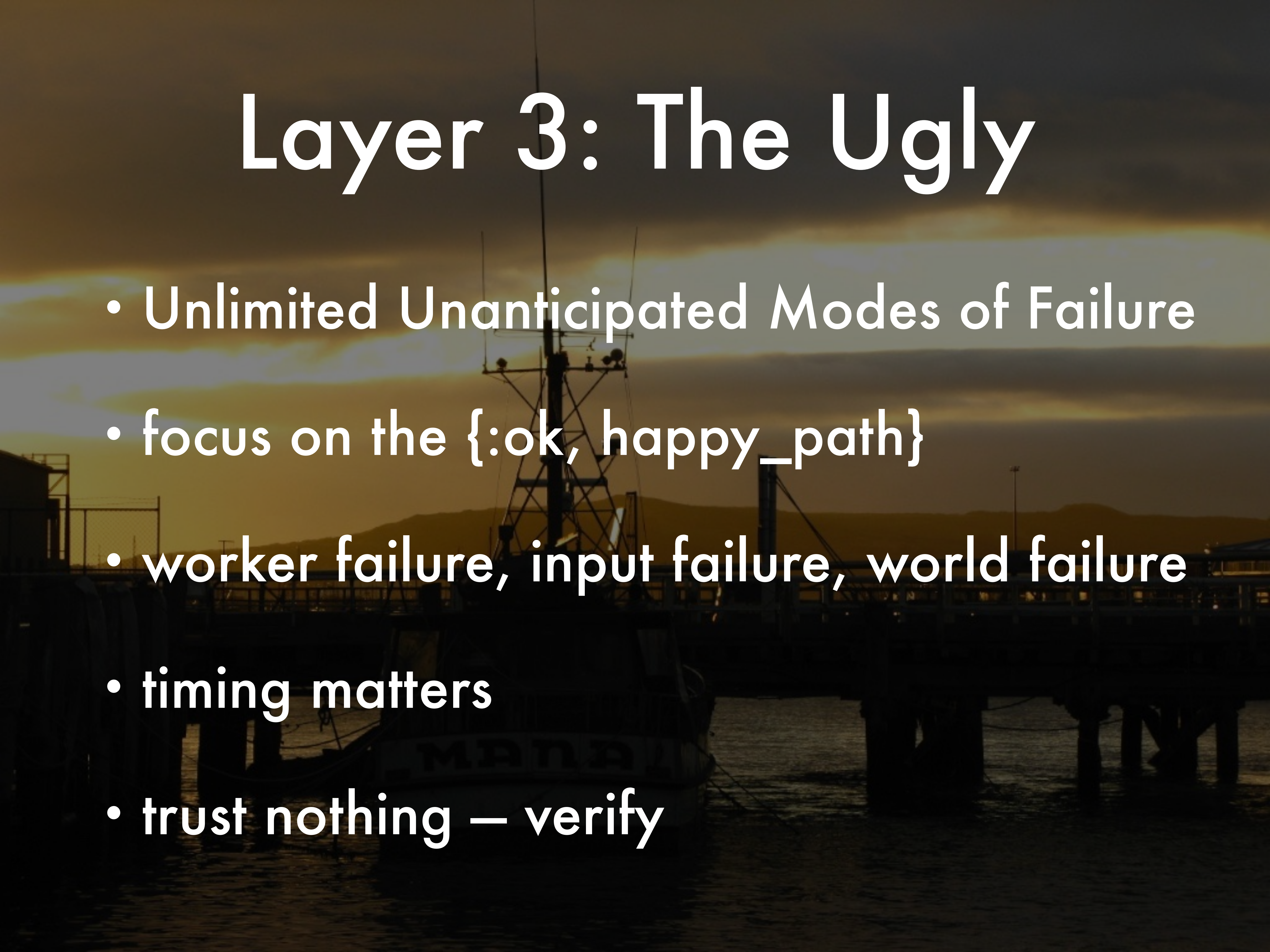
Layer 1: The Enemy is the State

- defined transitions
- validated states
- test it hard
- delegate everything else



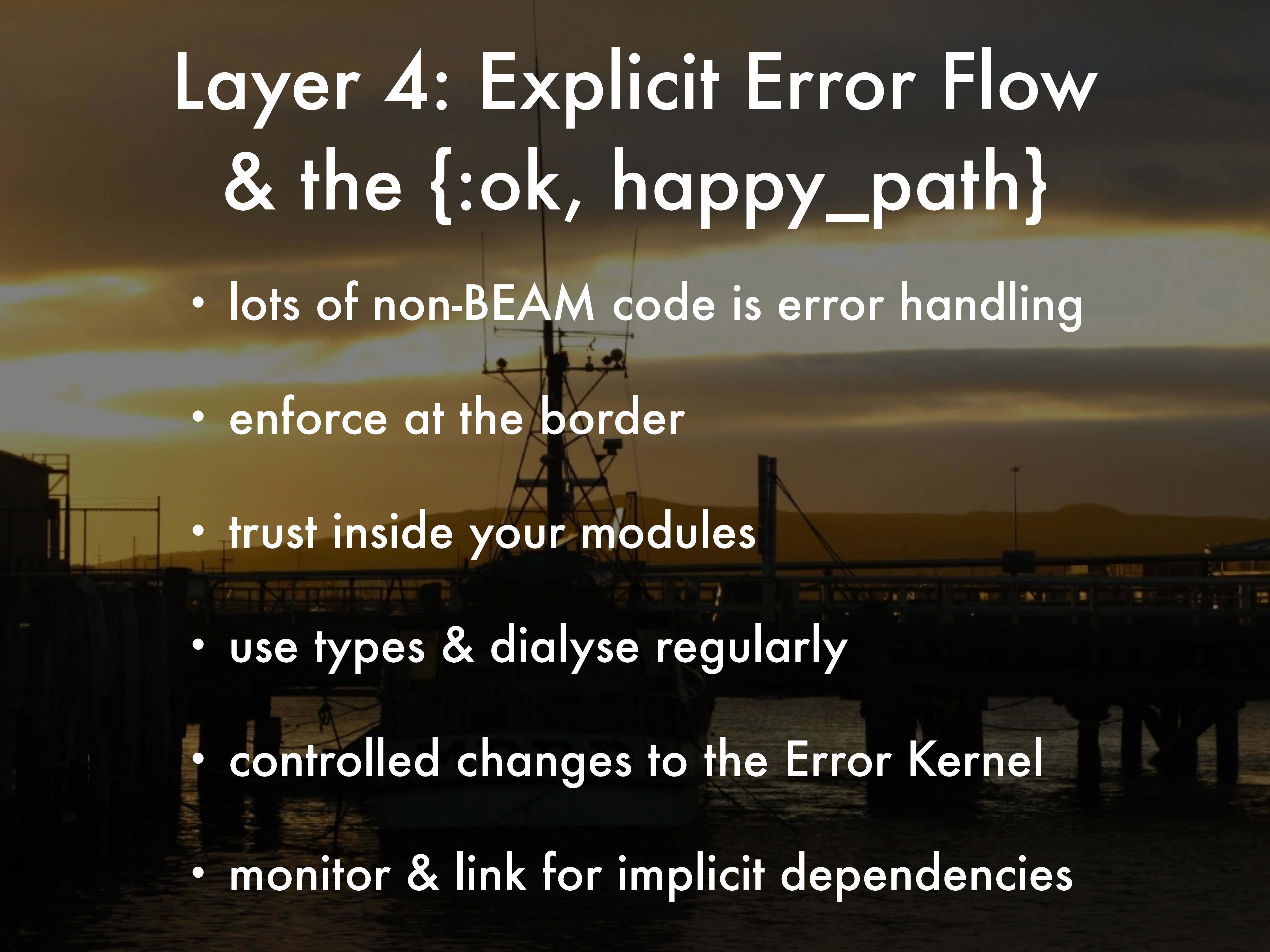
Layer 2: the Queue

- A set of pending actions to be applied to the state machine
- Monitor throughput and latency
- Active management: Defer, Dump or Delegate
- A coordination point
- Predictable Modes of Failure



Layer 3: The Ugly

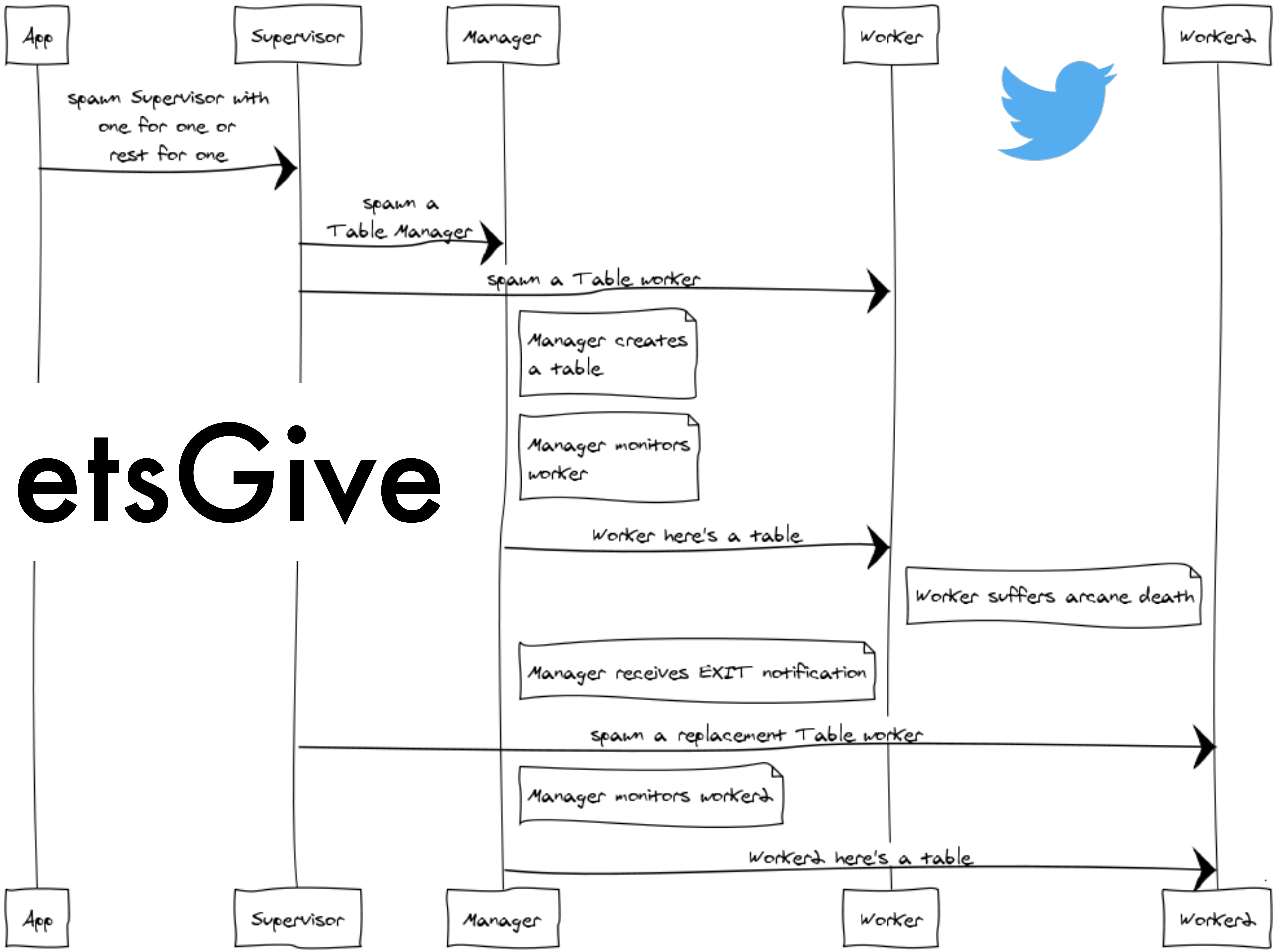
- Unlimited Unanticipated Modes of Failure
- focus on the {:ok, happy_path}
- worker failure, input failure, world failure
- timing matters
- trust nothing – verify



Layer 4: Explicit Error Flow & the {:ok, happy_path}

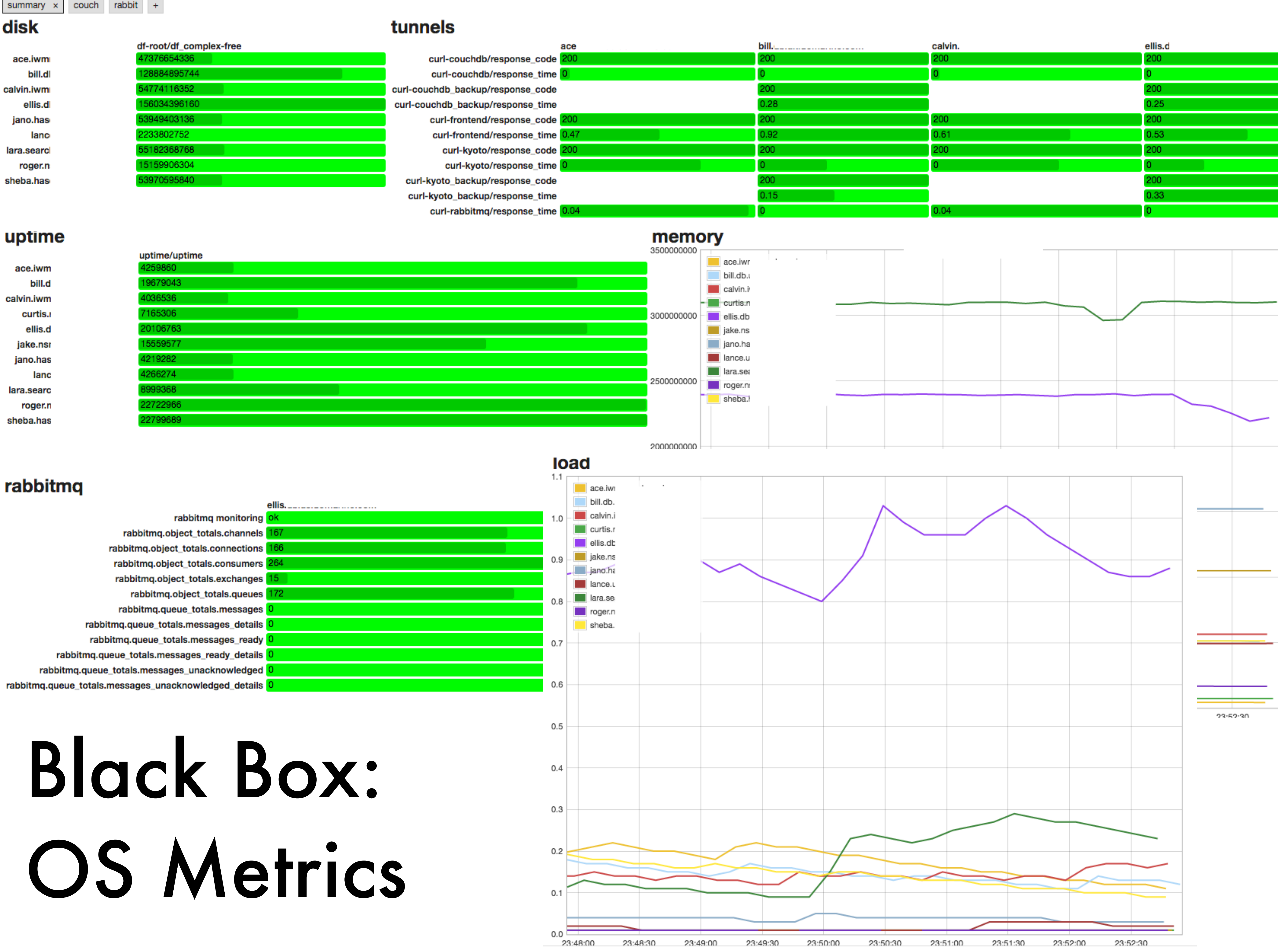
- lots of non-BEAM code is error handling
- enforce at the border
- trust inside your modules
- use types & dialyse regularly
- controlled changes to the Error Kernel
- monitor & link for implicit dependencies

etsGive


App
Supervisor
Manager
Worker
Worker2
spawn Supervisor with one for one or rest for one
spawn a Table Manager
spawn a Table worker
Manager creates a table
Manager monitors worker
Worker here's a table
Worker suffers arcane death
Manager receives EXIT notification
spawn a replacement Table worker
Manager monitors worker2
Worker2 here's a table
App
Supervisor
Manager
Worker
Worker2

BEAM Ops

- <3 Releases: pure BEAM== dependency free
- Hate Releases: anti-UNIX != logging, SIG*
- Logging: jury still out
- Monitoring: do both
    - White box: event your code
    - Black box: look from the OS & network in
- Live debugging



disk

	df-root/df_complex-free
ace.iwm	47376654336
bill.dl	128884895744
calvin.iwm	54774116352
ellis.dl	156034396160
jano.has	53949403136
lanc	2233802752
lara.searcl	55182368768
roger.n	15159906304
sheba.has	53970595840

tunnels

	ace	bill.[redacted]	calvin.	ellis.d
curl-couchdb/response_code	200	200	200	200
curl-couchdb/response_time	0	0	0	0
curl-couchdb_backup/response_code		200		200
curl-couchdb_backup/response_time		0.28		0.25
curl-frontend/response_code	200	200	200	200
curl-frontend/response_time	0.47	0.92	0.61	0.53
curl-kyoto/response_code	200	200	200	200
curl-kyoto/response_time	0	0	0	0
curl-kyoto_backup/response_code		200		200
curl-kyoto_backup/response_time		0.15		0.33
curl-rabbitmq/response_time	0.04	0	0.04	0

uptime

	uptime/uptime
ace.iwm	4259860
bill.d	19679043
calvin.iwm	4036536
curtis.i	7165306
ellis.d	20106763
jake.nsi	15559577
jano.has	4219282
lanc	4266274
lara.searc	8999368
roger.n	22722966
sheba.has	22799689

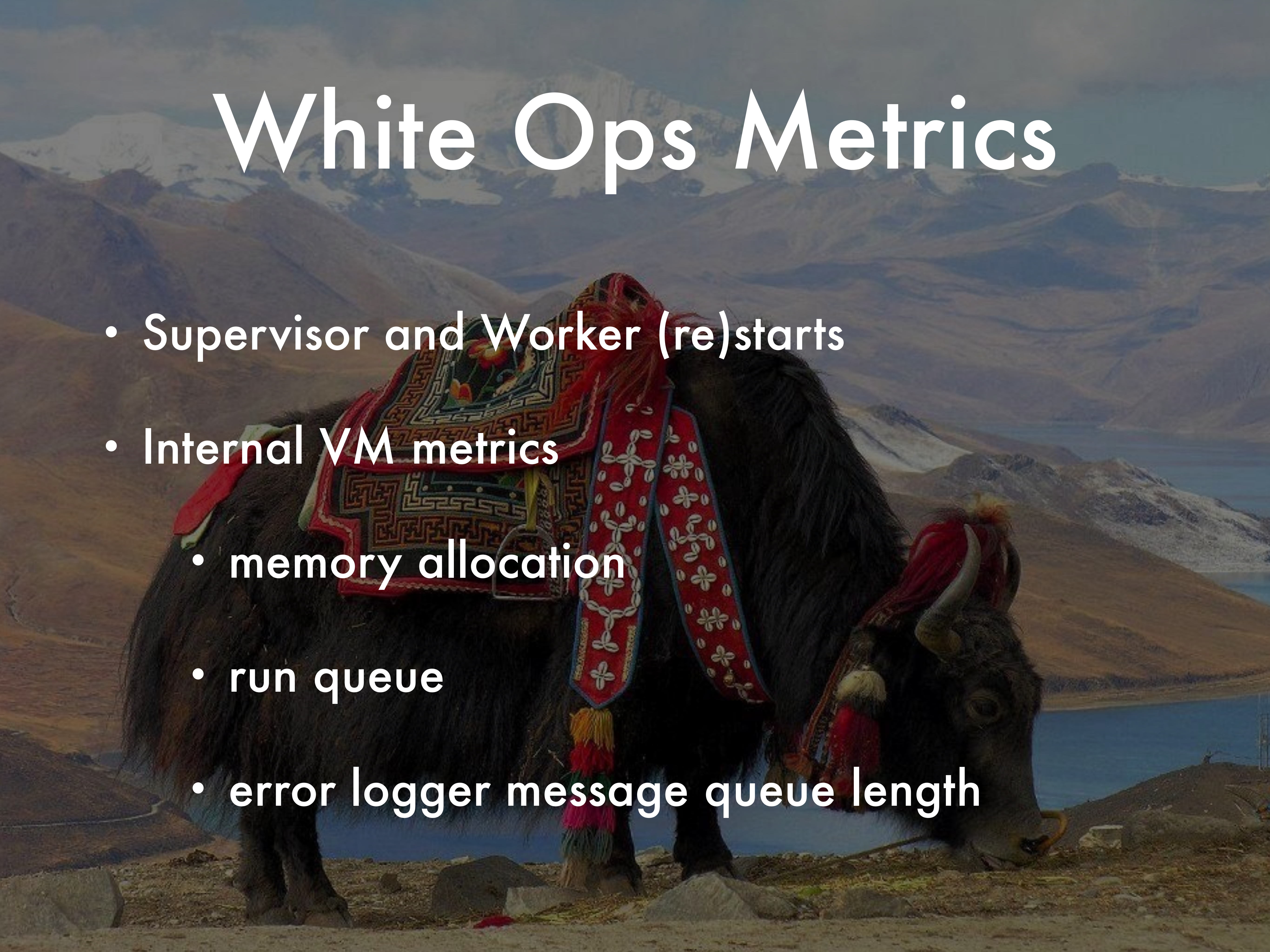
memory

load

rabbitmq

	ellis.[redacted]
rabbitmq monitoring	ok
rabbitmq.object_totals.channels	167
rabbitmq.object_totals.connections	166
rabbitmq.object_totals.consumers	264
rabbitmq.object_totals.exchanges	15
rabbitmq.object_totals.queues	172
rabbitmq.queue_totals.messages	0
rabbitmq.queue_totals.messages_details	0
rabbitmq.queue_totals.messages_ready	0
rabbitmq.queue_totals.messages_ready_details	0
rabbitmq.queue_totals.messages_unacknowledged	0
rabbitmq.queue_totals.messages_unacknowledged_details	0

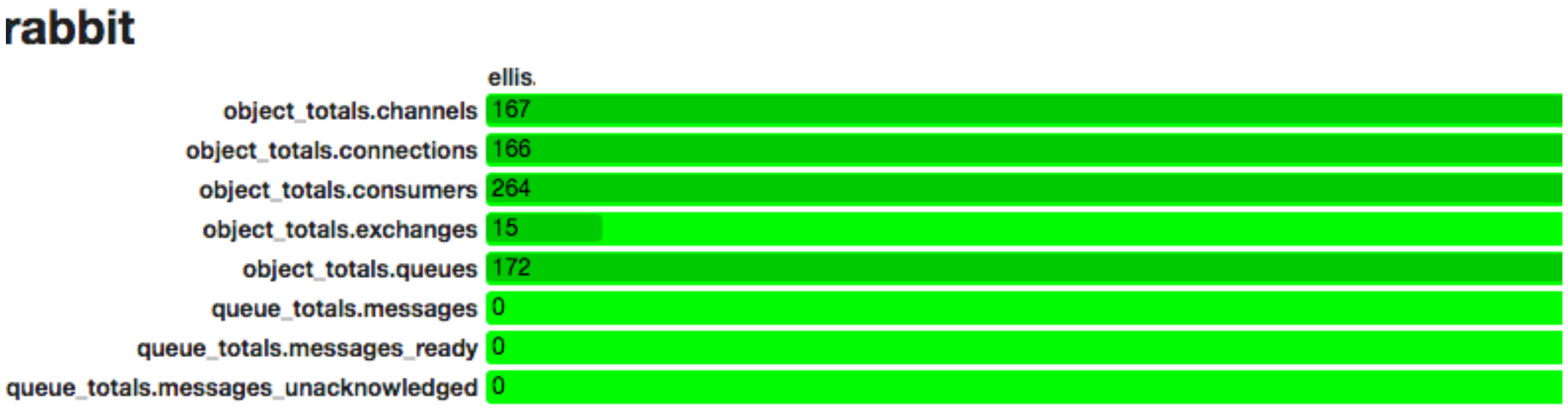
Black Box: OS Metrics



White Ops Metrics

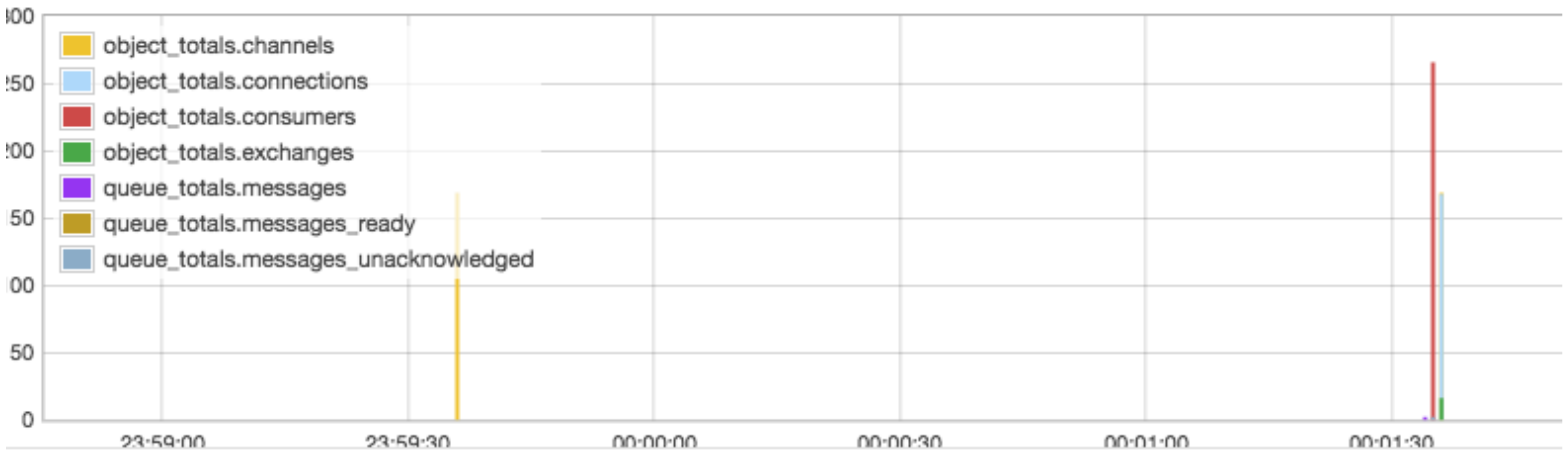
- Supervisor and Worker (re)starts
- Internal VM metrics
  - memory allocation
  - run queue
  - error logger message queue length

App-Specific – RabbitMQ

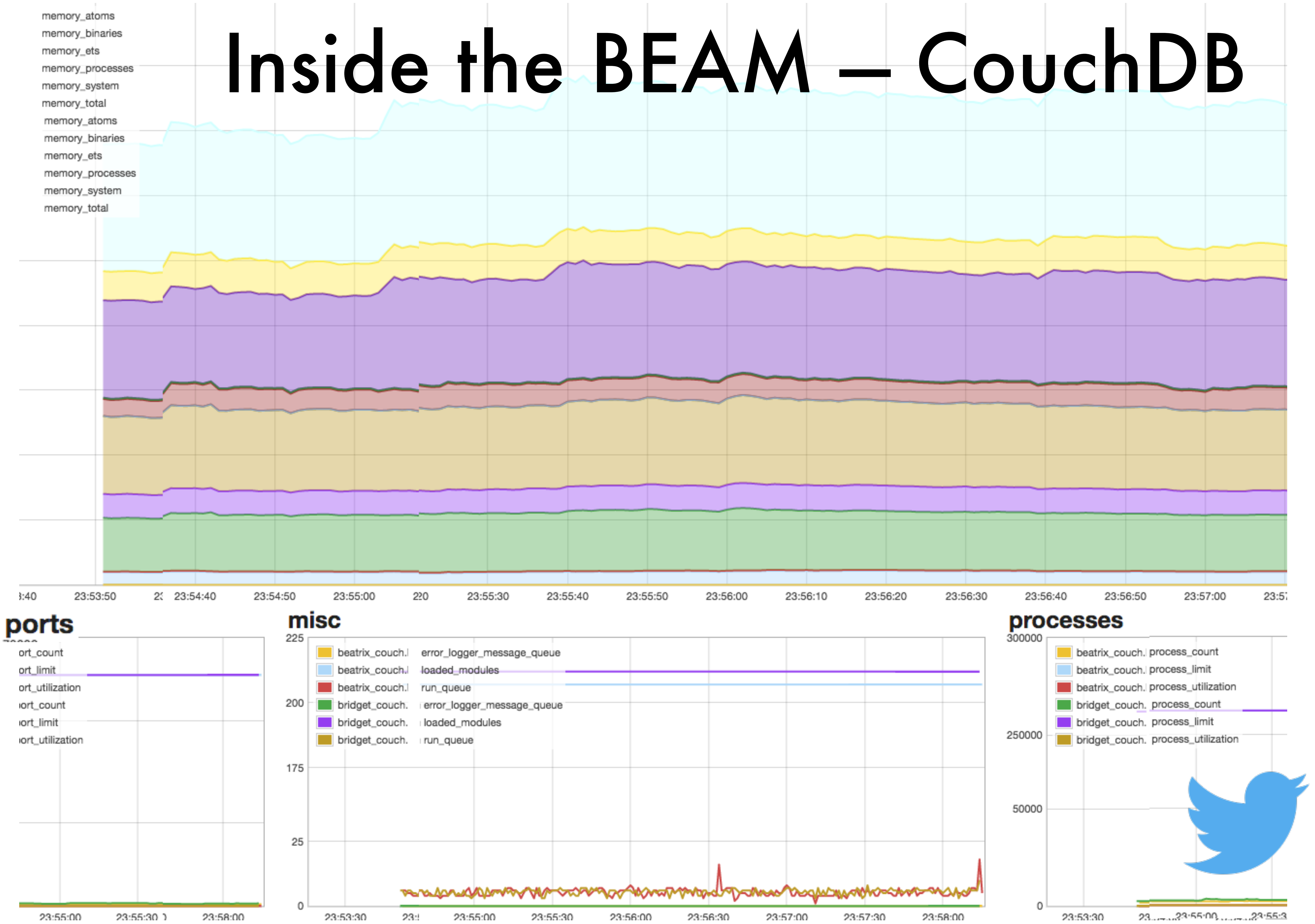


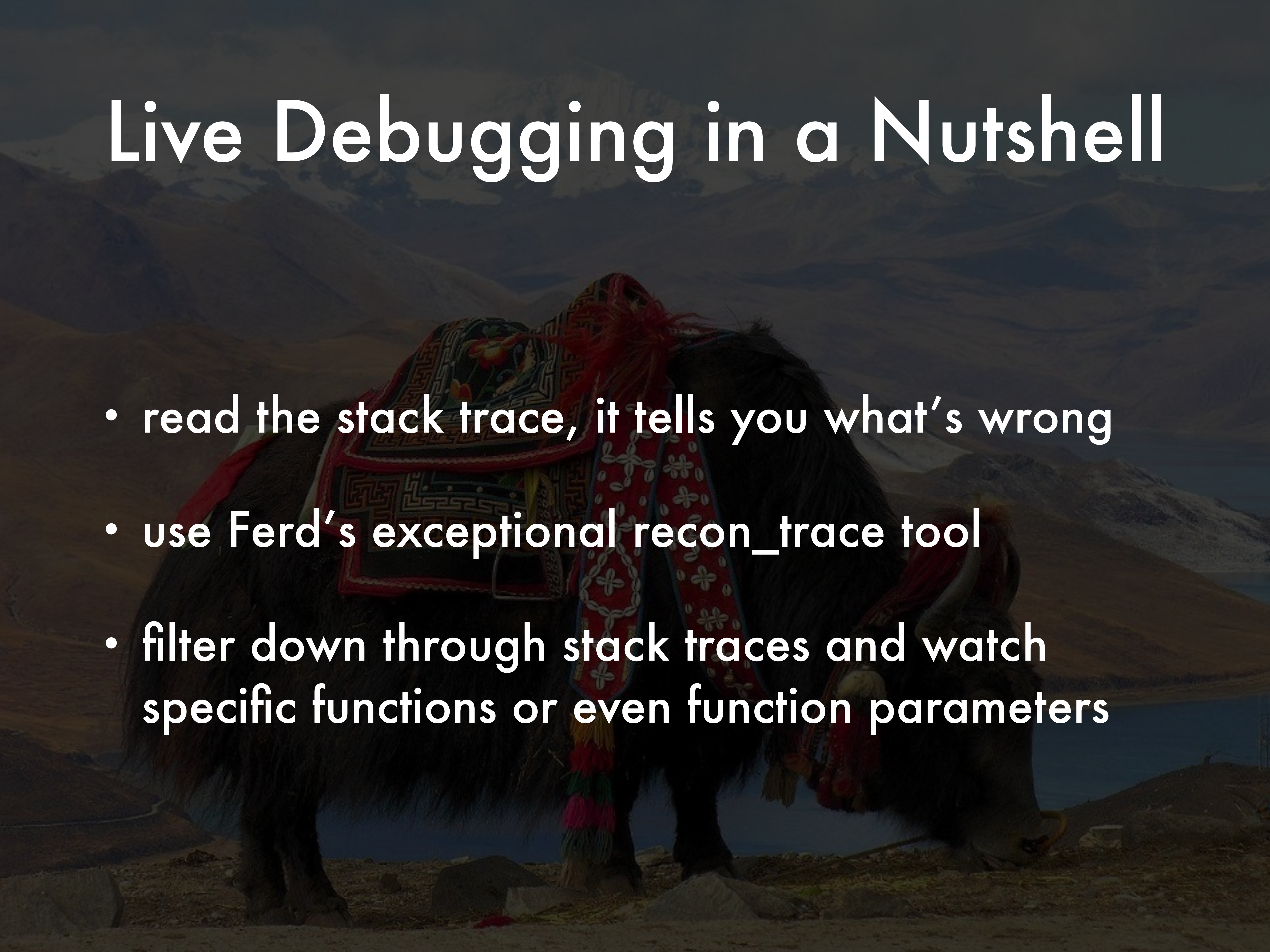
rabbit

	ellis.
object_totals.channels	167
object_totals.connections	166
object_totals.consumers	264
object_totals.exchanges	15
object_totals.queues	172
queue_totals.messages	0
queue_totals.messages_ready	0
queue_totals.messages_unacknowledged	0



Inside the BEAM – CouchDB





Live Debugging in a Nutshell

- read the stack trace, it tells you what's wrong
- use Ferd's exceptional recon_trace tool
- filter down through stack traces and watch specific functions or even function parameters

@dch__

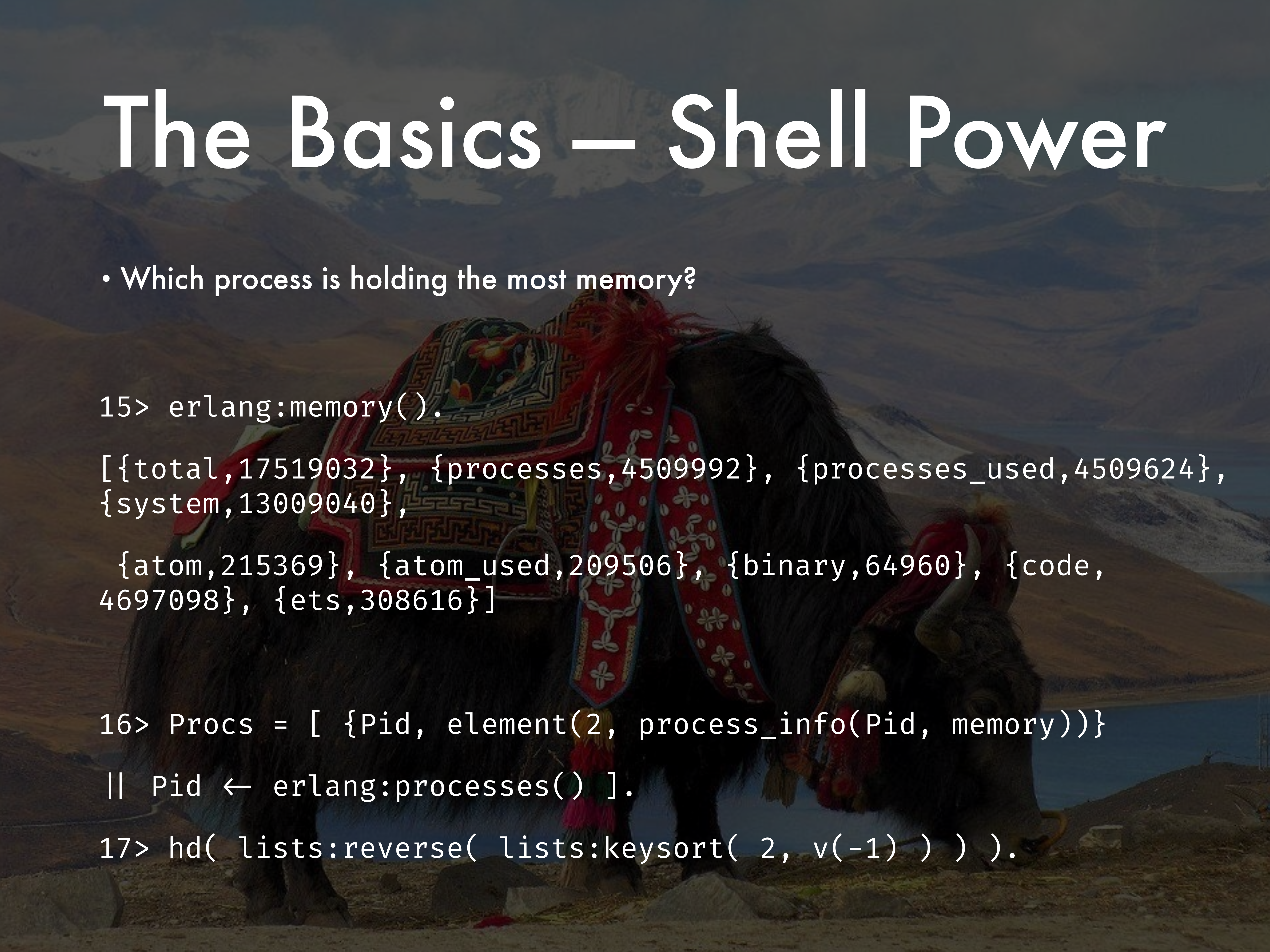
https://home.apache.org/~dch

iwantmyname

Image Credits

- Zombie Rabbit: http://checker-bee.deviantart.com/art/Fluffy-Zombie-Bunny-325012765
- Yak: http://ideastochill.blogspot.com/2014/01/yak-animal.html
- Photos: @dch__ taken onsite in New Zealand

iwantmyname



The Basics – Shell Power

- Which process is holding the most memory?

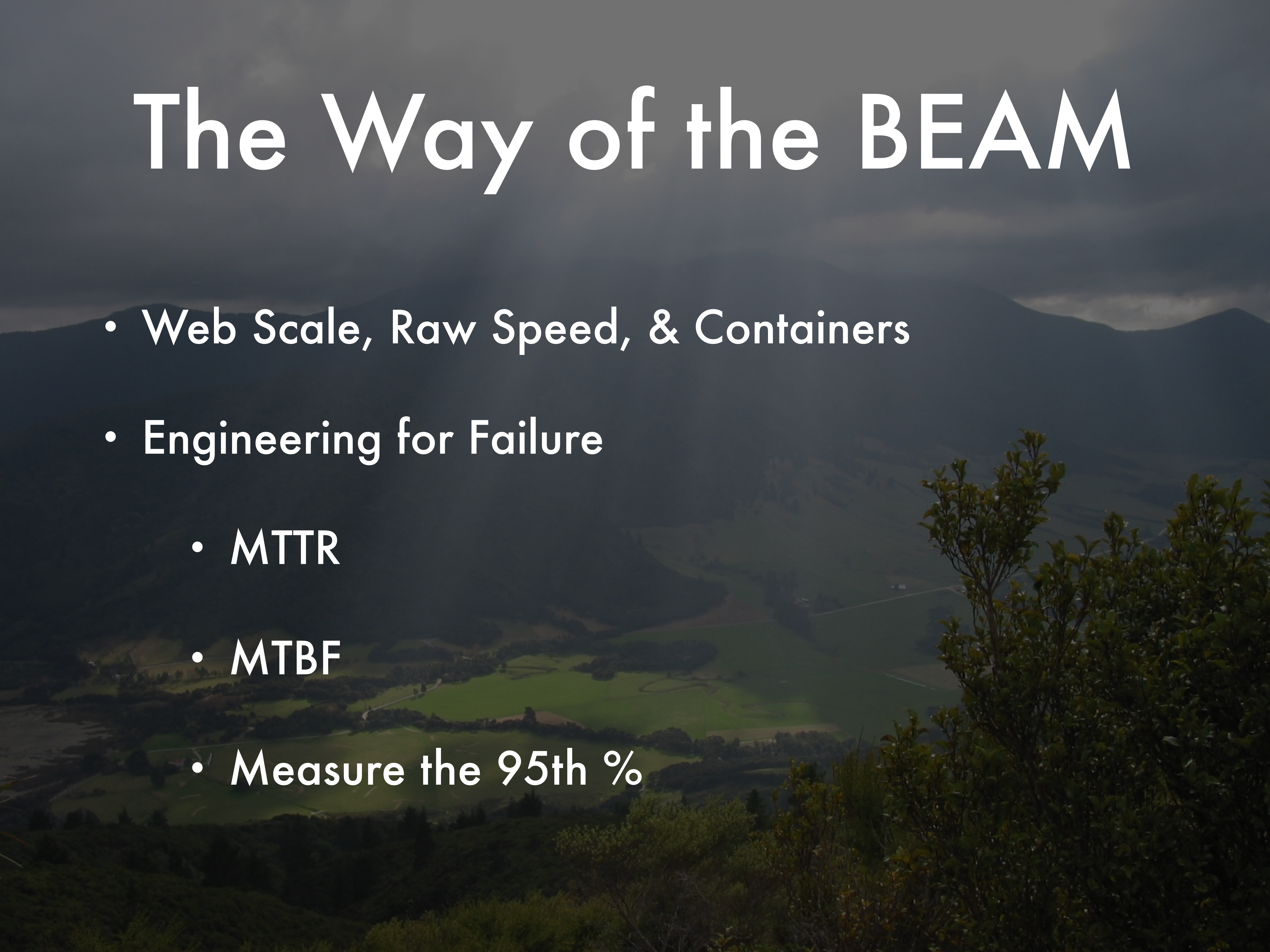
```
15> erlang:memory().

[{total,17519032}, {processes,4509992}, {processes_used,4509624},
{system,13009040},

 {atom,215369}, {atom_used,209506}, {binary,64960}, {code,
4697098}, {ets,308616}]



16> Procs = [ {Pid, element(2, process_info(Pid, memory))}

|| Pid ← erlang:processes() ].

17> hd( lists:reverse( lists:keysort( 2, v(-1) ) ) ).
```

What is that MFA doing?

```
4> Procs = [ {Pid, element(2,
process_info(Pid, memory))}

|| Pid ← erlang:processes() ].

5>  hd( lists:reverse( lists:keysort( 2,
v(-1) ) ) ).

6>  exit(list_to_pid("<0.36.0>"),
your_time_has_come ). %% any signal will do

true unless its trapping exits. Then use
kill.
```

Gone.. Permanently.

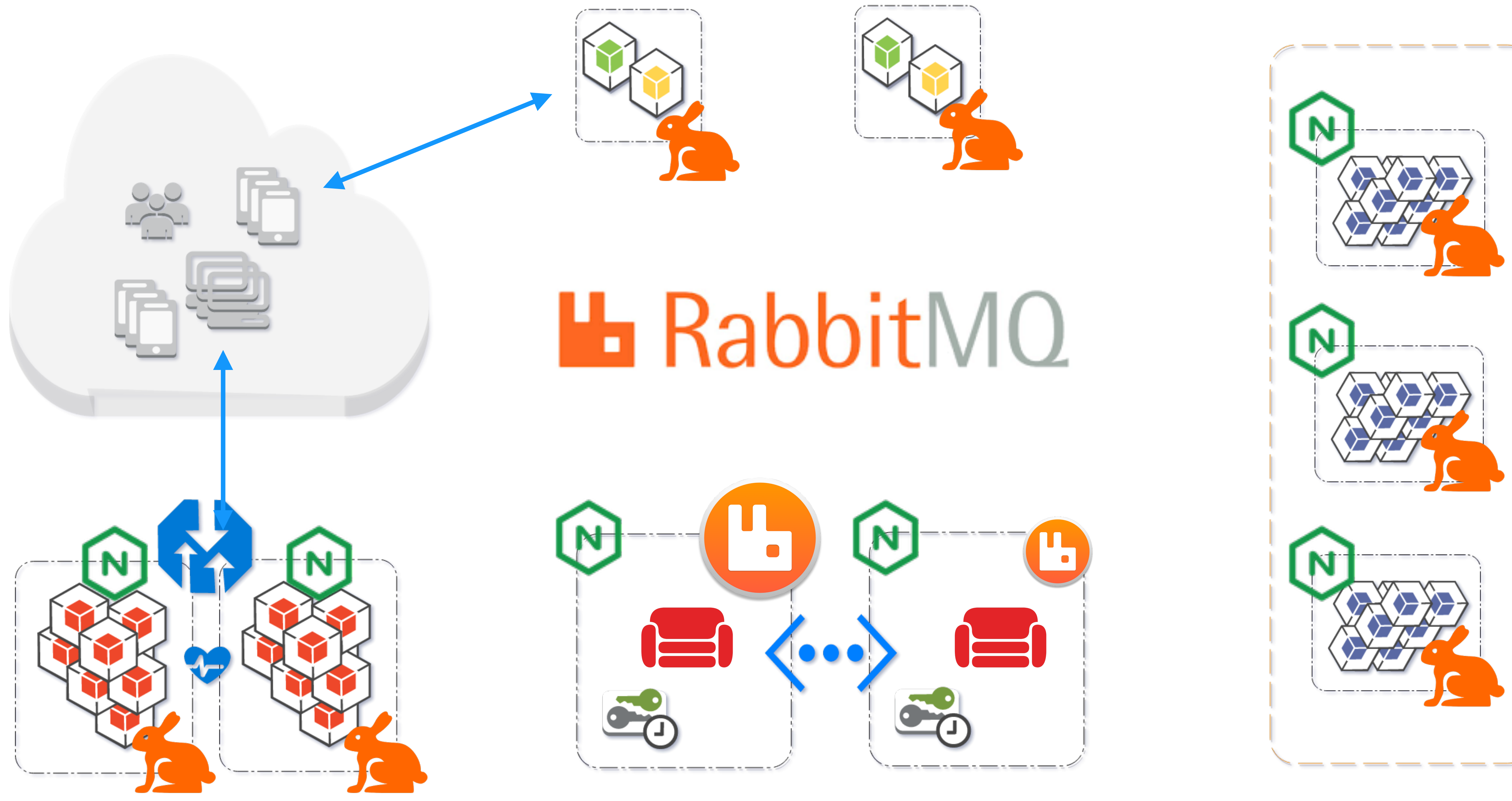
```
12> exit(list_to_pid("<0.36.0>"), kill).

Observer: Child (unknown) crashed exiting:  <0.76.0> shutdown

true

13> 2016-11-24 08:55:03 std_info

    application: kernel

    exited: shutdown

    type: permanent

{"Kernel pid
terminated",application_controller,"{application_terminated,kernel,s
hutdown}"}

Kernel pid terminated (application_controller)
({application_terminated,kernel,shutdown})
```



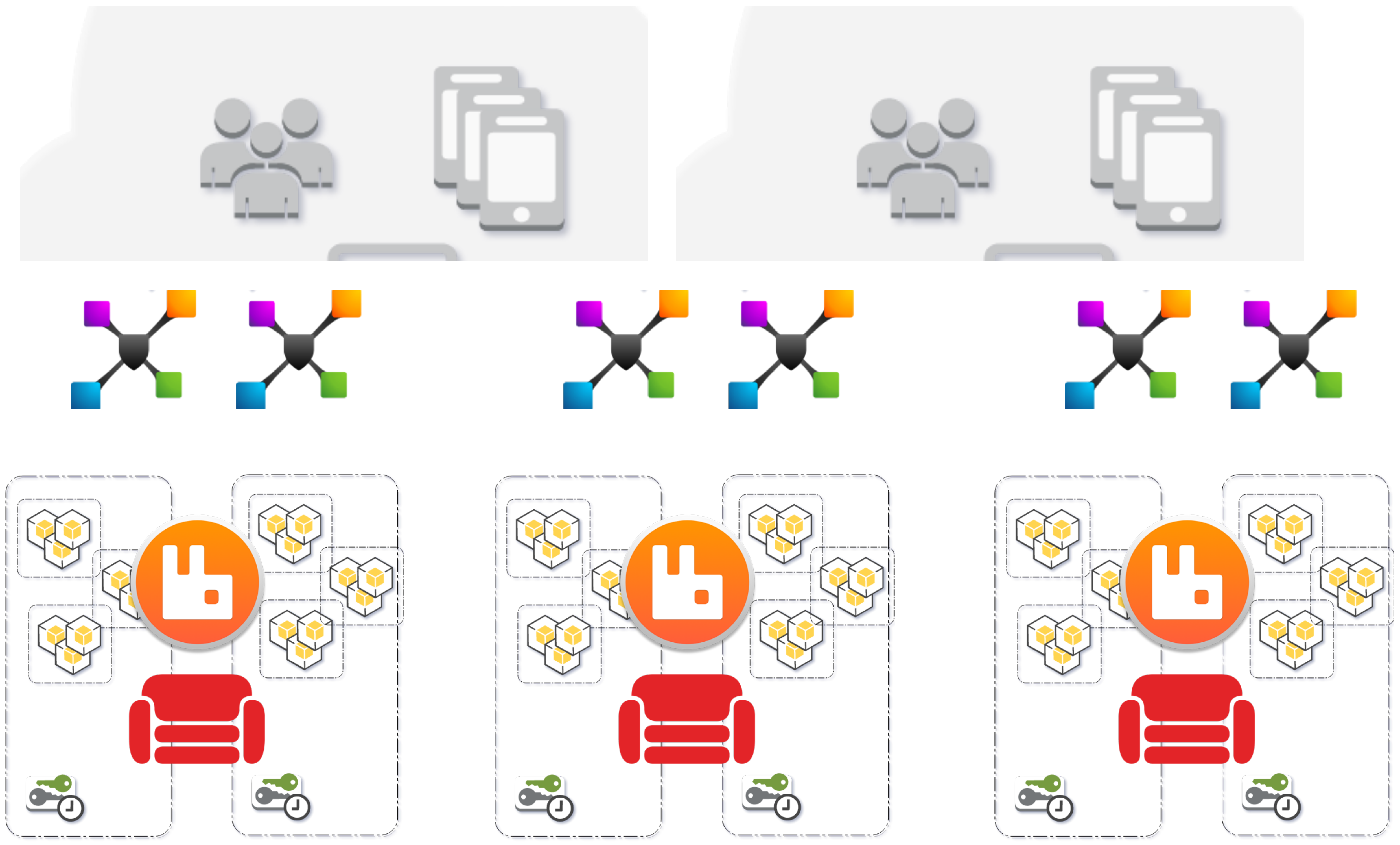
The Way of the BEAM

- Web Scale, Raw Speed, & Containers
- Engineering for Failure
  - MTTR
  - MTBF
  - Measure the 95th %

Gen#1



Gen#2



@dch__

https://home.apache.org/~dch

iwantmyname

Security – epmd & net_kernel

- epmd:
  - limit its listening ports
  - block epmd from outgoing ports
  - run epmd as privilege dropped daemon & not as root
  - require that daemon as a dependency of your apps (systemd/rc.d/…) check your distort

```
> epmd -address 12.13.14.15,10.0.0.11
```

Security — epmd & net_kernel

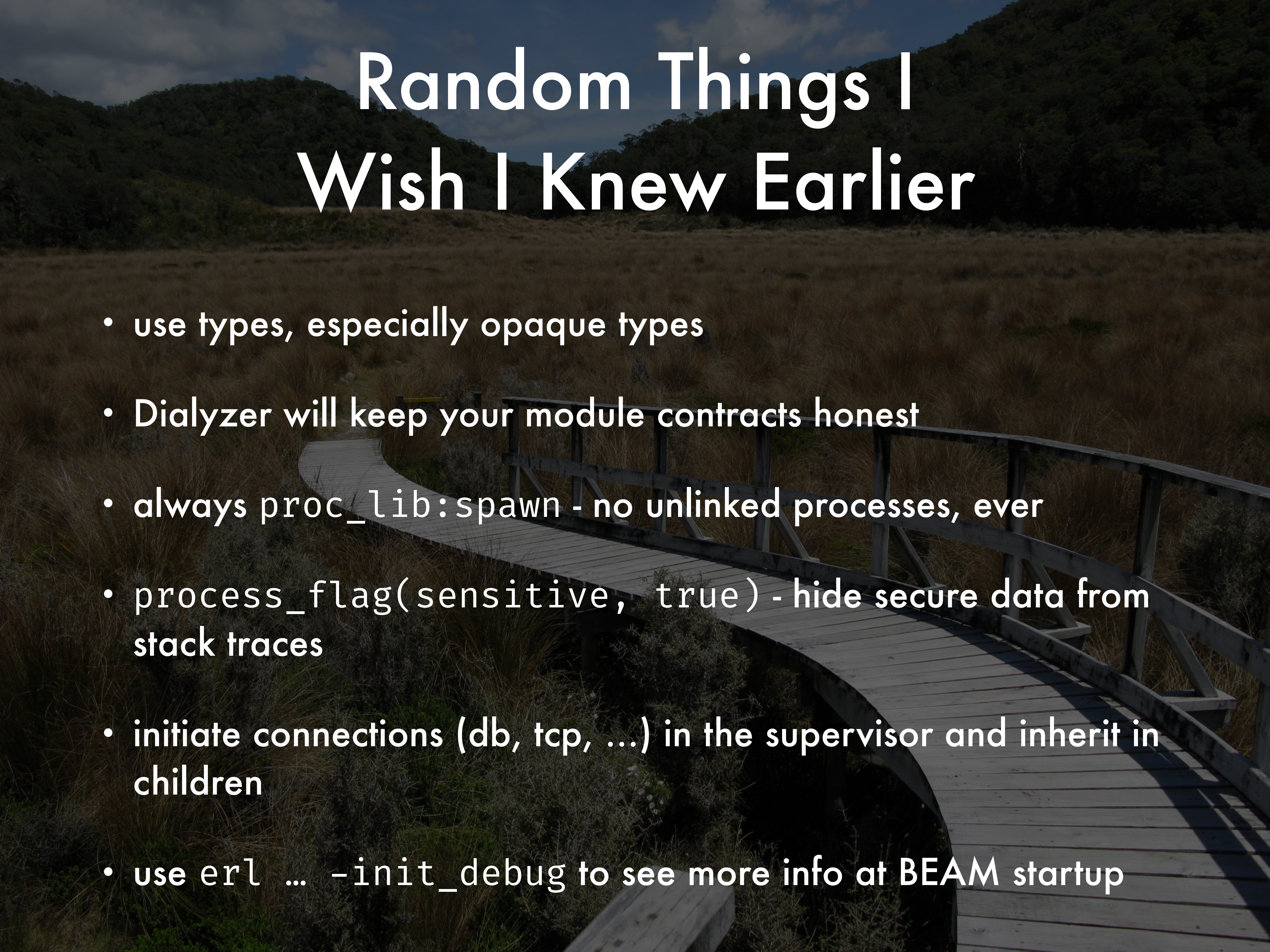
- your app:
  - add `-start_epmd false` to your vm.args
  - use secure tunnels (spiped/ssh/ipsec/…) for your distributed mesh
  - restrict net_kernel range for firewalling in sys.config

```
> {kernel, [{inet_dist_listen_min, 4370},
{inet_dist_listen_max, 4380}]}
```

- read http://erlang.org/doc/man/net_kernel.html & http://erlang.org/doc/man/erl.html

Essential Tools

- external:
  - recon for tracing (erlang)
  - tap is an elixir wrapper around recon
  - redbug for quick queries (erlang)
  - distillery for building and running releases (elixir)
  - katja & katja_vmstats for runtime metrics (erlang)
- internal:
  - observer FTW
  - dbg & trace modules
  - remsh / rebar3 shell / etc (ssh + remsh to localhost)



Random Things I Wish I Knew Earlier

- use types, especially opaque types
- Dialyzer will keep your module contracts honest
- always `proc_lib:spawn` - no unlinked processes, ever
- `process_flag(sensitive, true)` - hide secure data from stack traces
- initiate connections (db, tcp, ...) in the supervisor and inherit in children
- use `erl … -init_debug` to see more info at BEAM startup

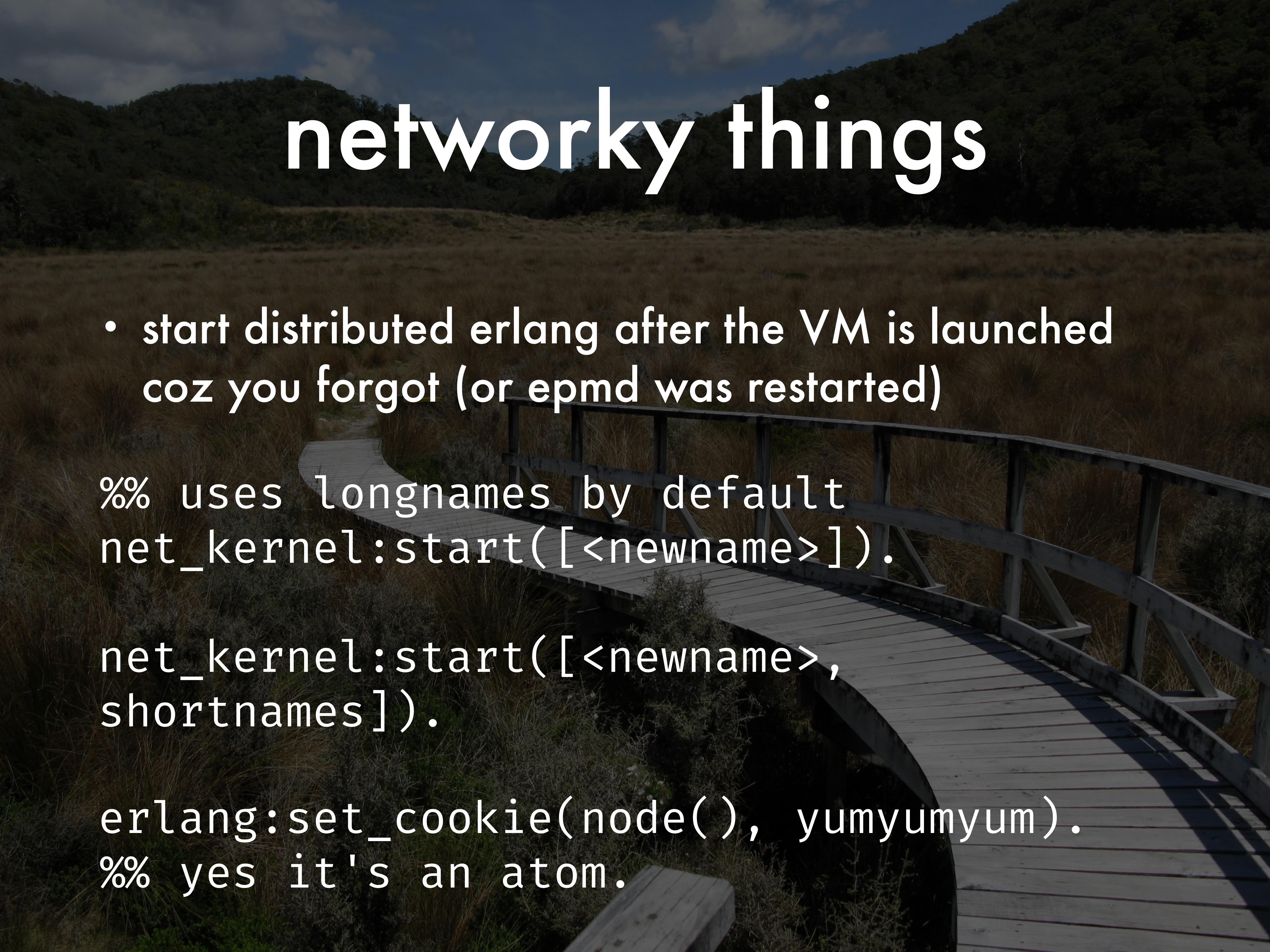
ETS Things

- using `:compressed` tables when the *value* doesn't need to be inspected will save memory
- `:read_concurrency` & `:write_concurrency` are off by default
- use `:heir` & `:ets.giveaway/2` allow `:ets.new(:table, [{:heir, trusted_pid, data_to_send_with_termination_message}])` to make your ets tables survive worker death

networky things

- start distributed erlang after the VM is launched coz you forgot (or epmd was restarted)

```
%% uses longnames by default
net_kernel:start([<newname>]).

net_kernel:start([<newname>,
shortnames]).

erlang:set_cookie(node(), yumyumyum).
%% yes it's an atom.
```



@dch__

https://home.apache.org/~dch

iwantmyname

Credits

- Zombie Rabbit: http://checker-bee.deviantart.com/art/Fluffy-Zombie-Bunny-325012765
- Yak: http://ideastochill.blogspot.com/2014/01/yak-animal.html
- Photos: @dch__ taken onsite in New Zealand

iwantmyname