Knock Knock

Understanding Who is Using Your Web Applications

Braintree

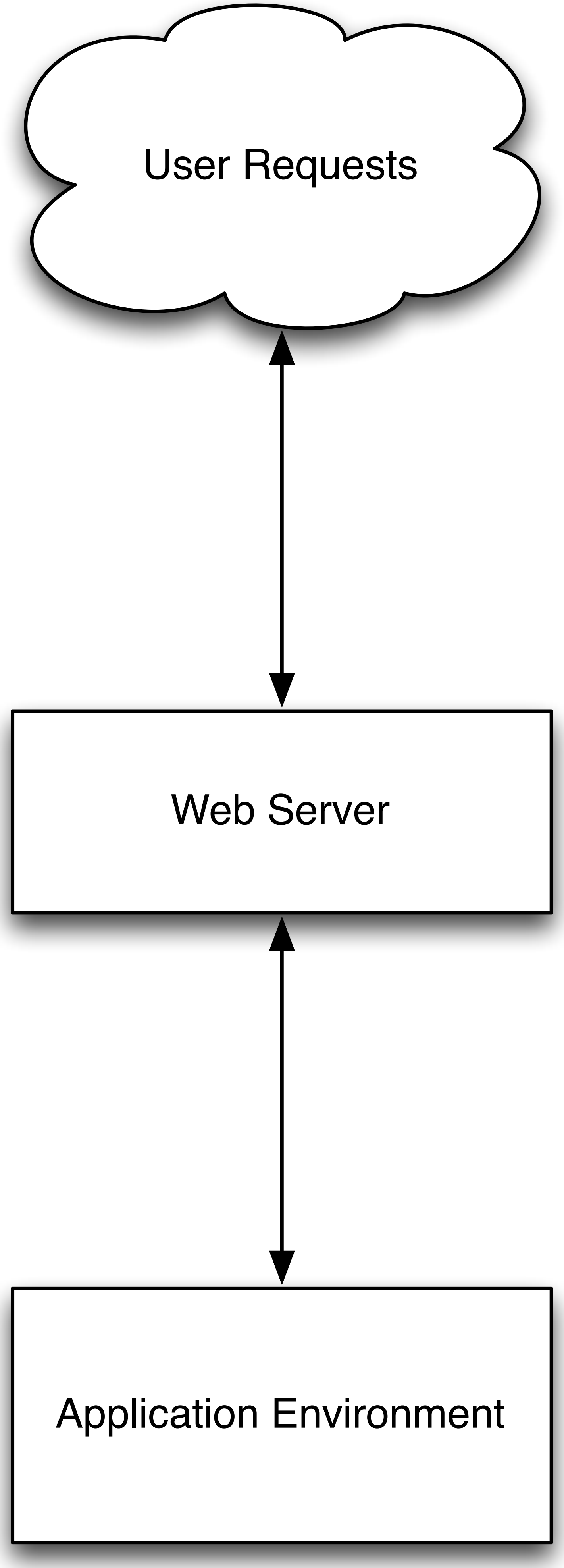
Aaron Bedra
Application Security Lead
Braintree Payments

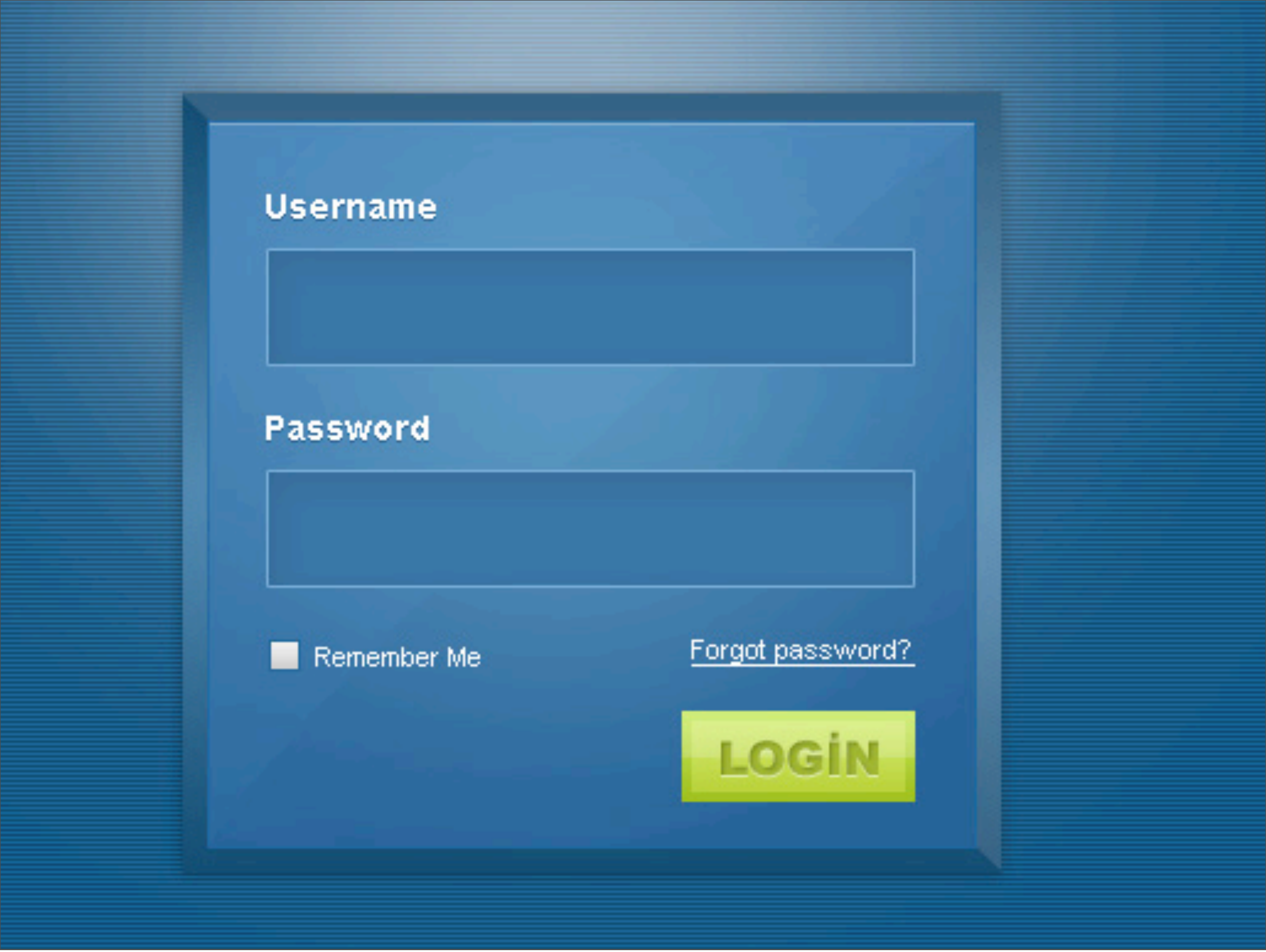
Right now, your web applications are being attacked

And it will happen again, and again, and again

But not always in the way you think

Let’s take a look at typical application security measures


User Requests
Web Server
Application Environment


Username
Password
□ Remember Me
Forgot password?
LOGIN

roland : 12345

roland : 12345

And we go on with our day

How many of you stop there?

It’s time to start asking more questions

But remember…

Don’t impact user experience!

???

- Signature based detection
- Anomaly detection
- Reputational intelligence
- Action
- Repsheet

Signatures

Mod Security

Web Application Firewall

Rule based detection

Allows you to block or alert if traffic matches a signature

Improved by the OWASP Core Rule Set

A great tool to add to your stack

Works with Apache, nginx, and IIS

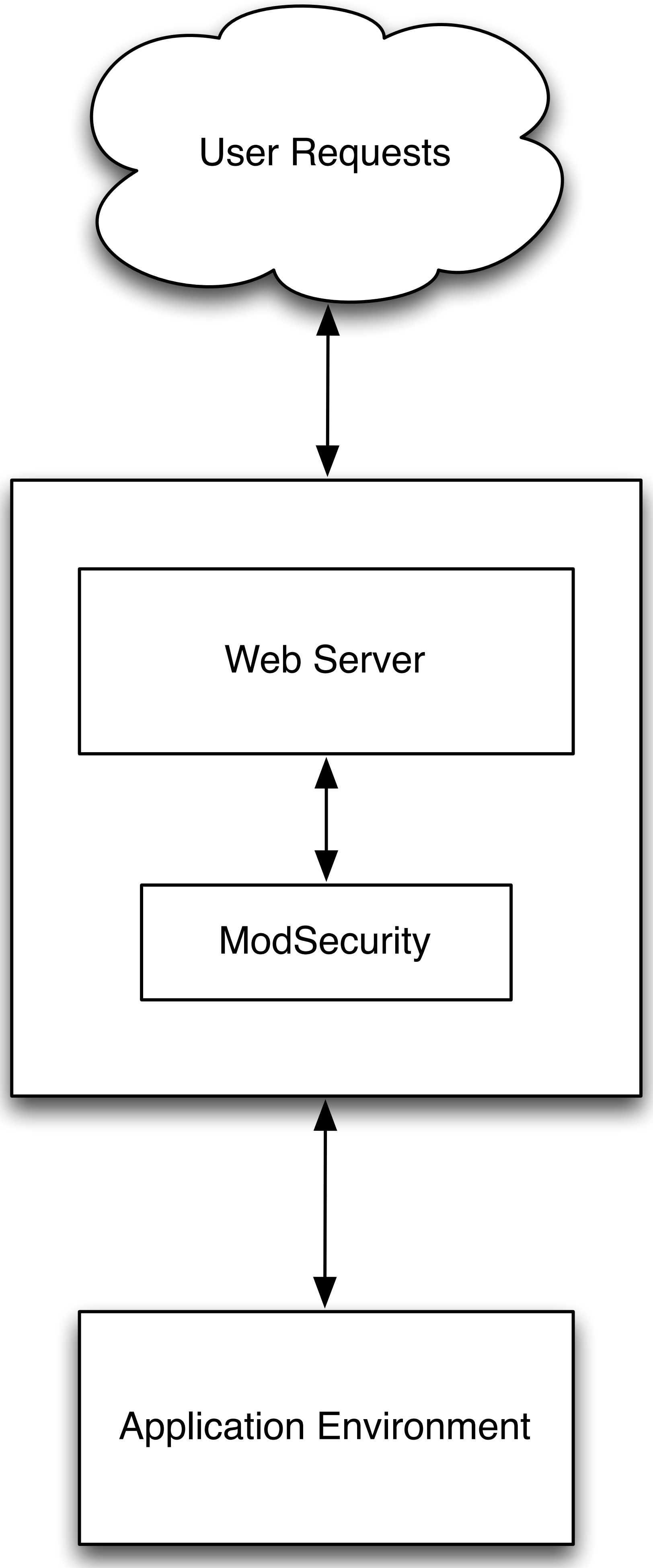
Works *well* with Apache

Like most signature based tools it requires tuning

And has a high possibility of false positives

Great for helping with 0-day attacks

Favor alerting over blocking in most scenarios


User Requests
Web Server
ModSecurity
Application Environment

Anomalies

```
10.20.253.8 - - [23/Apr/2013:14:20:21 +0000]
"POST /login HTTP/1.1" 200 267"-" "Mozilla/
5.0 (Windows NT 6.1; WOW64; rv:8.0) Gecko/
20100101 Firefox/8.0" "77.77.165.233"
```

```
10.20.253.8 - - [23/Apr/2013:14:20:22 +0000]
"POST /users/king-roland/cc_records HTTP/1.1"
302 2085 "-" "Mozilla/5.0 (Windows NT 6.1;
WOW64; rv:8.0) Gecko/20100101 Firefox/8.0"
"77.77.165.233"
```

```
10.20.253.8 - - [23/Apr/2013:14:20:23 +0000]
"POST /users/king-roland/cc_records HTTP/1.1"
302 2083 "-" "Mozilla/5.0 (Windows NT 6.1;
WOW64; rv:8.0) Gecko/20100101 Firefox/8.0"
"77.77.165.233"
```

```
10.20.253.8 - - [23/Apr/2013:14:20:24 +0000]
"POST /users/king-roland/cc_records HTTP/1.1"
302 2085 "-" "Mozilla/5.0 (Windows NT 6.1;
WOW64; rv:8.0) Gecko/20100101 Firefox/8.0"
"77.77.165.233"
```

What do you see?

I see a website getting carded

???

Play by play

Login Request

```
10.20.253.8 - - [23/Apr/2013:14:20:21 +0000]
"POST /login HTTP/1.1" 200 267"-" "Mozilla/
5.0 (Windows NT 6.1; WOW64; rv:8.0) Gecko/
20100101 Firefox/8.0" "77.77.165.233"
```

I sec delay

Add credit card to account #1

```
10.20.253.8 - - [23/Apr/2013:14:20:22 +0000]
"POST /users/king-roland/cc_records HTTP/1.1"
302 2085 "-" "Mozilla/5.0 (Windows NT 6.1;
WOW64; rv:8.0) Gecko/20100101 Firefox/8.0"
"77.77.165.233"
```

Add credit card to account #2

1 sec delay

```
10.20.253.8 - - [23/Apr/2013:14:20:23 +0000]
"POST /users/king-roland/cc_records HTTP/1.1"
302 2083 "-" "Mozilla/5.0 (Windows NT 6.1;
WOW64; rv:8.0) Gecko/20100101 Firefox/8.0"
"77.77.165.233"
```

FF 8 on Windows 7
or Bot?

Add credit card to account #3

1 sec delay

```
10.20.253.8 - - [23/Apr/2013:14:20:24 +0000]
"POST /users/king-roland/cc_records HTTP/1.1"
302 2085 "-" "Mozilla/5.0 (Windows NT 6.1;
WOW64; rv:8.0) Gecko/20100101 Firefox/8.0"
"77.77.165.233"
```

FF 8 on Windows 7
or Bot?

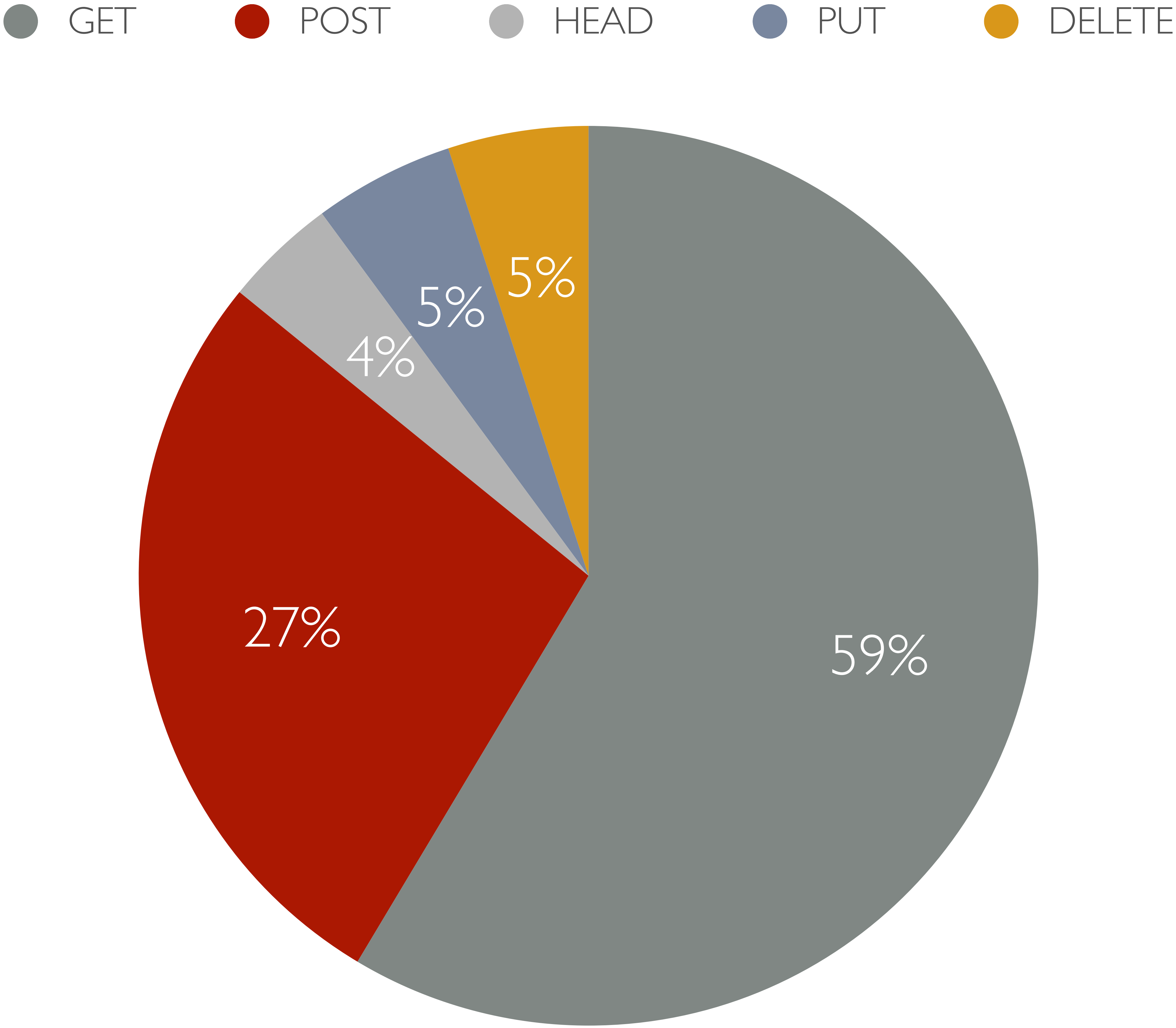
Plovdiv Bulgaria

And this continues…

10,000 more times

Those were the only requests that IP address made

Aside from the number of requests what else gave it away?


● GET
● POST
● HEAD
● PUT
● DELETE
5%
5%
4%
27%
59%

HTTP method distribution is important

When an actor deviates significantly, there must be a reason!

Let’s talk GeoIP

Adding GeoIP information is generically useful

But it also helps in the face of an attack

It can help protect you and your users

Scenario

King Roland gets his GMail account hacked

Hacker sends a password reset request to your server

Normally, you would email the reset

Unless...

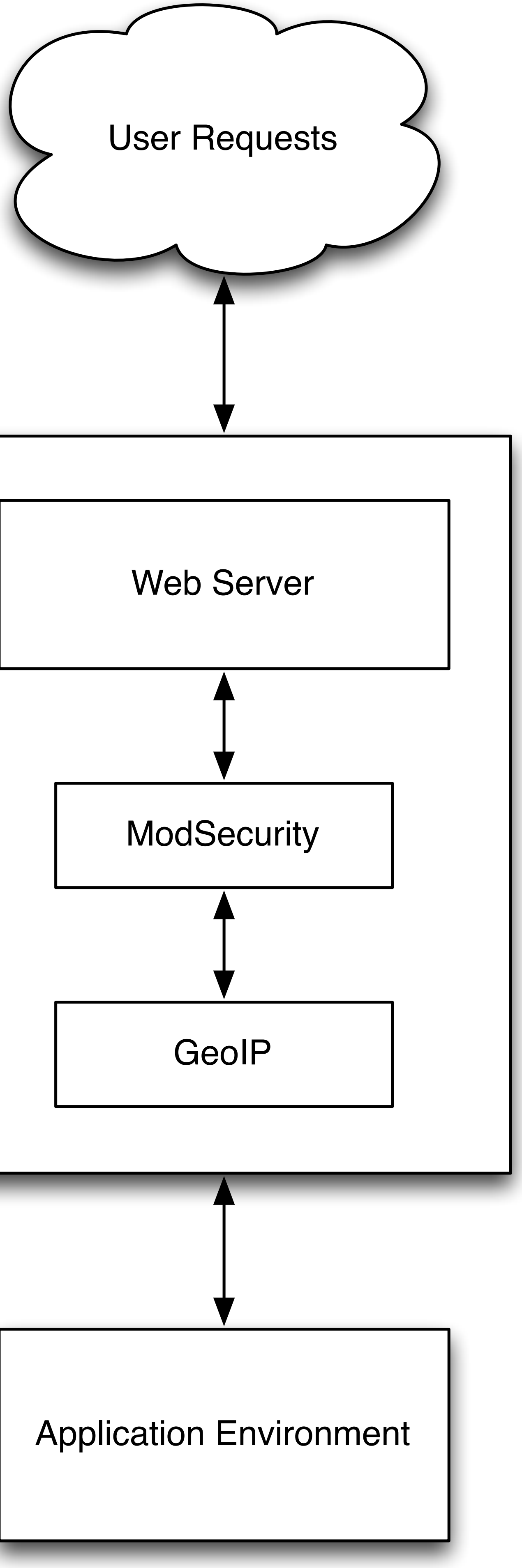
You realize that King Roland always logs in from Druidia

But the hacker is requesting the reset from Spaceball City

Instead of sending the reset, you now ask some questions

And hopefully protect King Roland from further bad actions

GeoIP detection also helps you block traffic from unwanted countries


User Requests
Web Server
ModSecurity
GeoIP
Application Environment

Other Anomalies

- Request Rate
- TCP Fingerprint vs. User Agent
- Account Create/Delete/Subscribe
- Anything you can imagine

What do they have in common?

Does the behavior fit an equation?

If so, your detection is simple

Request rate > Threshold

TCP fingerprint != User Agent

But the HTTP method deviation is harder

100% GET requests with a known UA (e.g. Google) is ok

100% POST requests is not

But it’s not always that simple

Scenario

A high rate of account create requests are coming from a single address

Is it a NATted IP or a fraud/spam bot?

We have patterns and data…

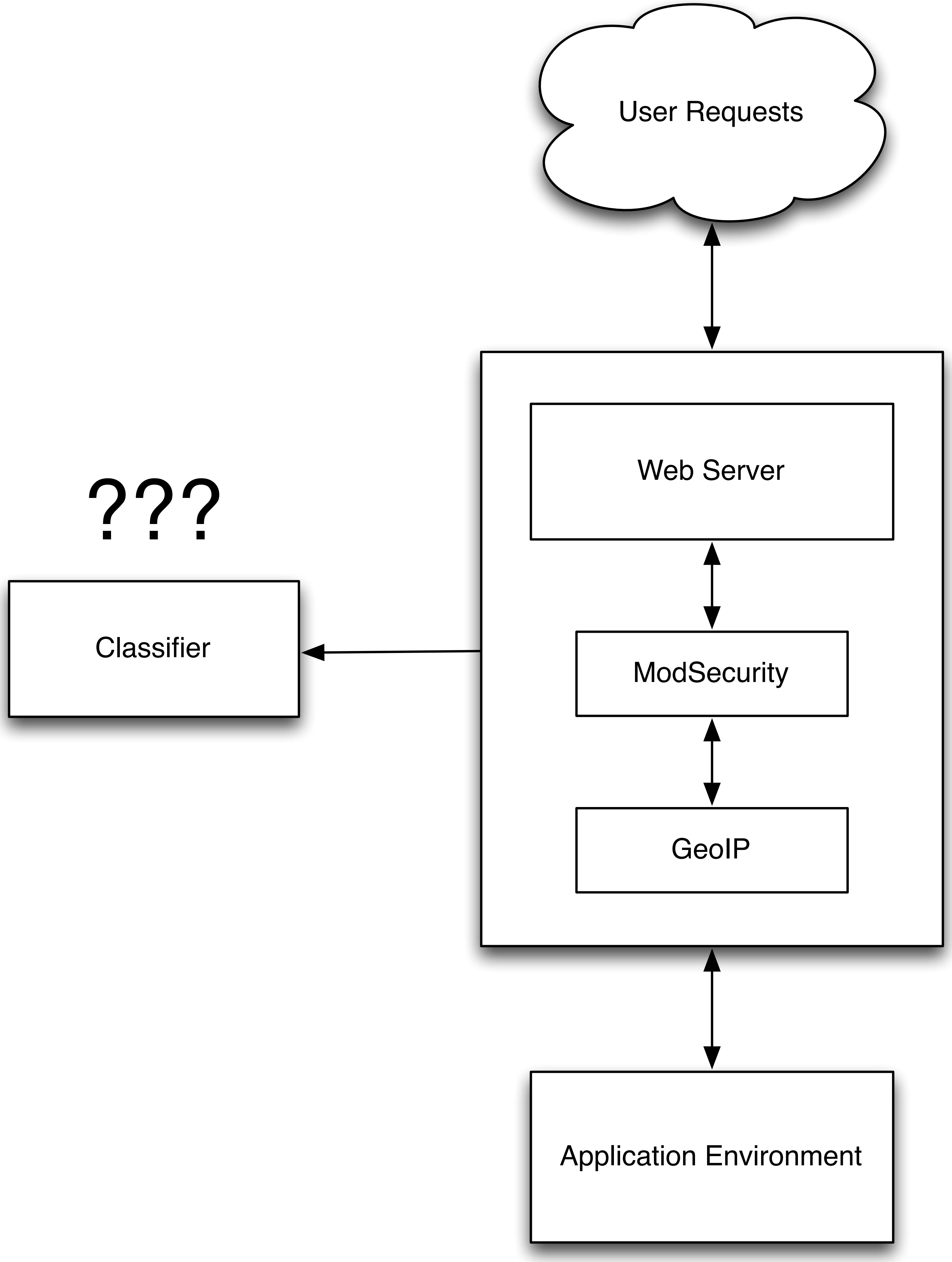
What’s the next step?

Quantitative Analysis

~~Quantitative Analysis~~

Security as a Data Science Probelm

We can apply some machine learning to the data in an attempt to classify it


User Requests
Web Server
???
Classifier
ModSecurity
GeoIP
Application Environment

This is where a lot of the value comes from

And combined with signature detection helps correlate attack events

But you still need a way to keep track of it all

Reputational Intelligence

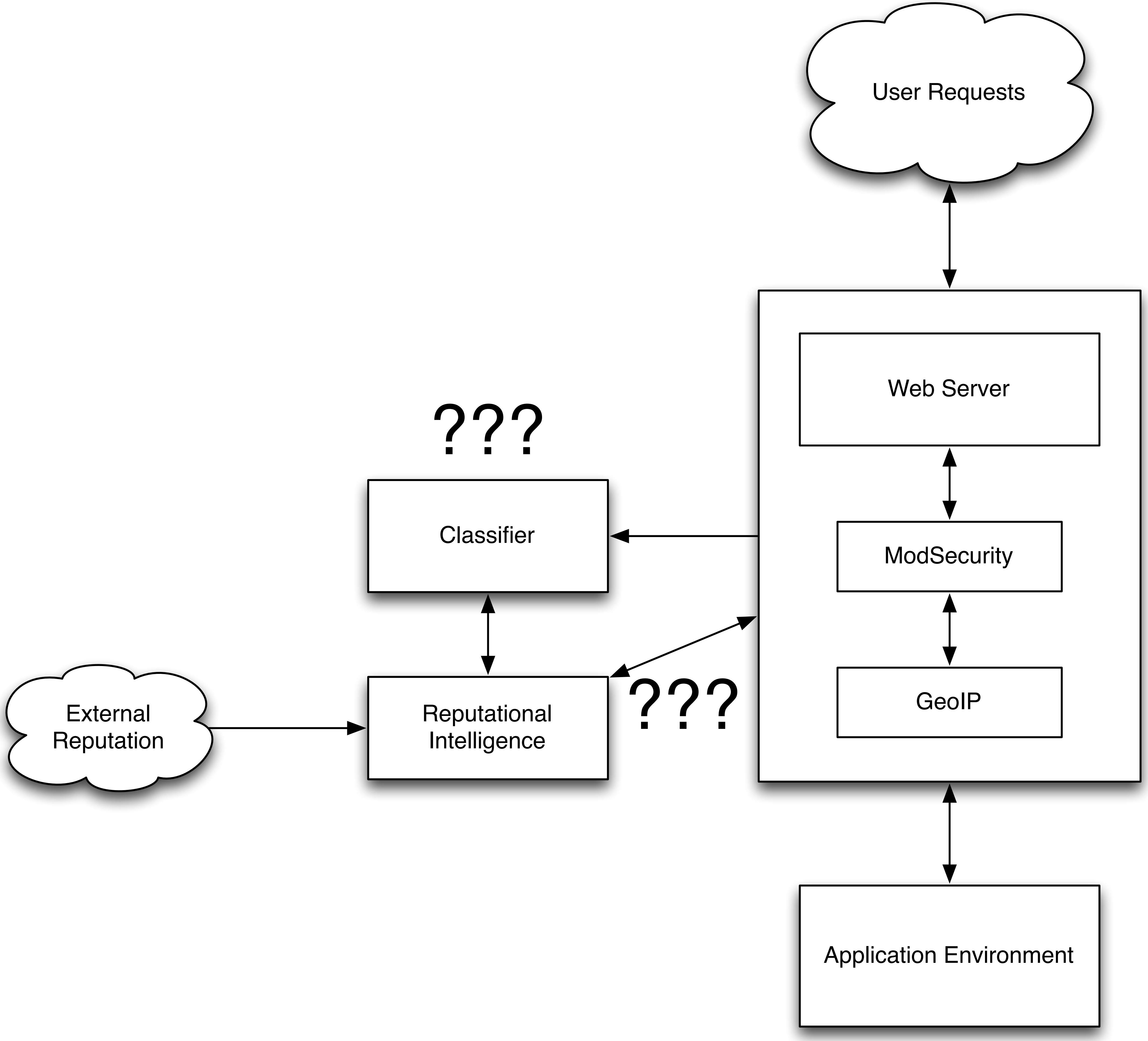
Who’s naughty and who’s really naughty

Built up from the tools/ techniques mentioned previously

Provides local reputation

You can also purchase external reputation feeds

The combination gives you solid awareness of bad actors


User Requests
Web Server
ModSecurity
GeoIP
???
Classifier
External
Reputation
Reputational
Intelligence
???
Application Environment

Action

So now you have a ton of new information

What do you do with it?

Options

- Block the traffic
- Honeypot the attacker
- Attack back
- Contact the authorities

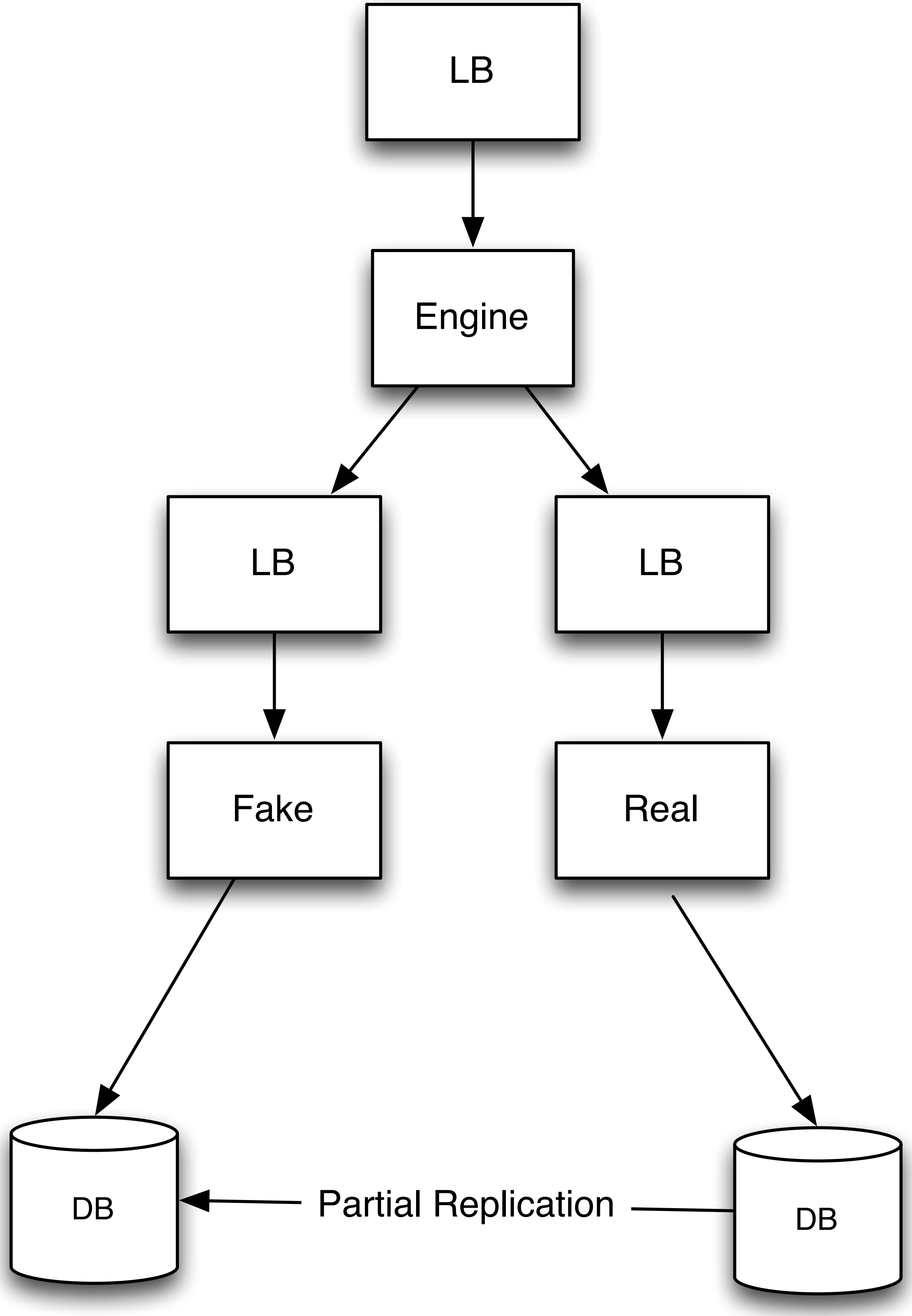
Blocking the traffic is straight forward

Block at the web server level (403)

Block at the firewall level

Both have advantages/ disadvantages

Honeypots are much more interesting

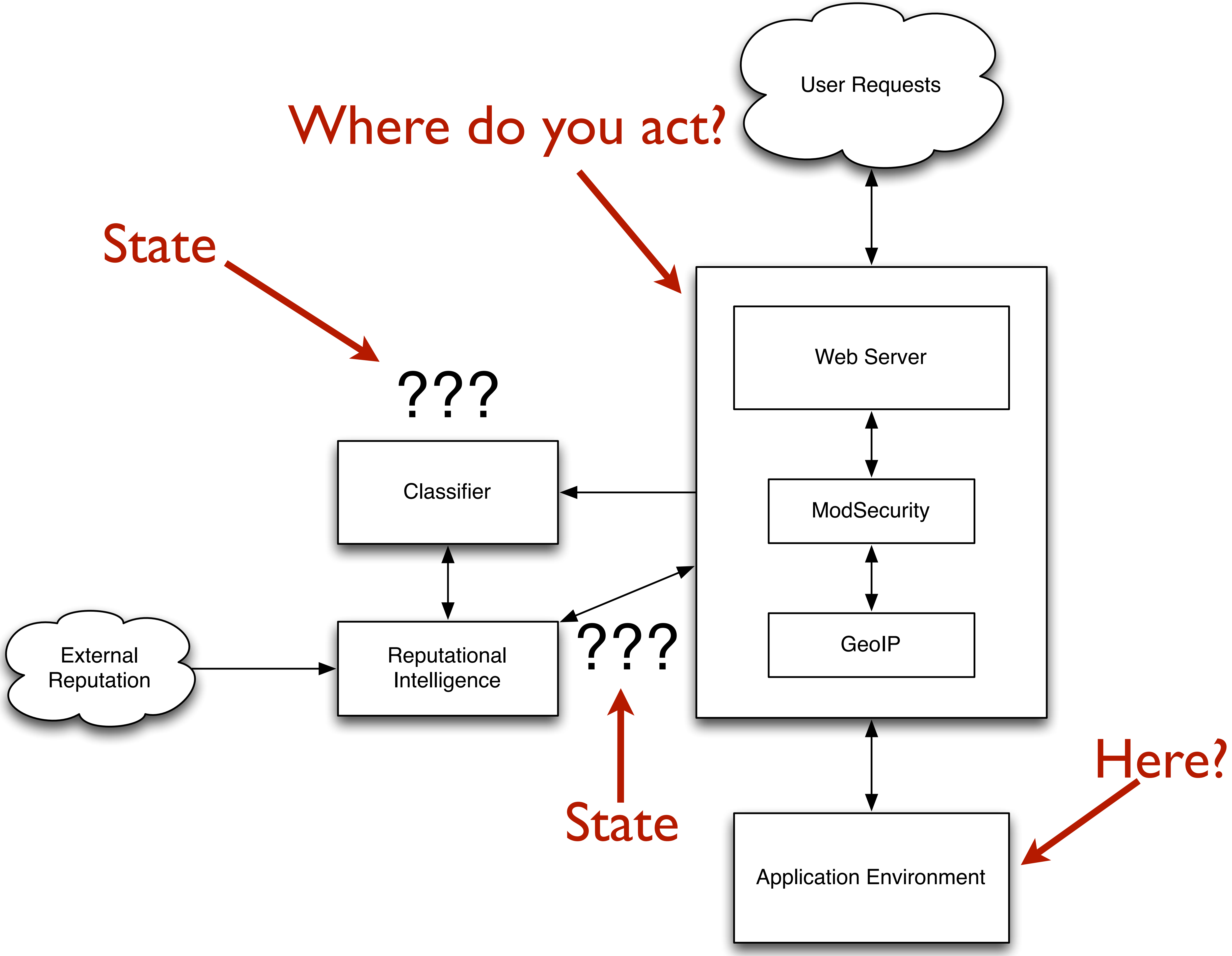

LB
Engine
LB
LB
Fake
Real
DB
Partial Replication
DB

When you honeypot, the attacker doesn’t know they’ve been caught

And it allows you to study their behavior

And update your approach to preventing attacks

But all of this requires a way to manage state and act on bad behavior



Where do you act?

User Requests

State

???

Web Server

Classifier

ModSecurity

External Reputation

Reputational Intelligence

???

GeoIP

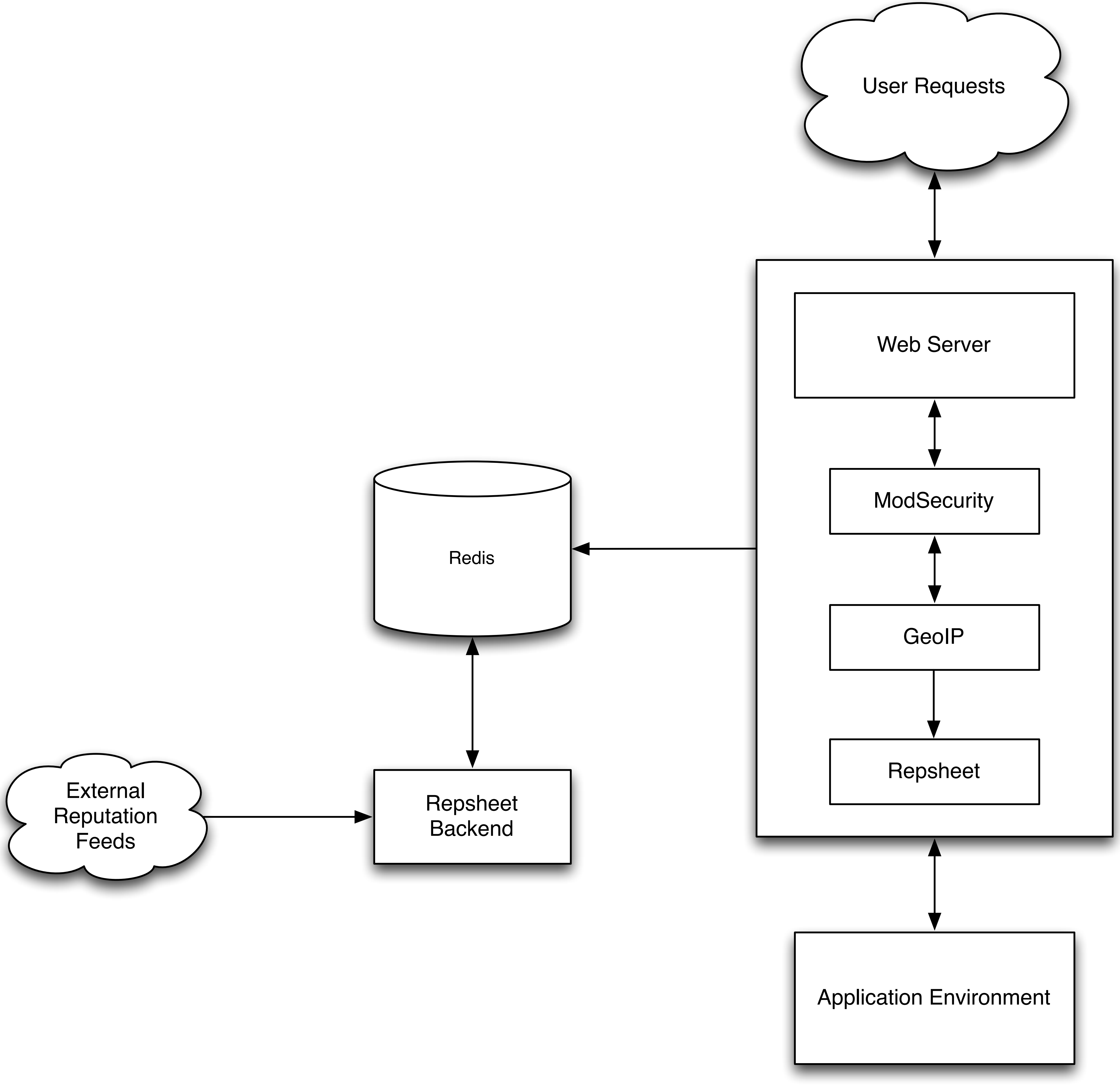
Here?

State

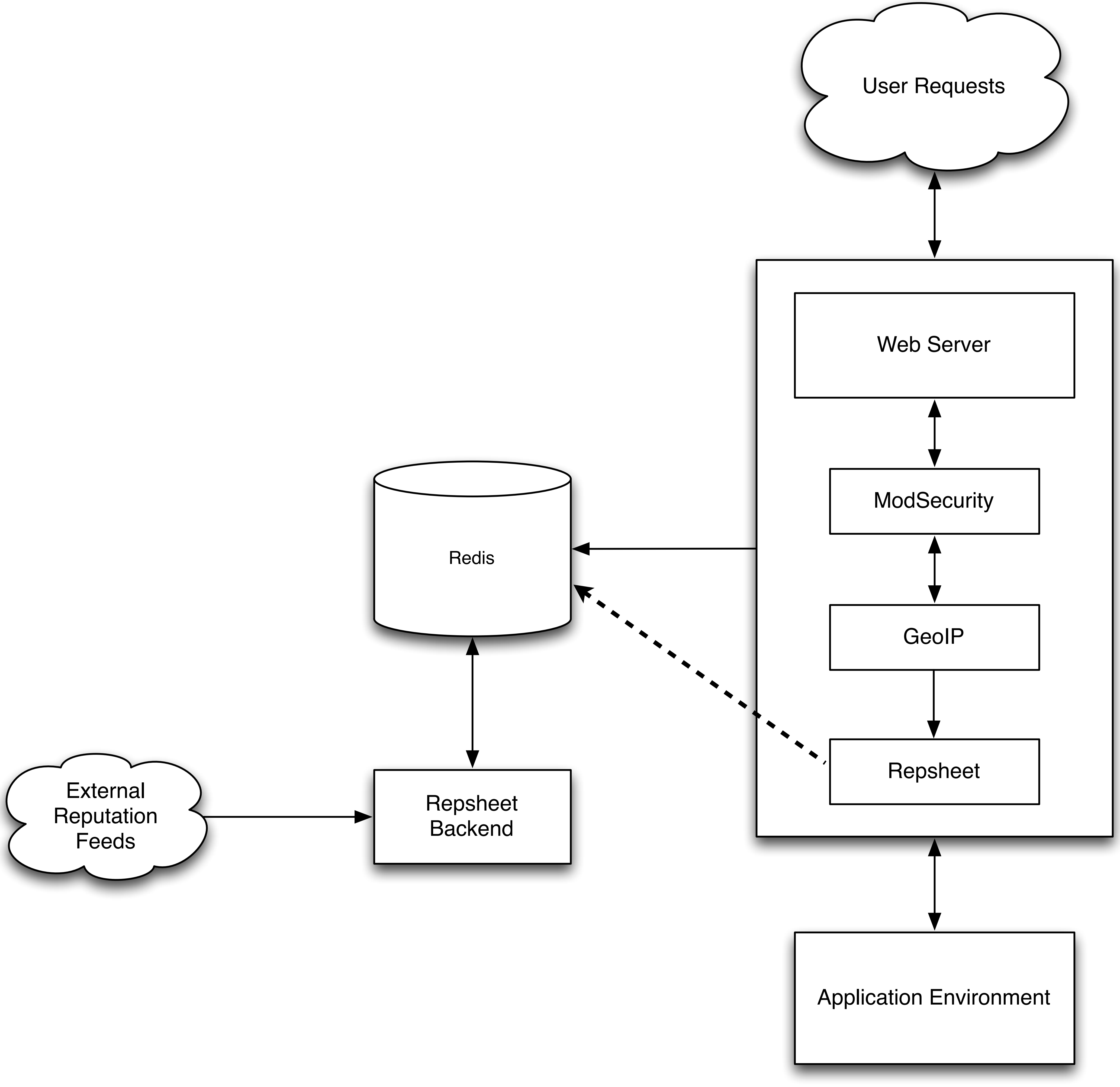
Application Environment

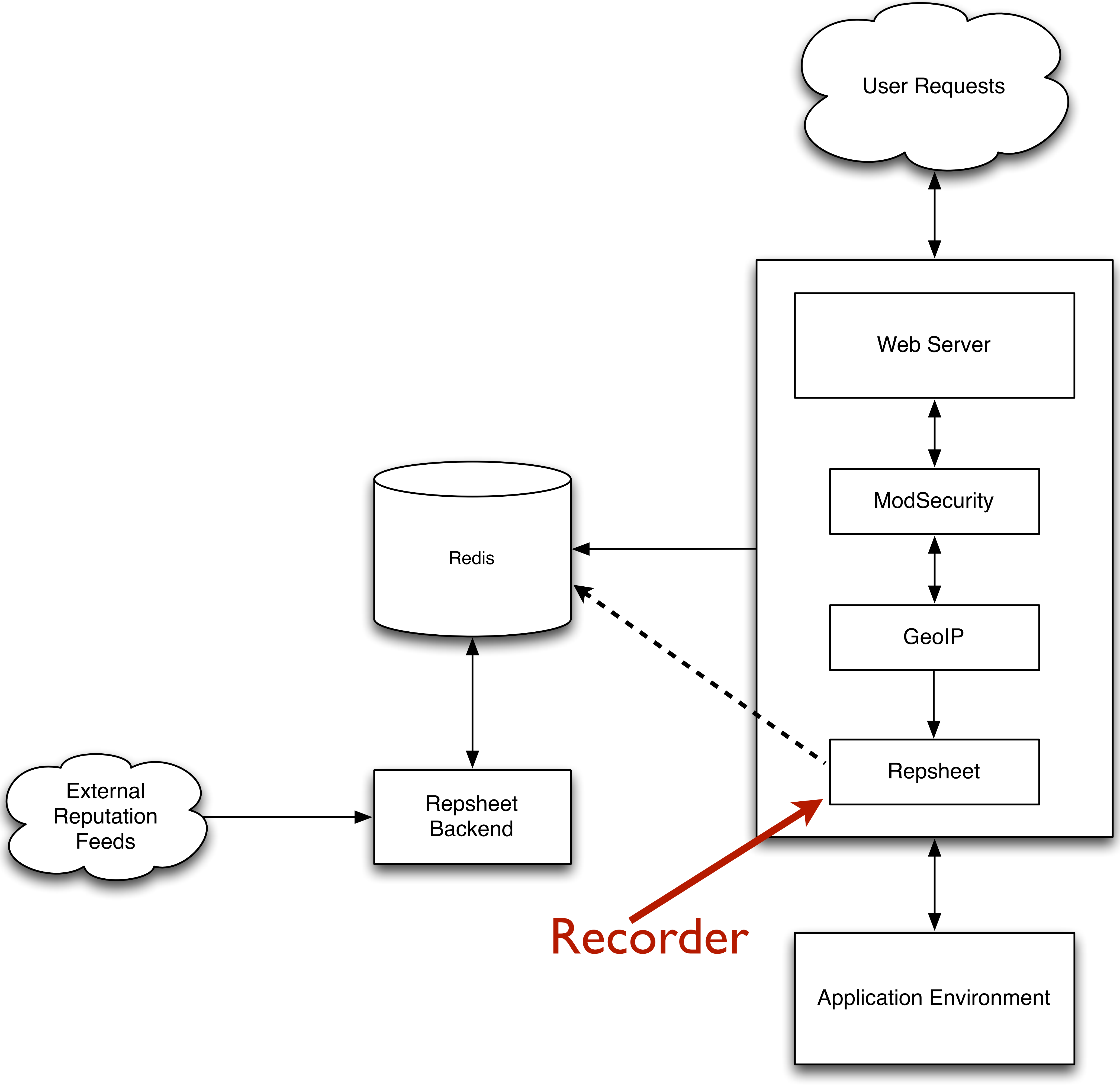
Repsheet

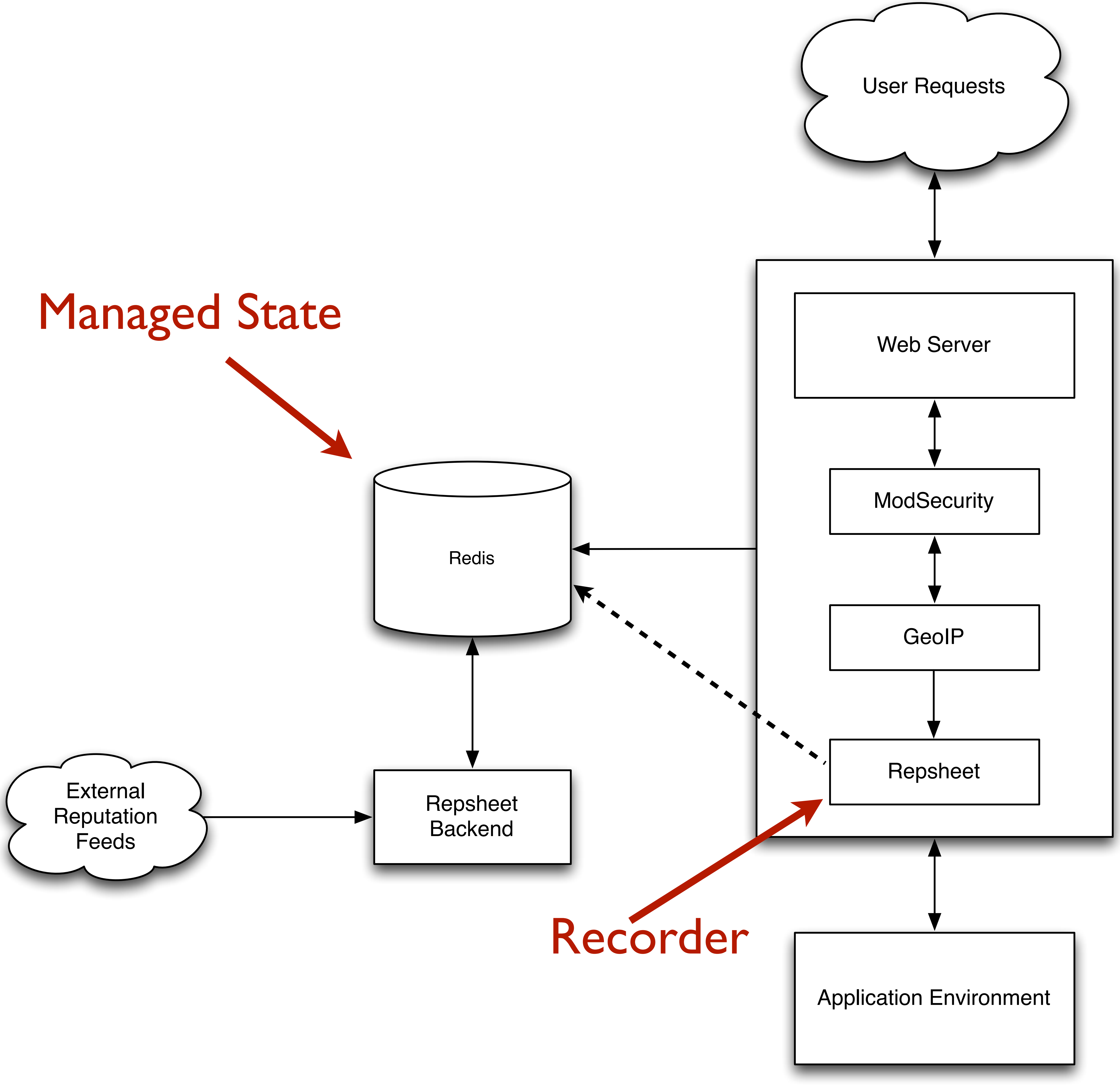
Reputation Engine



User Requests
Web Server
ModSecurity
GeoIP
Repsheet
Redis
External Reputation Feeds
Repsheet Backend
Application Environment


User Requests
Web Server
ModSecurity
GeoIP
Repsheet
Redis
External Reputation Feeds
Repsheet Backend
Application Environment


User Requests
Web Server
ModSecurity
GeoIP
Repsheet
Redis
External Reputation Feeds
Repsheet Backend
Recorder
Application Environment



Managed State

Recorder

User Requests

Web Server

ModSecurity

GeoIP

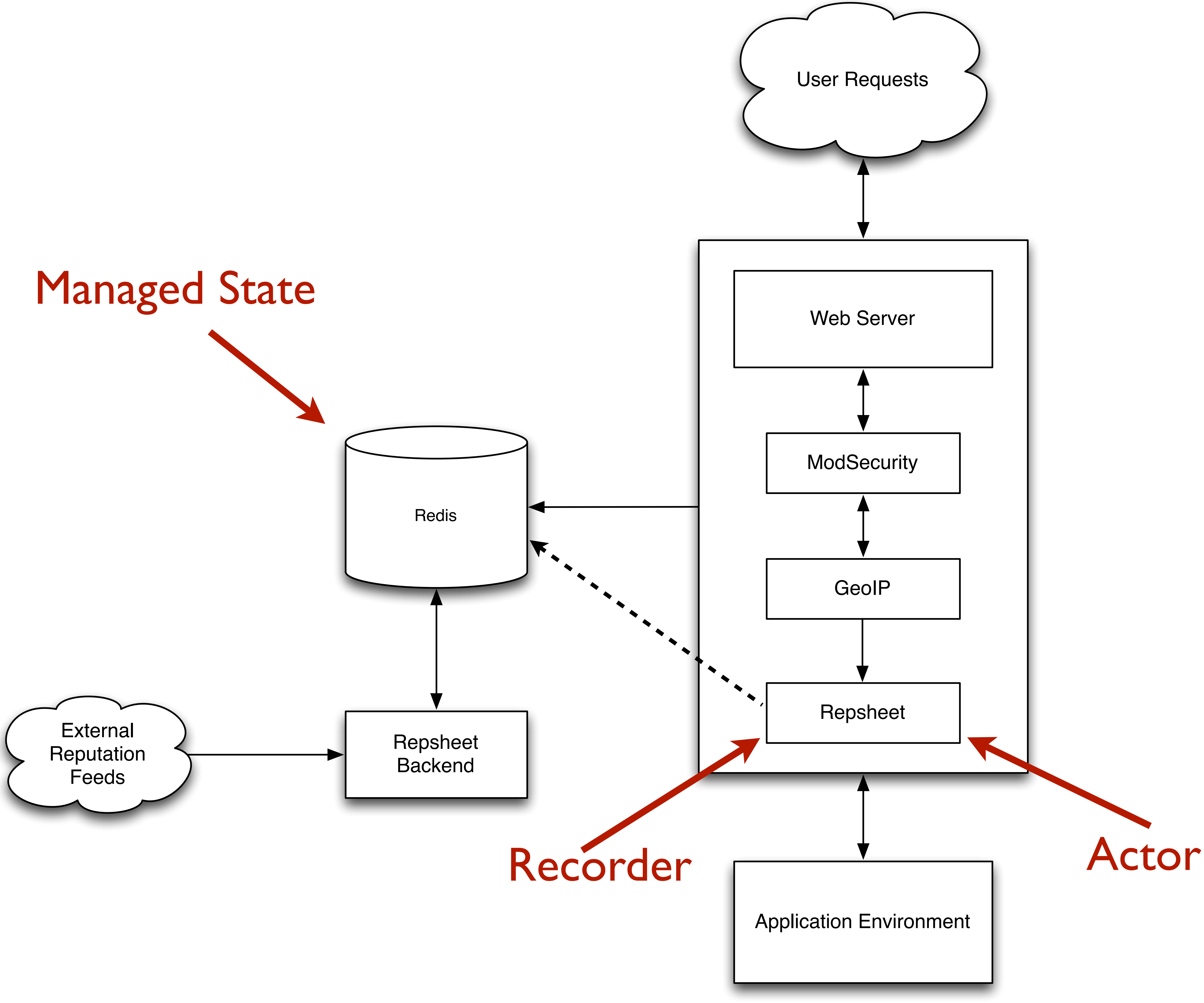
Repsheet

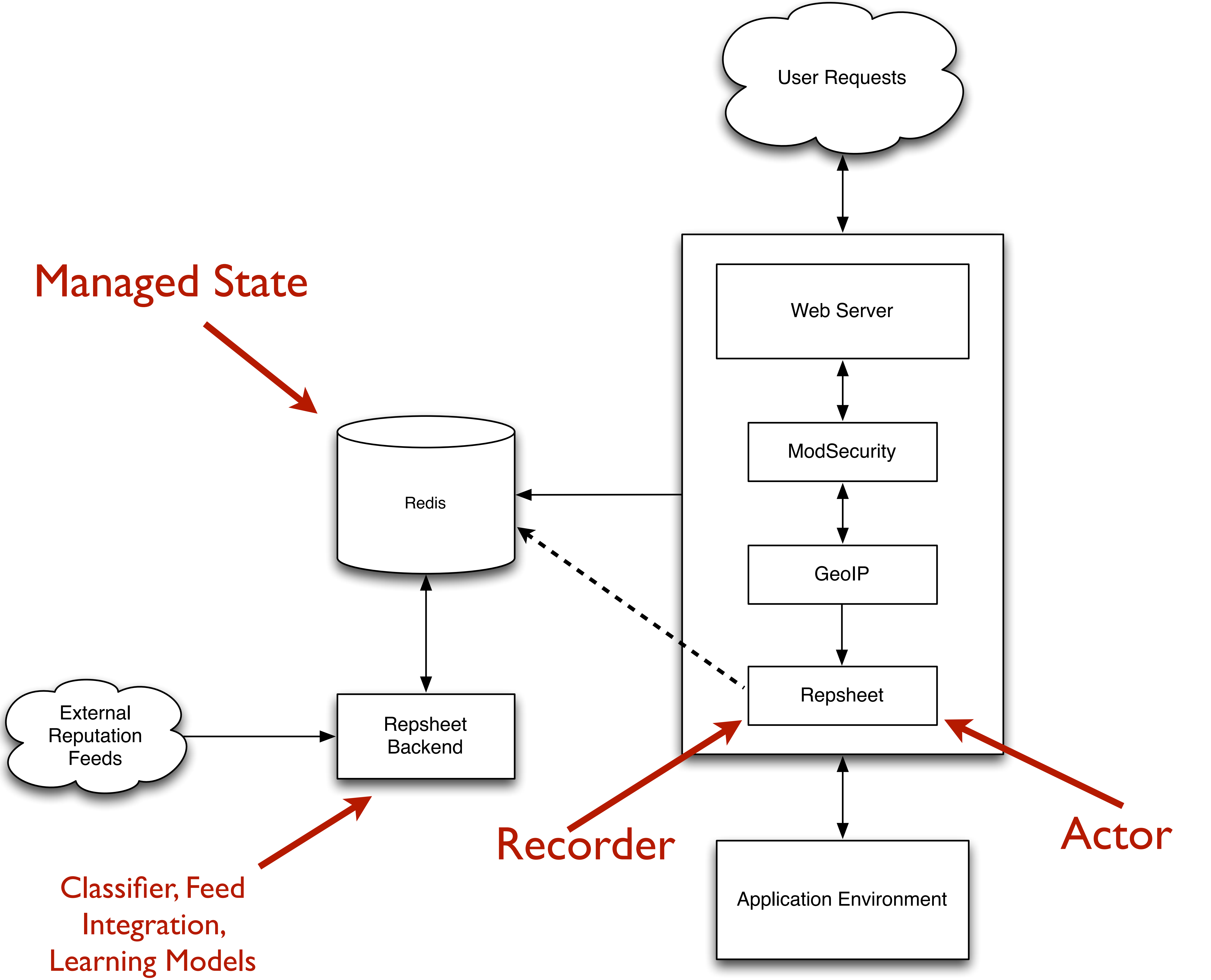
Redis

Repsheet Backend

External Reputation Feeds

Application Environment


User Requests
Web Server
ModSecurity
GeoIP
Repsheet
Application Environment
Redis
Repsheet Backend
External Reputation Feeds
Managed State
Recorder
Actor



User Requests

Web Server

ModSecurity

GeoIP

Repsheet

Application Environment

Redis

Repsheet Backend

External Reputation Feeds

Managed State

Recorder

Actor

Classifier, Feed Integration, Learning Models

Repsheet helps put everything together

Web server module records activity and looks for offenders in the cache

It listens to ModSecurity and adds offending IPs to it's list

It provides notification and/or blocking of offenders

Blocking happens at the web server level

But you can send the Repsheet data to your firewall for TCP level blocking

Notification sends headers to the downstream application

Which allows each app to chose how it is going to respond

For instance, show a captcha on signup if Repsheet alerts

Back end looks at the recorded data for bad behavior

And updates the cache when it finds offenders

You can supply your own learning models for the data

Repsheet will soon provide some defaults

github.com/abedra/
repsheet

Still in early stage development

But already in production for a few projects

Summary

There are lots of indicators of attack in your traffic

Build up a system that can capture the data and sort good from bad

Tools

- ModSecurity
- GeoIP
- Custom rules (velocity triggers, fingerprinting, device id, etc)
- Custom behavioral classification
- Repsheet

And Remember…

WINTER
IS
COMING
STARK

Questions?