OSSEC and Elasticsearch

David Crooks
for Scotgrid Glasgow

david.crooks@glasgow.ac.uk

HEPiX Spring 2015, Oxford University, 24th March 2015

Scotgrid Glasgow

- Part of GridPP/NGI_UK
- Member of distributed Tier 2 Scotgrid
- Sister sites
  - Edinburgh (ECDF)
  - Durham

Scotgrid Glasgow

- Part of GridPP/NGI_UK
- Member of distributed Tier 2 Scotgrid
- Sister sites
  - Edinburgh (ECDF)
  - Durham

OSSEC

- Host Intrusion Detection System
- Written by Daniel Cid
- About 10 years old
- http://www.ossec.net

Elasticsearch

- The ELK stack is an increasing popular tool in the analysis of a range of text based data.
  - Elasticsearch
  - Logstash
  - Kibana
- https://www.elastic.co (newly rebranded)

> *Elasticsearch is a flexible and powerful open source, distributed, real-time search and analytics engine.*

OSSEC & Elasticsearch

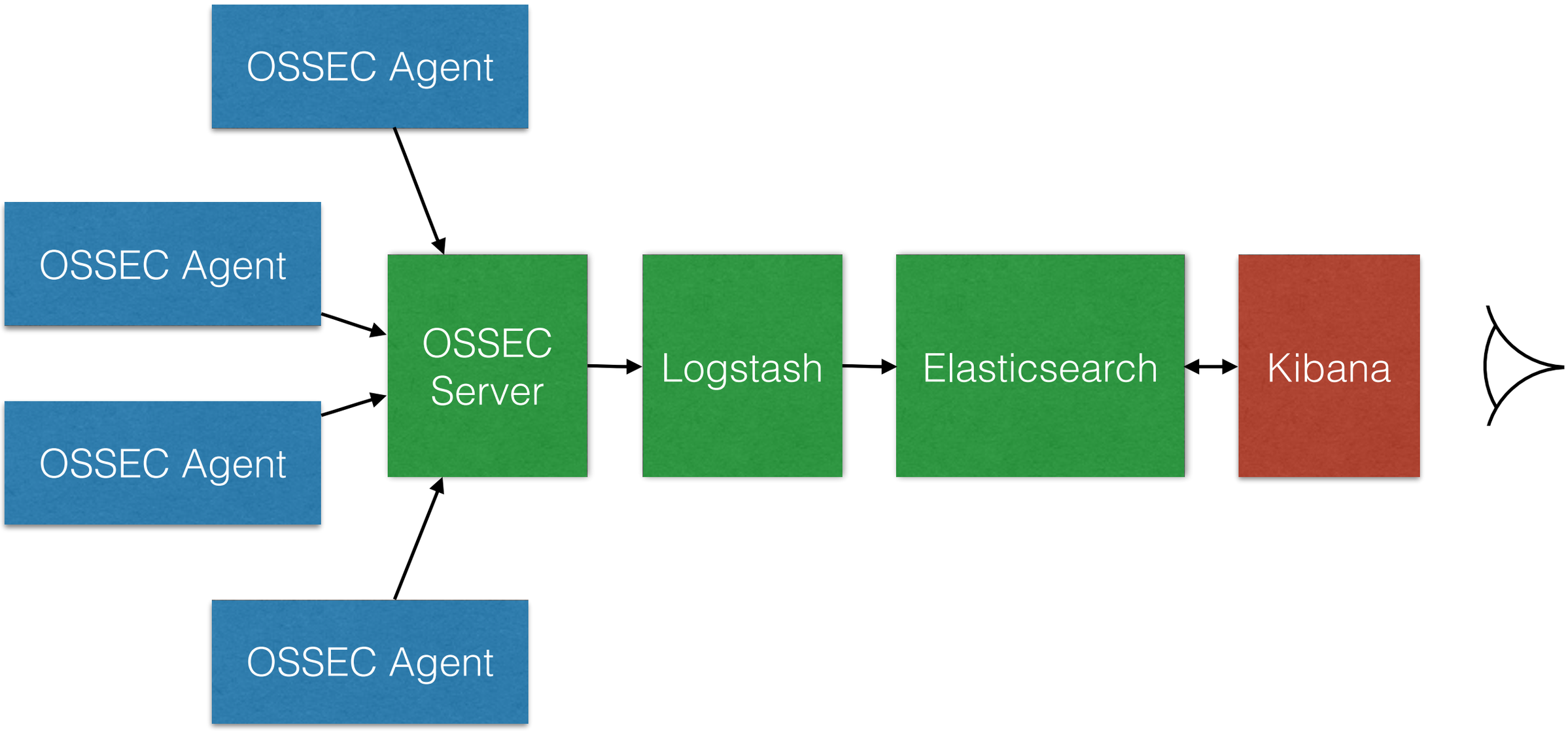
- Use the ELK stack to visualise the data aggregated by OSSEC.

- Useful blog: http://vichargrave.com

- http://vichargrave.com/ossec-log-management-with-elasticsearch/

- http://vichargrave.com/improved-ossec-log-parsing-with-logstash/

Data flow


OSSEC Agent
OSSEC Agent
OSSEC Agent
OSSEC Agent
OSSEC Server
Logstash
Elasticsearch
Kibana

OSSEC → Logstash

- OSSEC can output in syslog format. In `/var/ossec/etc/ossec.conf` add

```
<syslog_output>
  <server>SERVERNAME</server>
  <port>PORT</port>
  <format>default</format>
</syslog_output>
```

and restart `ossec`

- Logstash can then process the syslog data

logstash-syslog.conf

```
input {
    #stdin{}
    udp {
        port => LISTENINGPORT
        type => "syslog"
    }
}

filter {
    if [type] == "syslog" {
        grok {
            match => { "message" => "%{SYSLOGTIMESTAMP:syslog_timestamp} %{SYSLOGHOST:syslog_host} %
{DATA:syslog_program}: Alert Level: %{NONNEGINT:Alert_Level}; Rule: %{NONNEGINT:Rule} - %{DATA:Description};
Location: %{DATA:Location}; (user: %{USER:User};%{SPACE})?(srcip: %{IP:Src_IP};%{SPACE})?(user: %{USER:User};%
{SPACE})?(dstip: %{IP:Dst_IP};%{SPACE})?(src_port: %{NONNEGINT:Src_Port};%{SPACE})?(dst_port: %
{NONNEGINT:Dst_Port};%{SPACE})?%{GREEDYDATA:Details}" }
            add_field => [ "ossec_server", "%{host}" ]
        }
        mutate {
            remove_field => [ "message","syslog_timestamp", "syslog_program", "syslog_host", "syslog_message",
"syslog_pid", "@version", "type", "host" ]
        }
    }
}

output {
      #   stdout {
      #     codec => rubydebug
      #   }
    elasticsearch_http {
        host => "ESHOST"
    }
}
```

logstash-syslog.conf

```
input {
    #stdin{}
    udp {
        port => LISTENINGPORT
        type => "syslog"
    }
}

filter {
    if [type] == "syslog" {
        grok {
            match => { "message" => "%{SYSLOGTIMESTAMP:syslog_timestamp} %{SYSLOGHOST:syslog_host} %
{DATA:syslog_program}: Alert Level: %{NONNEGINT:Alert_Level}; Rule: %{NONNEGINT:Rule} - %{DATA:Description};
Location: %{DATA:Location}; (user: %{USER:User};%{SPACE})?(srcip: %{IP:Src_IP};%{SPACE})?(user: %{USER:User};%
{SPACE})?(dstip: %{IP:Dst_IP};%{SPACE})?(src_port: %{NONNEGINT:Src_Port};%{SPACE})?(dst_port: %
{NONNEGINT:Dst_Port};%{SPACE})?%{GREEDYDATA:Details}" }
            add_field => [ "ossec_server", "%{host}" ]
        }
        mutate {
            remove_field => [ "message","syslog_timestamp", "syslog_program", "syslog_host", "syslog_message",
"syslog_pid", "@version", "type", "host" ]
        }
    }
}

output {
      #   stdout {
      #     codec => rubydebug
      #   }
    elasticsearch_http {
        host => "ESHOST"
    }
}
```

Kibana

- Kibana 3
- Unpack in web directory
- Configure to point to local elasticsearch instance

```
elasticsearch: "http://ESHOST:PORT",
```

CORS & Elasticsearch

- Cross-Origin Resource Sharing

> *The main motivation behind Cross-Origin Resource Sharing (CORS) was to remove the same origin restriction from various APIs so that resources can be shared among different origins (i.e. servers).*

- http://www.w3.org/wiki/CORS

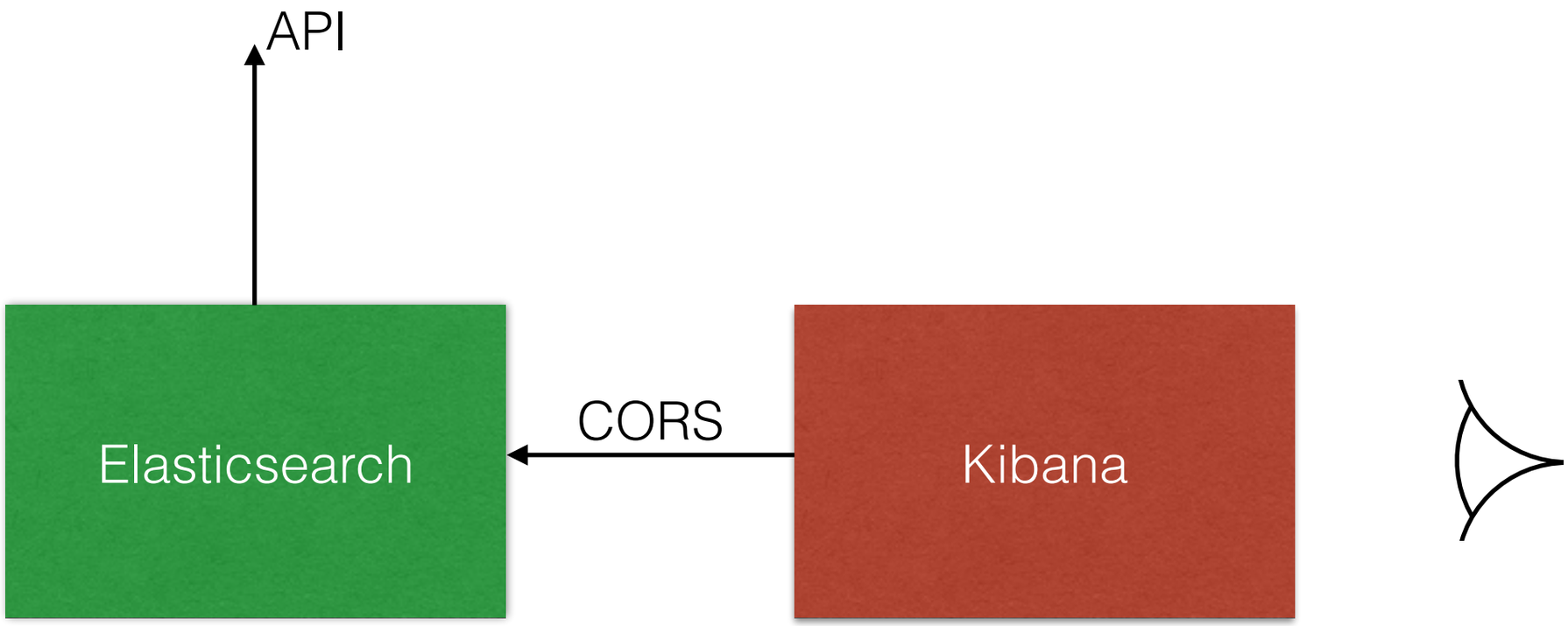
- If accessing Elasticsearch from a Kibana instance on a different domain, require in `/etc/elasticsearch/elasticsearch.yaml`

```
http.cors.enabled = true
http.cors.allow-origin = "KIBANAHOST"
```

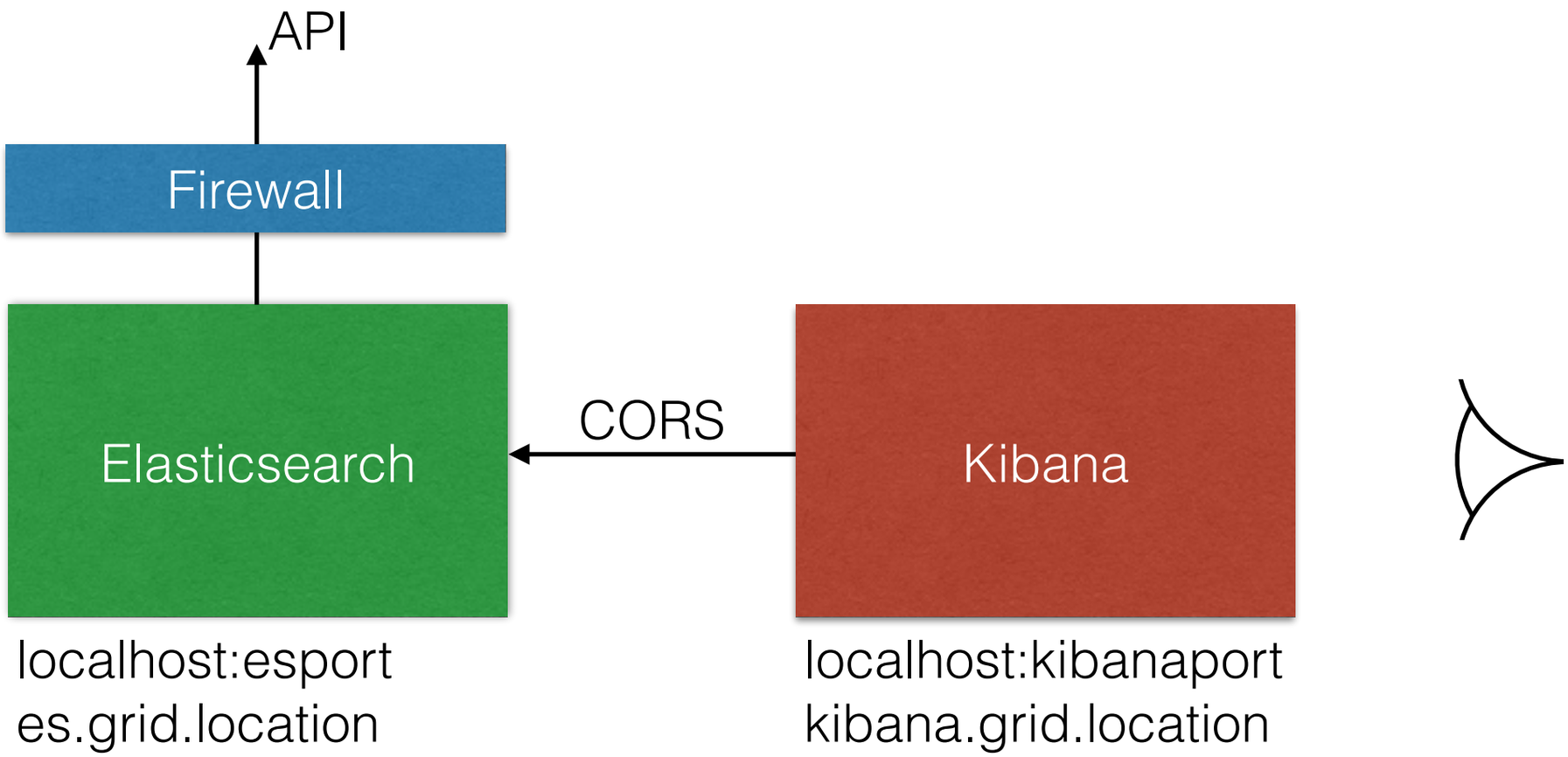
Securing Elasticsearch

- Elasticsearch provides a very flexible interface which can lead to exposure of data
- Does not provide security layer as part of initial product
- Take appropriate steps to secure installation
- (Elastic has released a subscription-based security product, “shield”: https://www.elastic.co/products/shield)

Securing Elasticsearch



Securing Elasticsearch



Secure Elasticsearch service from external access: Firewall

Securing Elasticsearch

- Secure access to Elasticsearch/Kibana
- Use https/authentication by grid cert
  - SSL configuration
  - Reverse proxy for elasticsearch
- Limit by IP range if desired

Securing Elasticsearch

- Reverse proxy elasticsearch

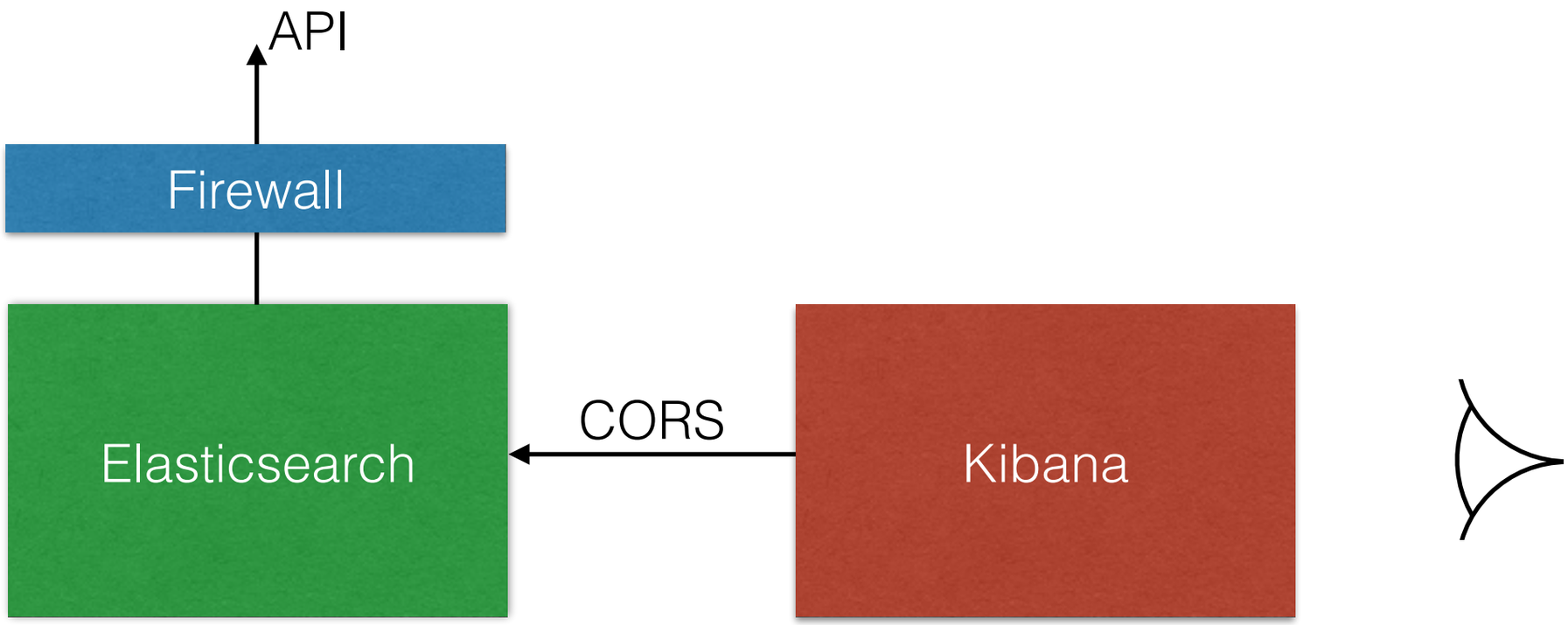
```
ProxyRequests off
ProxyPass /elasticsearch/ http://localhost:PORT/

<Location /elasticsearch/>
 ProxyPassReverse /
 SSLRequireSSL
</Location>
```

- Kibana

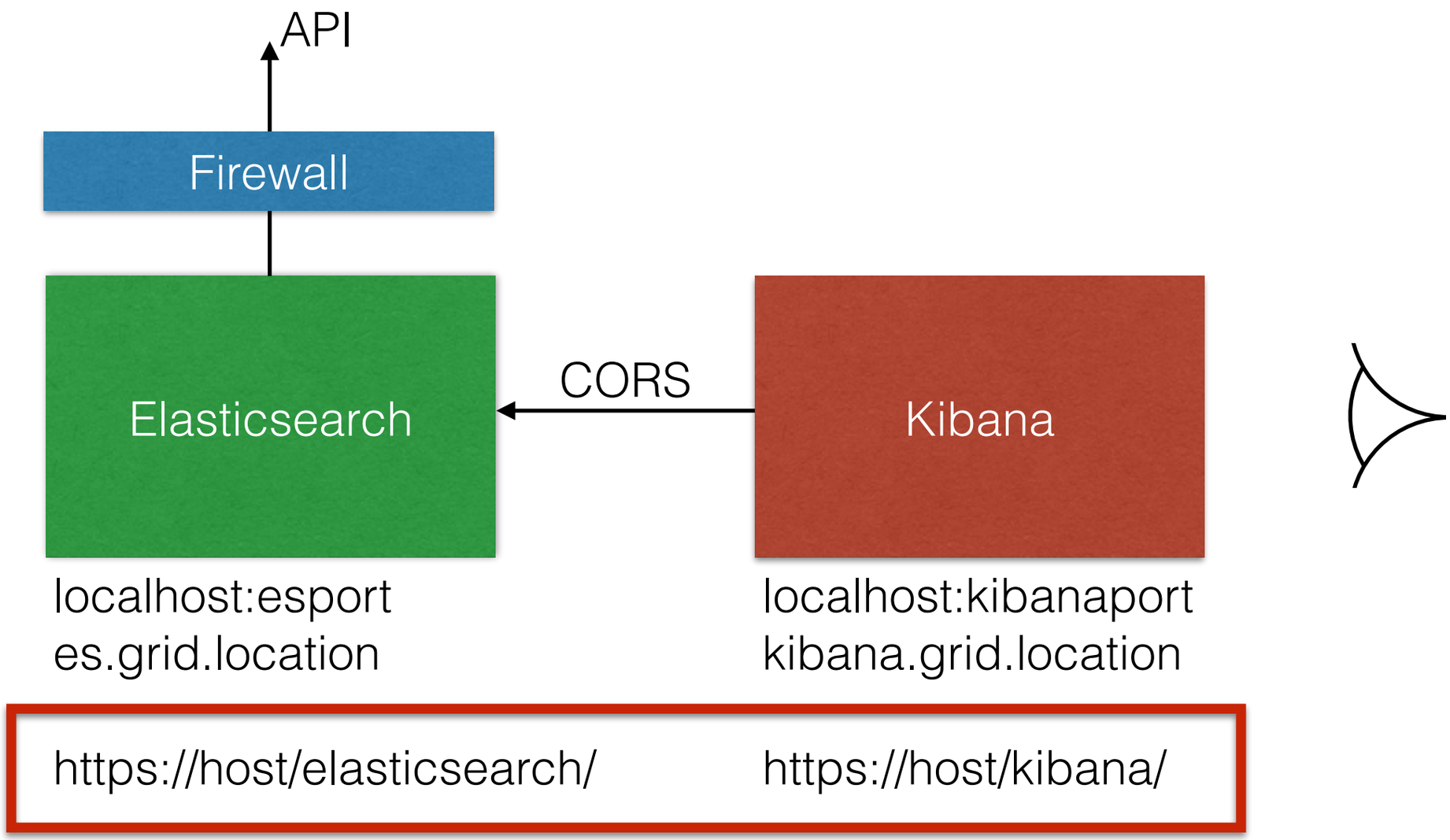
```
elasticsearch: "https://ESHOST/elasticsearch",
```

- CORS settings not necessary in this case

Securing Elasticsearch



Securing Elasticsearch



Grid additions

- One of the most useful features of OSSEC on installation is its wide range of initial rules

- Grid has its own set of logs and flags - we have followed process of picking out “unknown problems” and flagging them with a downstream rule.

- Details of local install available for interest.

Example

- One test available is to look for possible rootkit events through hidden files
- Interpretation important
- One node, for example, which showed large set of these triggers (node showed problems elsewhere which caused further investigation)
- Using Kibana it was straightforward to
  - isolate that node
  - pick out the rootcheck warnings
  - *remove* the rootcheck warnings
  - observe that in fact the node was throwing SATA errors - not a rootkit, but a bad hard disk

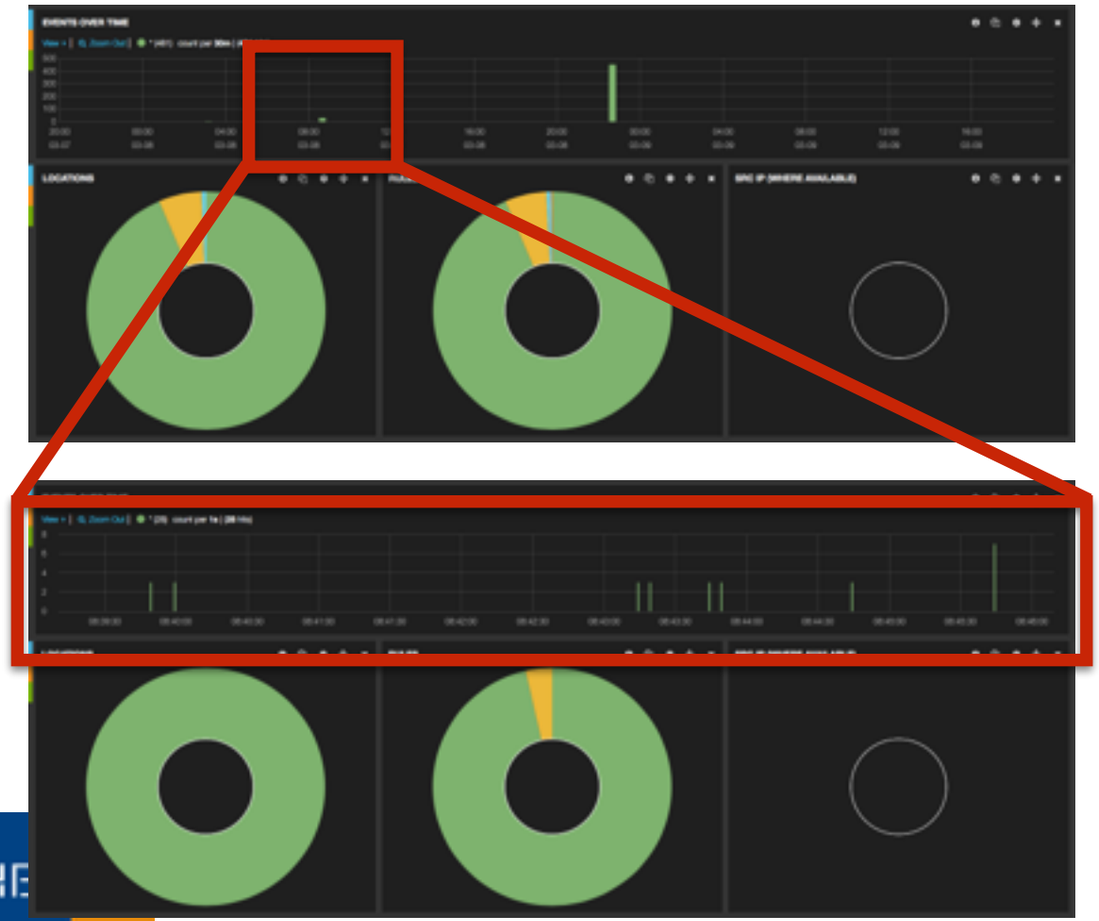
Example

48h view, whole cluster

~4h view, whole cluster

Example

~4h view, specific node



~24h view, specific node

~10min view, specific node

Conclusions & Forward look

- Largely in data gathering mode; establishing baseline
  - Common issues
- Use as system tool
- Integration with other site Elasticsearch initiatives (or run in parallel)
- Scaling
- ~ 1G of data over 4 months
- Currently using Kibana 3 with Apache
- Kibana 4? Repackaged as an elasticsearch plugin