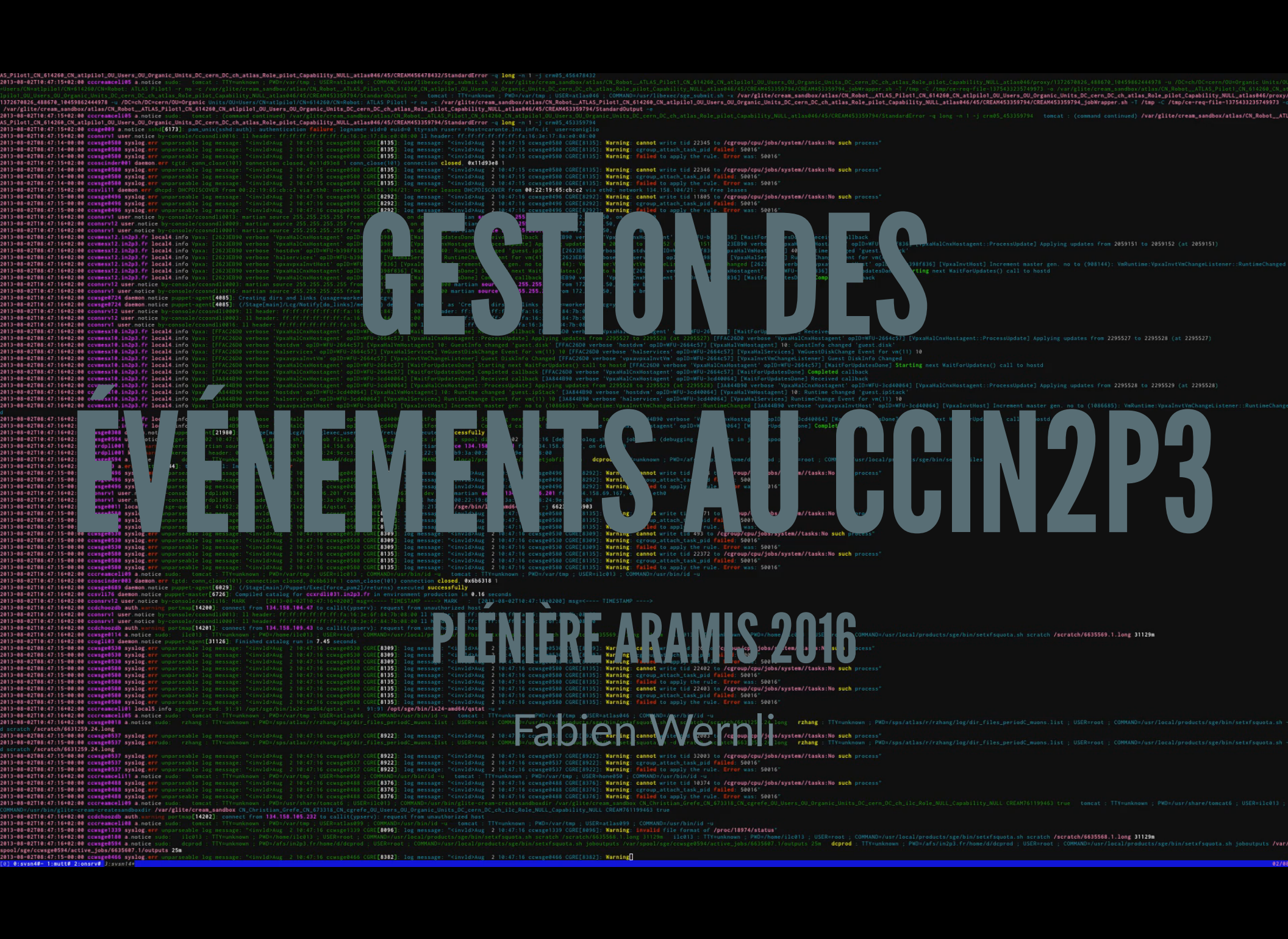

GESTION DES
ÉVÉNEMENTS AU CCIN2P3
PLÉNIÈRE ARAMIS 2016
Fabien Wernli

PLAN

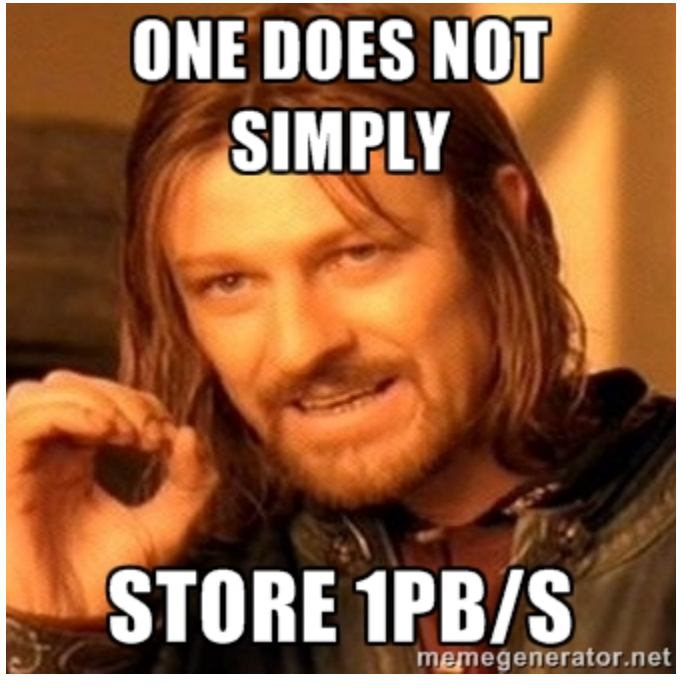
- contexte
- événements
- méthodes traditionelles
- big events: épingle dans botte de foin
- architecture actuelle
- exemples actuels
- perspectives

LARGE HADRON COLLIDER


LHCb
ATLAS
CERN Meyrin
CERN Prévessin
SPS 7 km
SUISSE
FRANCE
CMS
ALICE
LHC 27 km

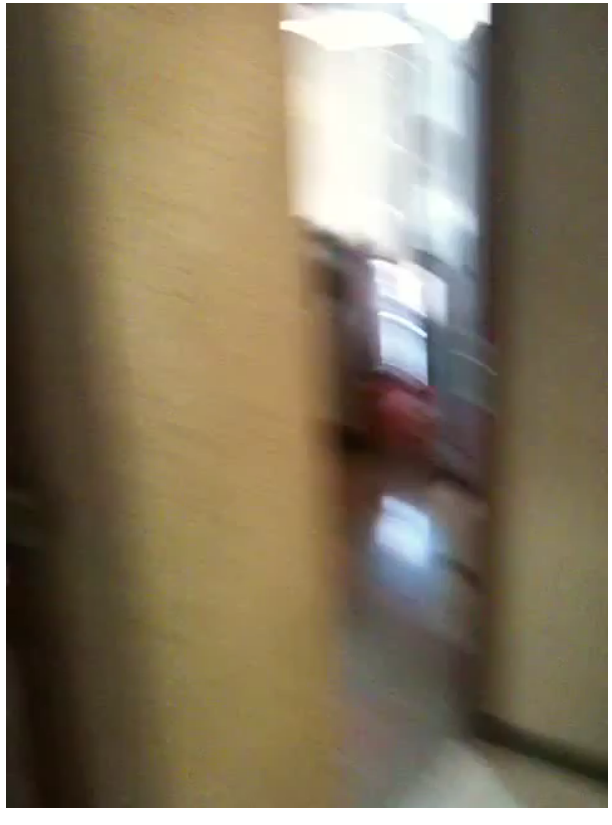
1 MILLIARD DE COLLISIONS PAR SECONDE

1 PO/S

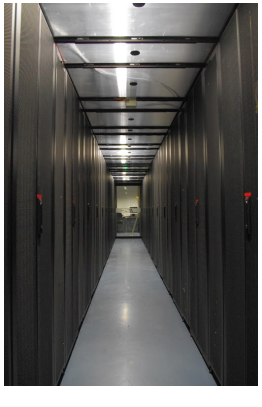


Heureusement il y a les *triggers* ⇒ 1 Go/s

CCIN2P3




Sun



ÉVÉNEMENTS

- 2 types d'évenements
  - journaux ("logs")
    - syslog (`/var/log/messages`)
  - métriques ("metrics")
    - collectd (`/proc/...`)
- 2 cas d'utilisation principaux
  - analyse post-incident
  - alerting et control-room

- Journaux
  - "push"
  - horodatés
  - message + clé/valeurs
- Métriques: pull
  - "pull" régulier
  - horodatés
  - nombre + clé/valeurs

CCIN2P3: CHIFFRES

- Journaux : ~ 100 million par jour
- Métriques : ~ 1 milliard par jour
- points de mesure : ~ 400'000
- unités : ~ 2400 serveurs, alims, clims, onduleurs, générateur, …

LHC génère la même quantité d'événements chaque seconde…

EXEMPLES

MÉTRIQUES

```
ccsvli64.in2p3.fr     cpu/cpu-idle                          ok   99.53
ccosvms0034.in2p3.fr  interface-eth0/if_octets/rx           ok   172.1
ccwntest14.in2p3.fr   df-var/percent_bytes-free    critical    0.0
```

JOURNAUX

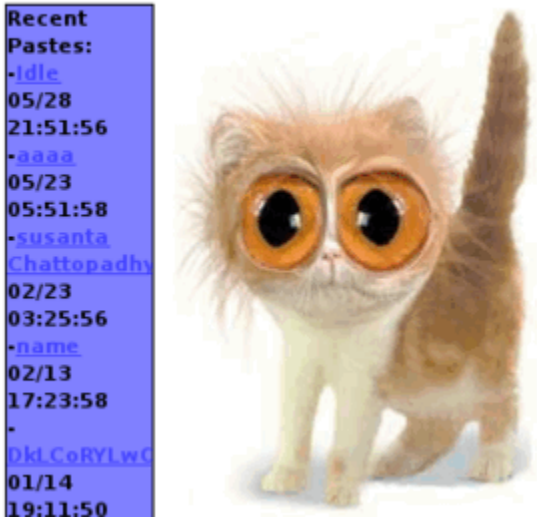
```
kernel: Killed process 29959, UID 42046, (hadd) total-vm:202363492kB, ar
ata2.00: exception Emask 0x0 SAct 0xffff SErr 0x0 action 0x0
puppet-agent[16528]: Finished catalog run in 44.06 seconds
```

MÉTHODES TRADITIONELLES

ANALYSE POST-INCIDENT

```
                          wc
                         find
                        less
                       sed
/var/log/messages     bc
                     getent
       ou          → grep
                     awk
syslog centralisé     cat
                       tail
                        cut
                         uniq
                          paste
```

Useless Use Of Cat



EXEMPLE: CLIENTS DHCP

```
2016-04-01T18:15:57+02:00 ccosctl05 <daemon.info> dnsmasq[25597]: DHCPACK(br816) 172.17.16.6 fa:16:
3e:b4:a8:d6 cloudatlas2-7ba664b9-c6c8-4b52-b9fa-3f30a9b84a0f
2016-04-01T19:48:01+02:00 ccosctl05 <daemon.info> dnsmasq[31067]: DHCPACK(br800) 172.17.0.58 fa:16:
3e:f3:87:fd ccosvm0818
2016-04-01T19:49:30+02:00 ccosctl05 <daemon.info> dnsmasq[31067]: DHCPACK(br800) 172.17.0.58 fa:16:
3e:f3:87:fd ccosvm0818
2016-04-01T19:50:58+02:00 ccosctl05 <daemon.info> dnsmasq[31067]: DHCPACK(br800) 172.17.0.58 fa:16:
3e:f3:87:fd ccosvm0818
2016-04-01T19:52:27+02:00 ccosctl05 <daemon.info> dnsmasq[31067]: DHCPACK(br800) 172.17.0.58 fa:16:
3e:f3:87:fd ccosvm0818
2016-04-01T19:53:56+02:00 ccosctl05 <daemon.info> dnsmasq[31067]: DHCPACK(br800) 172.17.0.58 fa:16:
3e:f3:87:fd ccosvm0818
2016-04-01T19:55:25+02:00 ccosctl05 <daemon.info> dnsmasq[31067]: DHCPACK(br800) 172.17.0.58 fa:16:
3e:f3:87:fd ccosvm0818
2016-04-01T19:56:58+02:00 ccosctl05 <daemon.info> dnsmasq[31067]: DHCPACK(br800) 172.17.0.58 fa:16:
3e:f3:87:fd ccosvm0818
2016-04-01T21:09:37+02:00 ccosctl05 <daemon.info> dnsmasq[31124]: DHCPACK(br802) 172.17.2.7 fa:16:3
e:f6:d6:66 testwin2
2016-04-01T21:13:20+02:00 ccosctl05 <daemon.info> dnsmasq[31124]: DHCPACK(br802) 172.17.2.3 fa:16:3
e:d6:49:d4 testwin
2016-04-01T23:32:26+02:00 ccosctl05 <daemon.info> dnsmasq[25597]: DHCPACK(br816) 172.17.16.27 fa:16
:3e:09:1b:17 cloudatlas2-daab1702-c217-4e0f-9bf4-d93fc2a97da9
2016-04-01T23:33:29+02:00 ccosctl05 <daemon.info> dnsmasq[25597]: DHCPACK(br816) 172.17.16.27 fa:16
:3e:09:1b:17 cloudatlas2-daab1702-c217-4e0f-9bf4-d93fc2a97da9
int m,mac[m]ui15/01(0)$ awk '$4 ~ /dnsmasq/ && $5 ~ /DHCPACK/ {mac[$7]++} END {for (m in mac) {pr
[10] 0:ssh*                                                                    ccfawe 03/04
```

```
awk '$4 ~ /^dnsmasq/ && $5 ~ /DHCPACK/ {mac[$7]++} END {for (m in mac) {
```

EXEMPLE: CONNEXIONS RÉUSSIES PAR UTILISATEUR

```
################################################################################

                  Bienvenue au centre de calcul de l'IN2P3
                    en partenariat avec le CEA/DSM/IRFU

 --------------------------------------------------------------------------------

                   Welcome to the IN2P3 computing center
                     in partnership with CEA/DSM/IRFU

################################################################################
.profile
fwernli@ccsvli15/~(0)$ cd /var/syslog-ng/remote/2016/04/01
fwernli@ccsvli15/01(0)$ grep -iP 'sshd.*Accepted.*password' by-host/ccosvms* | tail -n3
by-host/ccosvms0064:2016-04-01T12:14:27+02:00 ccosvms0064 <auth.info> sshd[21474]: Accepted passwor
d for calvat from 134.158.231.65 port 52693 ssh2
by-host/ccosvms0083:2016-04-01T10:55:50+02:00 ccosvms0083 <auth.info> sshd[1244498]: Accepted passw
ord for vamvakop from 134.158.70.117 port 57745 ssh2
by-host/ccosvms0083:2016-04-01T18:26:52+02:00 ccosvms0083 <auth.info> sshd[1263666]: Accepted passw
ord for vamvakop from 134.158.70.120 port 55938 ssh2
fwernli@ccsvli15/01(0)$
fwernli@ccsvli15/01(0)$
) {$count{$F[7]}++}; END {print Dumper \%count}' by-host/ccosvms*Accepted" && $F[5] eq "password"

[13] 0:ssh*                                                                    ccfawe 03/04
```

```
awk '$5 == "Accepted" && $6 == "password" {count[$8]++} \
     END {for (i in count) {print i,count[i]}}'
```

EXEMPLE: NOMBRE D'ACCÈS APACHE PAR IP

+10

Parsing log files for frequent IP's

So, I hacked this together while undergoing a DDOS attack to pull *naughty* ips out of my logs. Anyone have any improvements or other suggestions to make it better?

Here's the general idea:

1. pull ip's only out of log file
2. sort them
3. uniq and count them
4. sort them again

And the string o'pipes:

```
cut --delim " " -f7 /var/log/apache_access | sort | uniq -c | sort -rn > sorted-ips.txt
```

/ text-processing / logs / ip

edited Mar 21 '11 at 0:17 — Gilles — 331k 56 581 1016

asked Aug 12 — gabe. — 4,652

```
awk -F'[ "]+' '$7 == "/" { ipcount[$1]++ }
    END { for (i in ipcount) {
    printf "%15s - %d\n", i, ipcount[i] } }' logfile.log
```

MÉTHODES TRADITIONELLES

ALERTING ET TEMPS-RÉEL

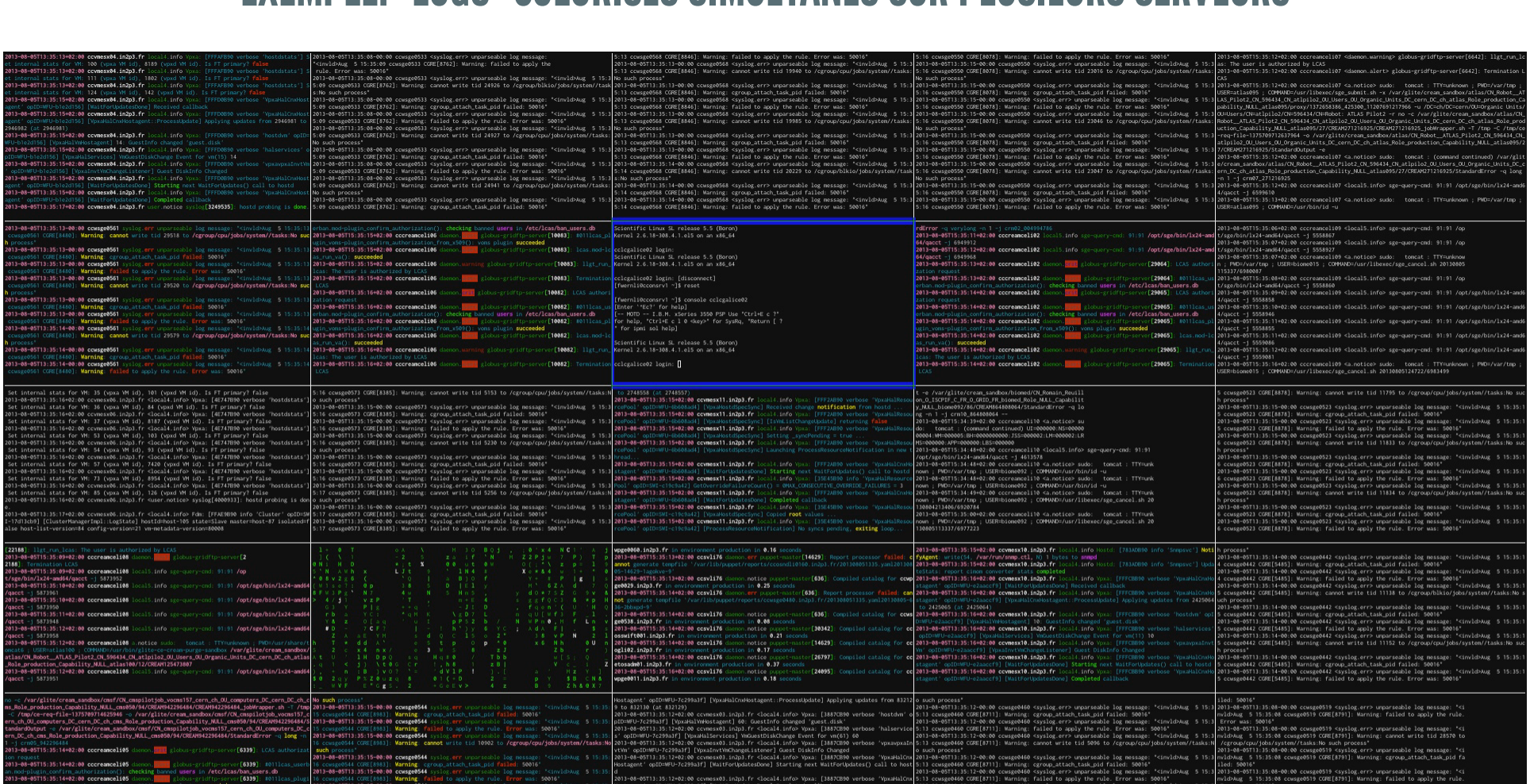
- swatch
- SEC
- `tail -F`
- ccze
- alertes via rsyslog ou syslog-ng via mots-clés

EXEMPLE: ADDRESSE IP VOLÉE

```
/etc/syslog-ng/syslog-ng.conf:

filter f_dupe_addr {
  match("Duplicate address .* detected on interface");
};
log {
  source(s_cisco);
  filter(f_dupe_addr);
  destination(d_email_telecom);
};
```

```
From: root@batman42.in2p3.fr (root)
To: alaaaaarm@cc.in2p3.fr
Subject: alerte syslog-ng

Alerte de syslog-ng :

Nov 24 10:25:10 Lyon-SUPER : 2015 Nov 24 10:25:10 CET: %EIGRP-X-DUP_ADDR
detected on interface VlanWZ
```

EXEMPLE: "LOGS" COLORISÉS SIMULTANÉS SUR PLUSIEURS SERVEURS



MÉTHODES TRADITIONELLES

INCONVÉNIENTS

- Ne passent pas à l'échelle
- Nécessitent des droits privilégiés
- Disparité des outils
- Maintenance nécessaire des "recettes"
- Pas de vue d'ensemble

UNE ÉPINGLE DANS UNE BOTTE DE FOIN

DONNÉES ANNUELLES

- 40 milliards de logs
- 400 milliards de métriques

C'est trop pour `grep`

MÉTADONNÉES

- Savoir ce que l'on cherche
- Corrélations
- Contexte
  1. Technique

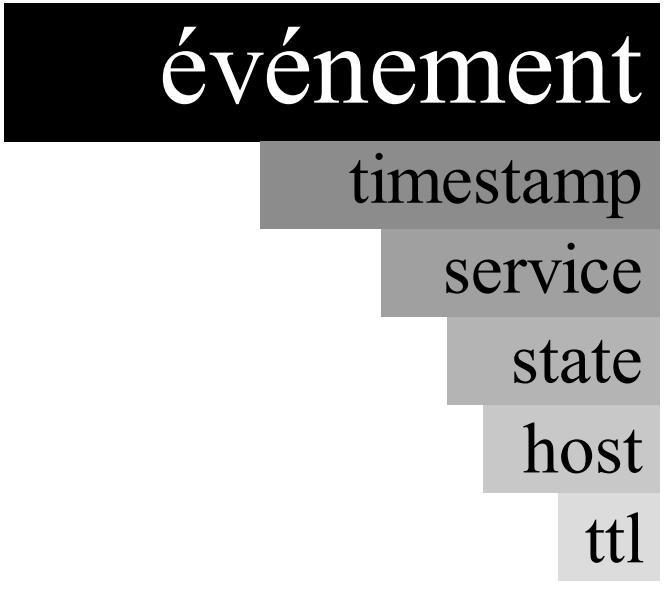
```
device=mmcblkp1
```

  2. Orienté "Site"

```
datacenter=Bron
```

NORMALISATION

- Modèle simple de haut niveau



- Pour *tous* les événements

DONNÉES "SITE"

- localisation (datacenter, rack, *etc.*)
- "facts" techniques (os, matériel, *etc.*)
- rôle des unités (worker, webserver, fileserver, *etc.*)

EXEMPLE: "LOG"


```
{
  "@timestamp": "2016-04-04T15:55:10+02:00"
  "service": "kernel",
  "state": "warning",
  "host": "ccplop1234",
  "ttl": 300
  "message": "end_request: I/O error, dev sda, sector 708561294",
  "syslog": {
    "severity_num": "3",
    "severity": "err",
    "facility": "kern"
  },
  "puppet": {
    "environment": "production_git"
  },
  "pdb": {
    "classifier": {
      "rule_id": "0e701906-cfcb-4868-ad60-d6ae12789fe1",
```

EXEMPLE: MÉTRIQUE

```
{
  "@timestamp": "2016-04-04T01:46:56Z",
  "service": "memory/memory-free/avg/20",
  "state": "ok",
  "host": "plop1234.in2p3.fr",
  "ttl": 20
  "metric": 22206713856,
  "smurf": {
    "rack": "Service 5",
    "status": "Test"
  },
  "type_instance": "free",
  "plugin": "memory",
  "sum": 22206713856,
  "ds_index": "0",
  "tf": "YYYY.MM.dd",
  "count": 1,
  "ds_name": "value",
```

STRUCTURATION

- logs
  - logs structurés à la source (!)
  - sinon syslog-ng/patterndb
  - syslog-ng pour décision state ok/warning/critical
- métriques
  - collectd: métadonnées techniques et seuils (state ok/warning/critical)
  - collectd-write_riemann: transfert des événements structurés vers riemann

EXEMPLE: LOG

- syslog-ng/patterndb "parse" et analyse selon des règles

```
2014-05-30T14:34:53 node01 ata2.00: exception \
Emask 0x0 SAct 0xffff SErr 0x0 action 0x0
```

↓ patterndb ↓

```
{
  "timestamp": "2014-05-30T14:34:53",
  "host":      "node01",
  "service":   "kernel-drivers/ata-2.00",
  "state":     "warning",
  "ttl":       300
  "kernel": {
    "type":   "exception",
    "emask":  "0x0",
    "sact":   "0xffff",
    "serr":   "0x0",
    "action": "0x0"
  }
}
```

EXEMPLE: MÉTRIQUE

- collectd mesure et seuille selon la consigne

```
df -b /images
```

↓ collectd ↓


```
{
  "time": 1427120155,
  "host": "ccetosadm01.in2p3.fr",
  "state": "warning",
  "service": "df-images/percent_bytes-free",
  "ttl": 120,
  "metric": 17.42259,
  "attributes": {
    "cmdb.role": "openstack compute",
    "facter.productname": "Poweredge R420",
    "collectd.plugin": "df",
    "collectd.plugin_instance": "images",
    "collectd.type": "percent_bytes",
    "collectd.type_instance": "free"
  },
  "tags": ["collectd"]
}
```

MÉTADONNÉES: AVANTAGES

- indexation
- performance
- corrélations
- requêtes
- archivage ciblé
  - *p.ex.* selon "severity"

ARCHIVAGE: LOGS

- Légalement on doit garder un an certains événements
- 1 an de logs au CCIN2P3 c'est possible, mais cher

---

il faut donc cibler les purges ou faire des aggrégations p.ex. générer des statistiques

logs → métriques

ARCHIVAGE: MÉTRIQUES

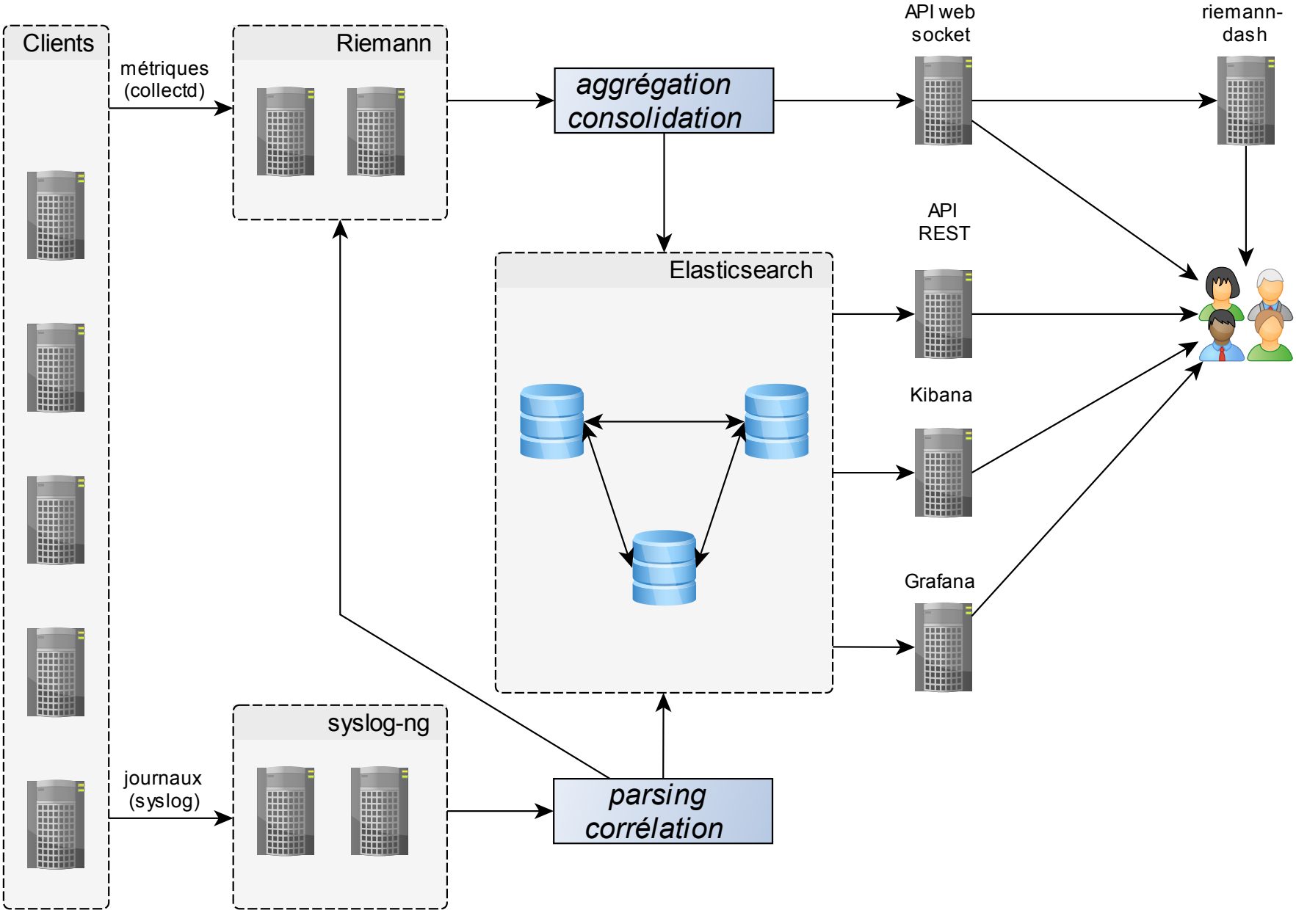
- 1 an de métriques au CCIN2P3 c'est pas possible
  - stockage trop important
  - requêtes trop lentes

---

on fait donc des aggrégations en-ligne (p.ex. min/max/avg)

métriques moins précises plus on remonte le temps

ARCHITECTURE ACTUELLE


Clients
métriques
(collectd)
Riemann
aggrégation
consolidation
API web
socket
riemann-
dash
API
REST
Elasticsearch
Kibana
Grafana
journaux
(syslog)
syslog-ng
parsing
corrélation

PILE LOGICIELLE

"BACK-ENDS"

- syslog-ng
- collectd
- riemann
- elasticsearch

SYSLOG-NG

- transport "logs"
- structuration
- analyse ("pattern matching")
- intégration python, rust, perl
- corrélation
- alerting
- routage

COLLECTD

- collecte métriques
- définition de seuils
- transport vers riemann
- modulaire
- "plugins" C, perl, python, java

RIEMANN

- gestion en-ligne "logs" et "métriques"
- aggrégations, consolidations
- seuillage
- alerting
- routage
- abonnements "websockets"
- tout est possible via clojure
- archivage via samplerr

ELASTICSEARCH

- "You know, for search!"
- Moteur de recherche
- Indexation des documents
- distribué
- API très riche
- aggrégations multiples
- souple
- GUI puissant

"FRONT-ENDS"

- Kibana
- Grafana
- Riemann-dash
- CLIs REST et WS

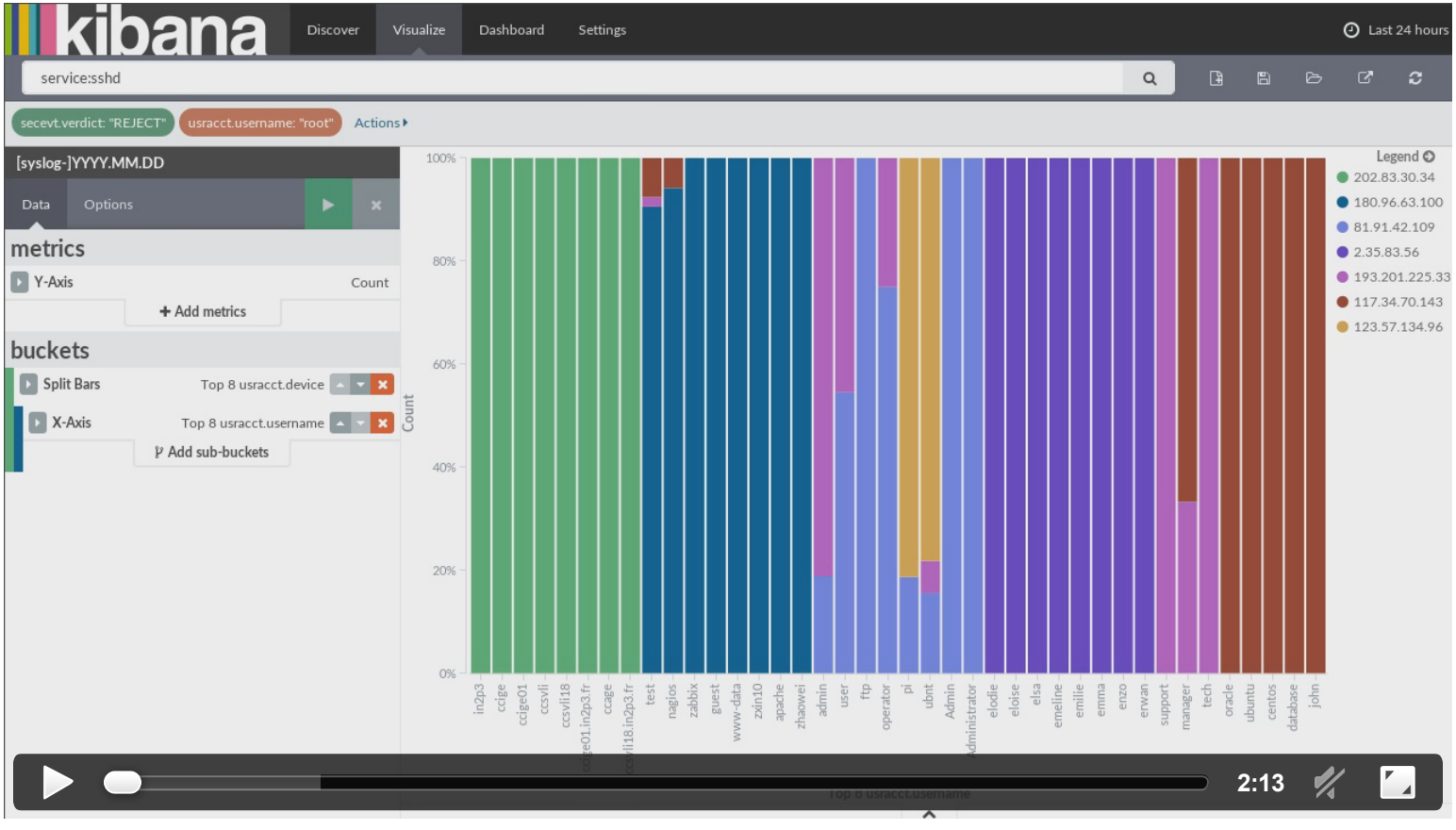
MÉTHODES MODERNES

ANALYSE POST-INCIDENT

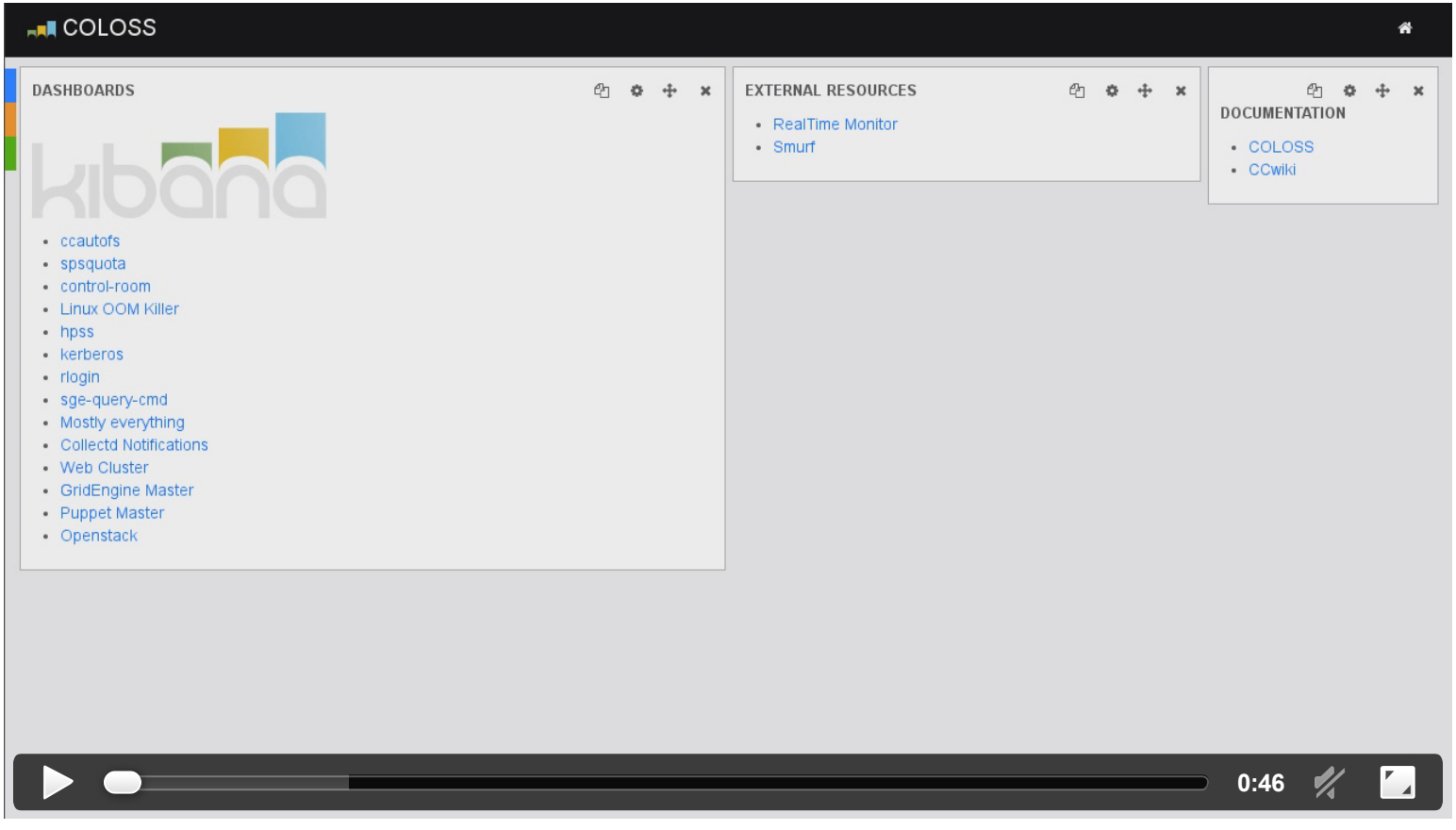
INDEXATION DES DONNÉES

- Requêtes via
  - API
  - GUI
- Système distribué
  - passage à l'échelle
  - facilité d'administration
- Authentification/autorisations
- Structuration des données

EXEMPLE: CONNEXIONS SSH PAR UTILISATEUR


kibana
Discover
Visualize
Dashboard
Settings
Last 24 hours
service:sshd
secevt.verdict: "REJECT"
usracct.username: "root"
Actions
[syslog-]YYYY.MM.DD
Data
Options
metrics
Y-Axis
Count
+ Add metrics
buckets
Split Bars
Top 8 usracct.device
X-Axis
Top 8 usracct.username
Add sub-buckets
100%
80%
60%
40%
20%
0%
Count
Legend
202.83.30.34
180.96.63.100
81.91.42.109
2.35.83.56
193.201.225.33
117.34.70.143
123.57.134.96
in2p3
ccige
ccige01
ccsvli
ccsvli18
ccige01.in2p3.fr
ccage
ccsvli18.in2p3.fr
test
nagios
zabbix
guest
www-data
zxin10
apache
zhaowei
admin
user
ftp
operator
pi
ubnt
Admin
Administrator
elodie
eloise
elsa
emeline
emilie
emma
enzo
erwan
support
manager
tech
oracle
ubuntu
centos
database
john
Top 8 usracct.username
2:13

EXEMPLE: ACCÈS APACHE


COLOSS
DASHBOARDS
kibana
ccautofs
spsquota
control-room
Linux OOM Killer
hpss
kerberos
rlogin
sge-query-cmd
Mostly everything
Collectd Notifications
Web Cluster
GridEngine Master
Puppet Master
Openstack
EXTERNAL RESOURCES
RealTime Monitor
Smurf
DOCUMENTATION
COLOSS
CCwiki
0:46

MÉTHODES MODERNES

ANALYSE "TEMPS-RÉEL"

ALERTING ET "TEMPS-RÉEL"

- Requêtes via
  - API
  - GUI
- Structuration des données
  - même structure que pour l'indexation

EXEMPLES: CLI

```
fwernli@ccosvms0003/~(0)$ loggmann -h
loggmann [-cdfHhqrSs] [long options...] [host] [service]
        -s --server         Riemann server to subscribe to
        -q --query          Riemann Query (see
                            https://github.com/aphyr/riemann/blob/master/test/riemann/query_test.clj)
        -H --host           Host to query (will be ANDed to query)
        -S --service        Service to query (will be ANDed to query)
        -h --help           Print help
        -d --debug          Toggle debugging
        -c --configfile     Path to configfile
        -f --format         logformat. Defaults to "time host service
                            description"
        -r --rawformat      log raw json data. supersedes --format
fwernli@ccosvms0003/~(0)$
```

```
_______
################################################################################

                  Bienvenue au centre de calcul de l'IN2P3
                    en partenariat avec le CEA/DSM/IRFU

 --------------------------------------------------------------------------------

                  Welcome to the IN2P3 computing center
                    in partnership with CEA/DSM/IRFU

we 05/04h*                                                                  ccfa
```

• • •

EXEMPLES: WEBGUI

RÉFÉRENCES

www.syslog-ng.org	cern.ch
LHC collision details	LHC closer look
riemann.io	riemann.io/dashboard
en.wikipedia.org/wiki/WebSocket	collectd.org
elastic.co/products/elasticsearch	elastic.co/products/kibana
puppetlabs.com	ccin2p3/patterndb
samplerr	

RÉFÉRENCES

- Processing log messages with Python in syslog-ng
- A How to Guide on Modern Monitoring and Alerting
- Transferring Conserver Logs to Elasticsearch, Linux Journal January 2016