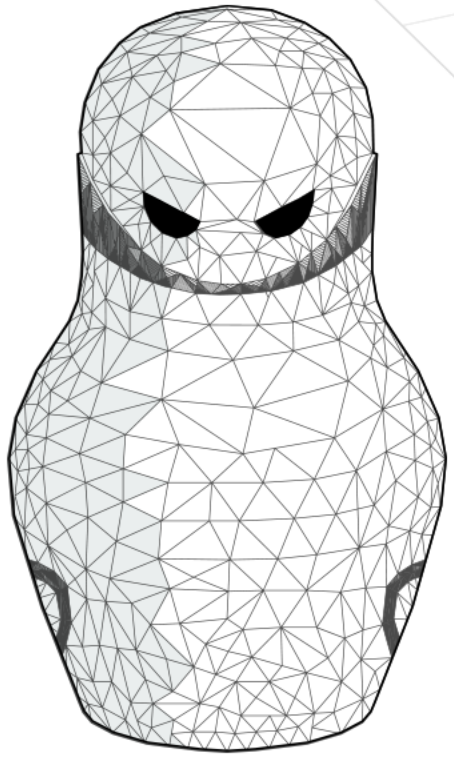


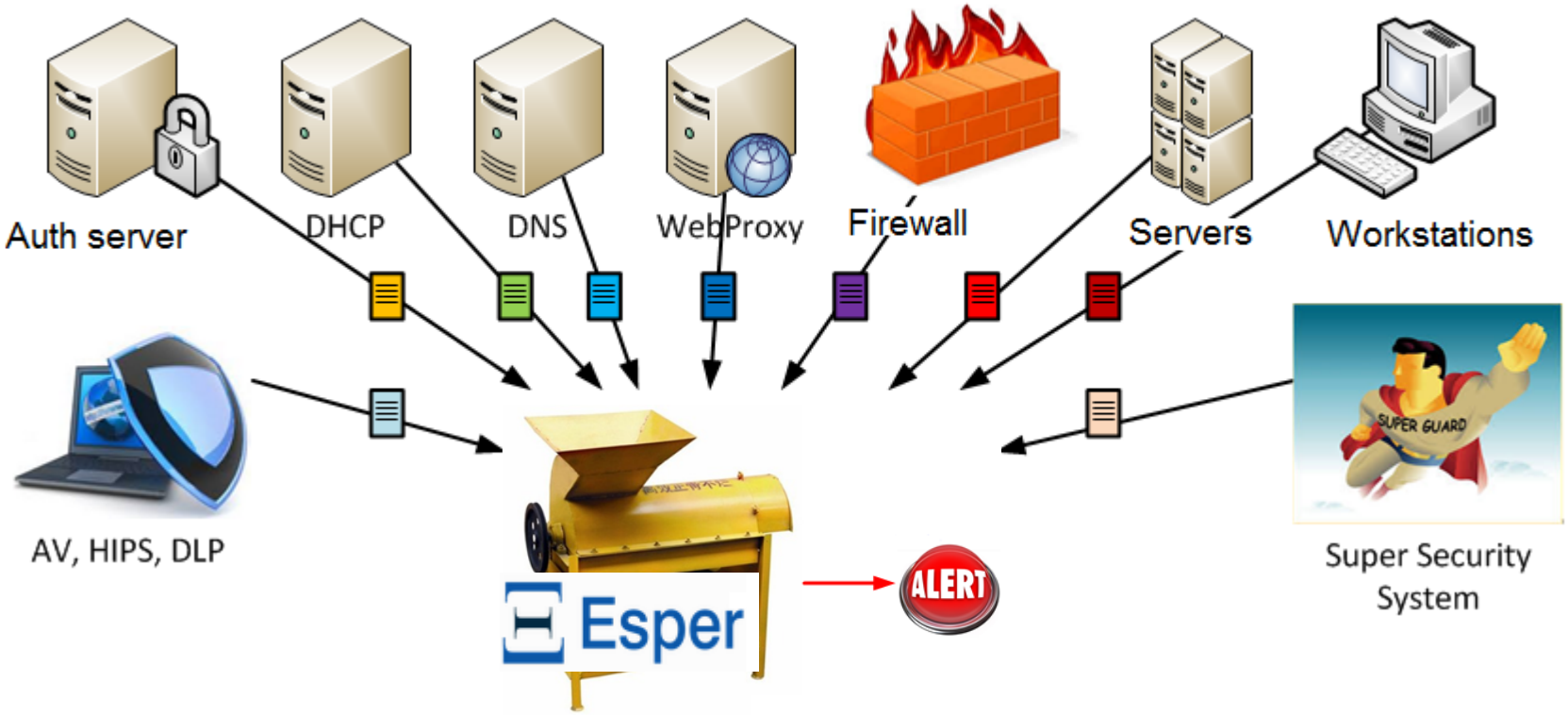
ZERO NIGHTS 2015

Security events correlation with Esper

Nikolay Klendar
bsploit @ gmail.com

INTRO

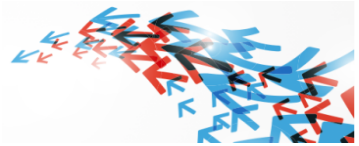
Complex Event Processing (correlation) * - is event processing that combines data from multiple sources to infer events or patterns that suggest more complicated circumstances.



*Wikipedia

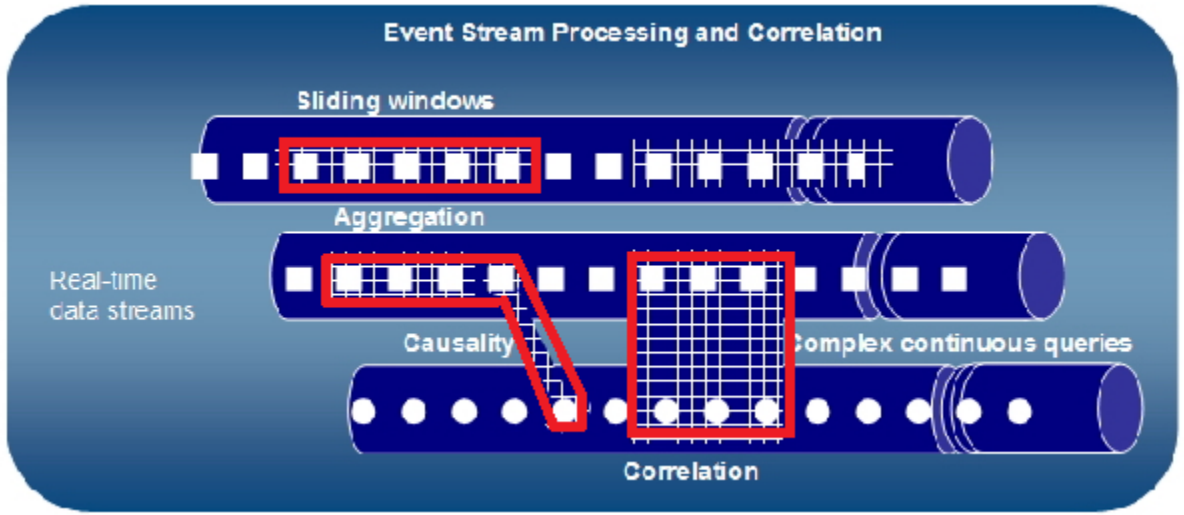
Esper

Library used for development
Java, .NET



Processes event STREAMS of predefined types.
Esper does not parse events!

Processing rules (correlation rules) are defined with Event Processing Language (EPL) similar to SQL



Network scan detection

Annotation

All dst_ip within 30 sec

Allowed monitoring systmes

Type event:

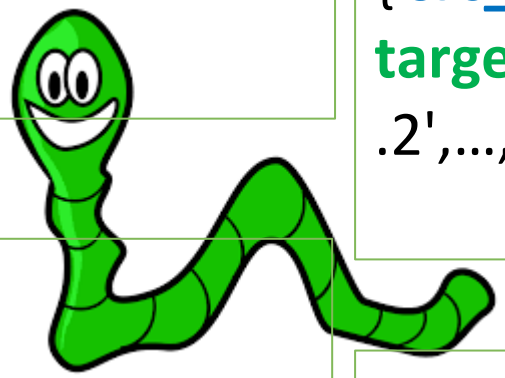
timestamp:string
type: string
src_ip: string
dst_ip: string
src_port:int
dst_port: int
bytes_sent: int
bytes_recieved: int
login: string

```
@Name('Scan')
SELECT src_ip,window(dst_ip)
FROM event (type='firewall'
        AND src_ip NOT IN ('10.0.0.1','10.0.0.2')
).win:time(30 sec)   /*sliding time window*/
GROUP BY src_ip
HAVING count(distinct dst_ip) > 50
output first every 1 hour   /*1 event per hour*/
```

Worm spreading detection

```
INSERT INTO scanning
SELECT src_ip,window(dst_ip) targets
FROM event().win:time(10
min).std:unique(dst_ip)
GROUP BY src_ip
HAVING count(distinct dst_ip)>50;
```

```
{ src_ip='10.0.0.1',
targets=['192.168.0.1',
'192.168.0.2',...,'192.168.0.254']}
{ src_ip='192.168.0.2',
targets=['192.167.0.1','192.167.0
.2',...,'192.167.0.254']}
```



```
@Name('warm_spreading')
SELECT a.src_ip,b.src_ip,b.targets
FROM pattern[
 every a=scanning -> b=scanning (
          b.src_ip!=a.src_ip AND
          Arrays.asList(a.targets).contains(b.src_ip)
)  WHERE timer:within(1 min) ];
```

```
{a.src_ip='10.0.0.1',
b.src_ip='192.168.0.2',
b.targets=[' 192.167.0.2 ',...,
'192.167.0.2 ',' 192.167.0.2 ']}
```

Money laundering detection

Total money transferred to card

```
@Name('obnal')
SELECT a.transaction,a.clientid,a.amount income, c.sumOf(i=>i.amount)
+b.amount total
FROM PATTERN[
   EVERY a=event(transaction like 'card_income') ->
          b=event(b.clientid=a.clientid AND transaction = 'card_outcome')
WHERE timer:within(3 hour) ->
  ( [3: ] c=event(c.clientid=a.clientid AND transaction = 'card_outcome' )
until timer:interval(20 min) )
]
```

Total outcome

Join & enrichment

```
CREATE WINDOW
LoginsIP.std:unique(ip) as (ip string, login string, last_seen string);

INSRT INTO LoginsIP
SELECT src_ip as ip, login.toLowerCase() as login, timestamp as last_seen
FROM Event(
type='windows' AND eventid='4624' AND src_ip IS NOT NULL
AND login IS NOT NULL AND login!='ANONYMOUS LOGON'
AND login NOT LIKE '%$' );
```

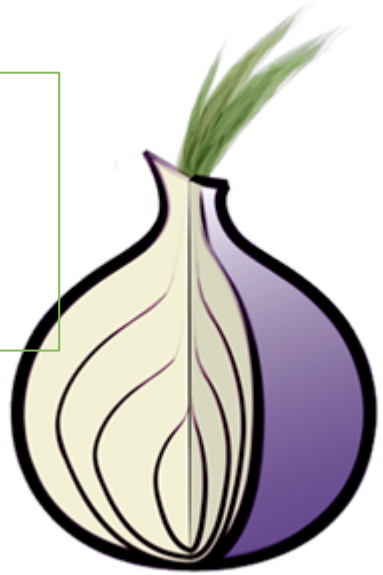
```
SELECT S.src_ip, S.targets, L.login,L.last_seen
FROM scanning.std:lastevent() as S
LEFT OUTER JOIN LoginsIP L on L.ip = S.src_ip
GROUP BY S.src_ip
output first every 1 hour;
```

```
{
    S.src_ip='10.0.0.1',
    L.login='ivanov',
    L.last_seen='17.11.2015 12:00:00'
    S.targets=[' 192.167.0.2 ',...,
'192.167.0.2 ',' 192.167.0.2 ']
}
```

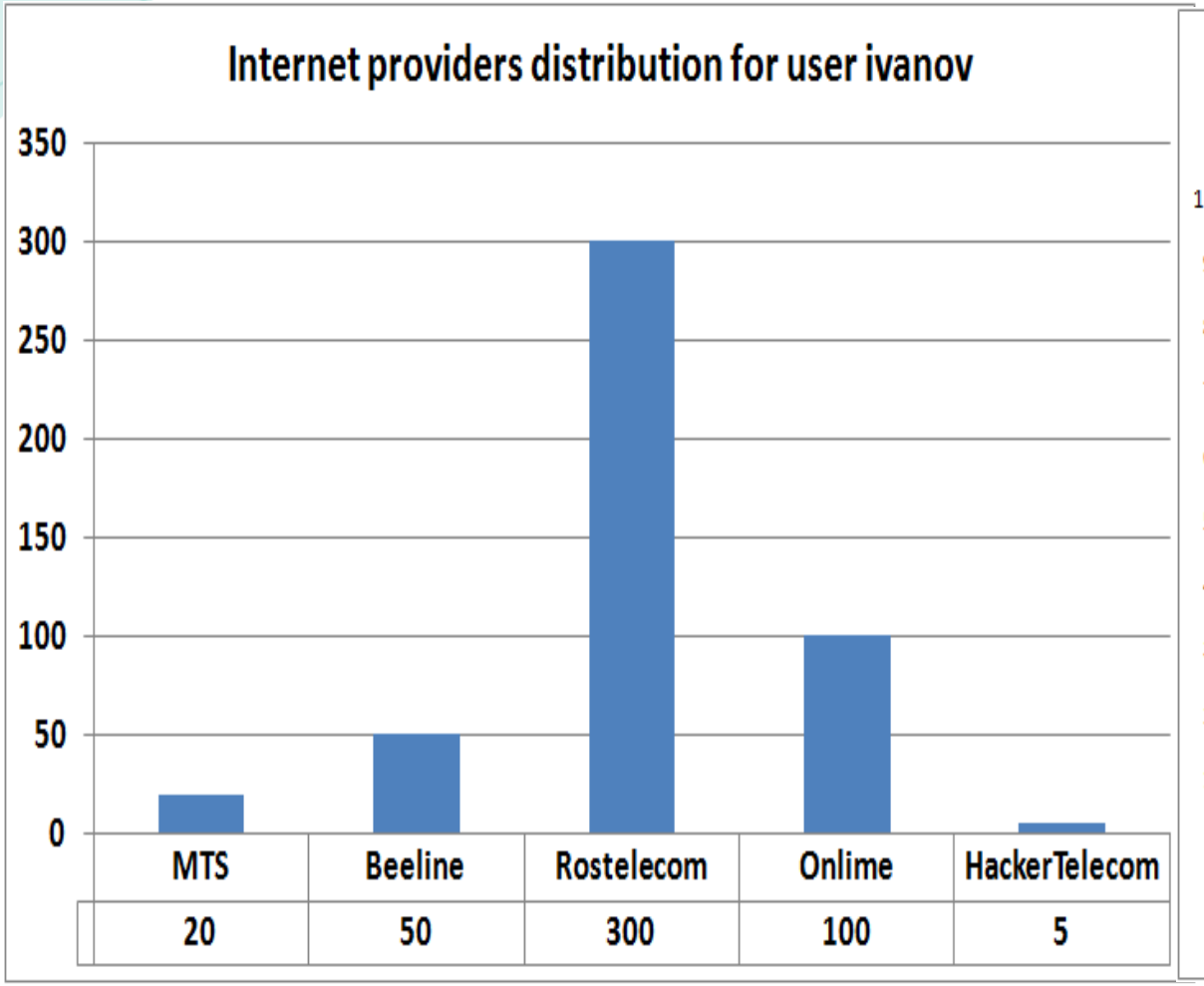
Integration with external sources

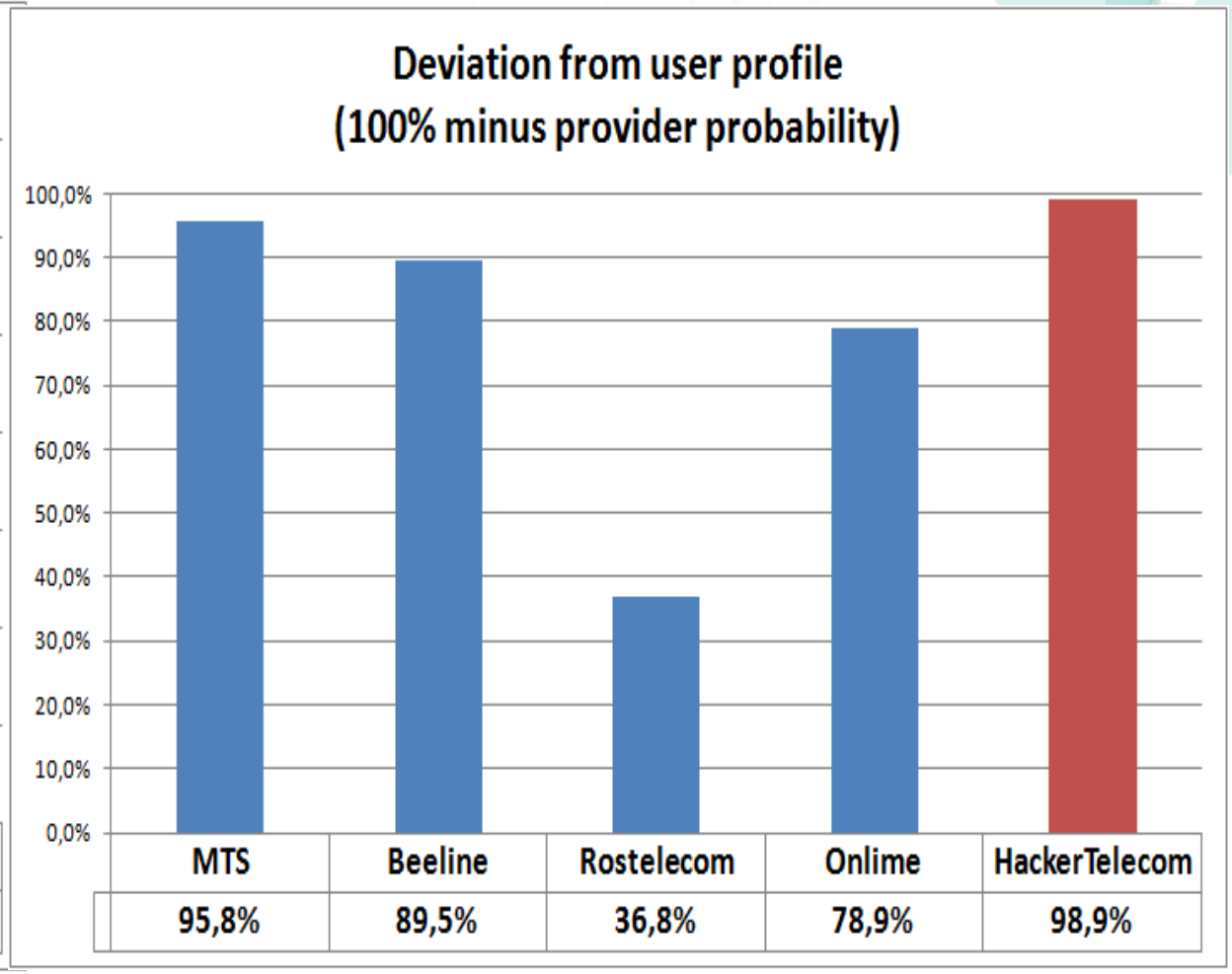
```
<database-reference name="mysql">
    <drivermanager-connection class-name="com.mysql.jdbc.Driver"
        url="jdbc:mysql://localhost/testDB">
    <connection-arg name="user" value ="user"/>
    <connection-arg name="password" value ="password"/>
    </drivermanager-connection>
</database-reference>
```

```
SELECT src_ip from event(type='firewall') as fw,
SQL:mysql ['select tornode_ip from tor_nodes'] as tor
where fw.src_ip=tor.tornode_ip
```



Users profiling





Building user profile

```
create window
loginProfileASN.win:keepall()
(login string,param string,value
string,v_count long)
```

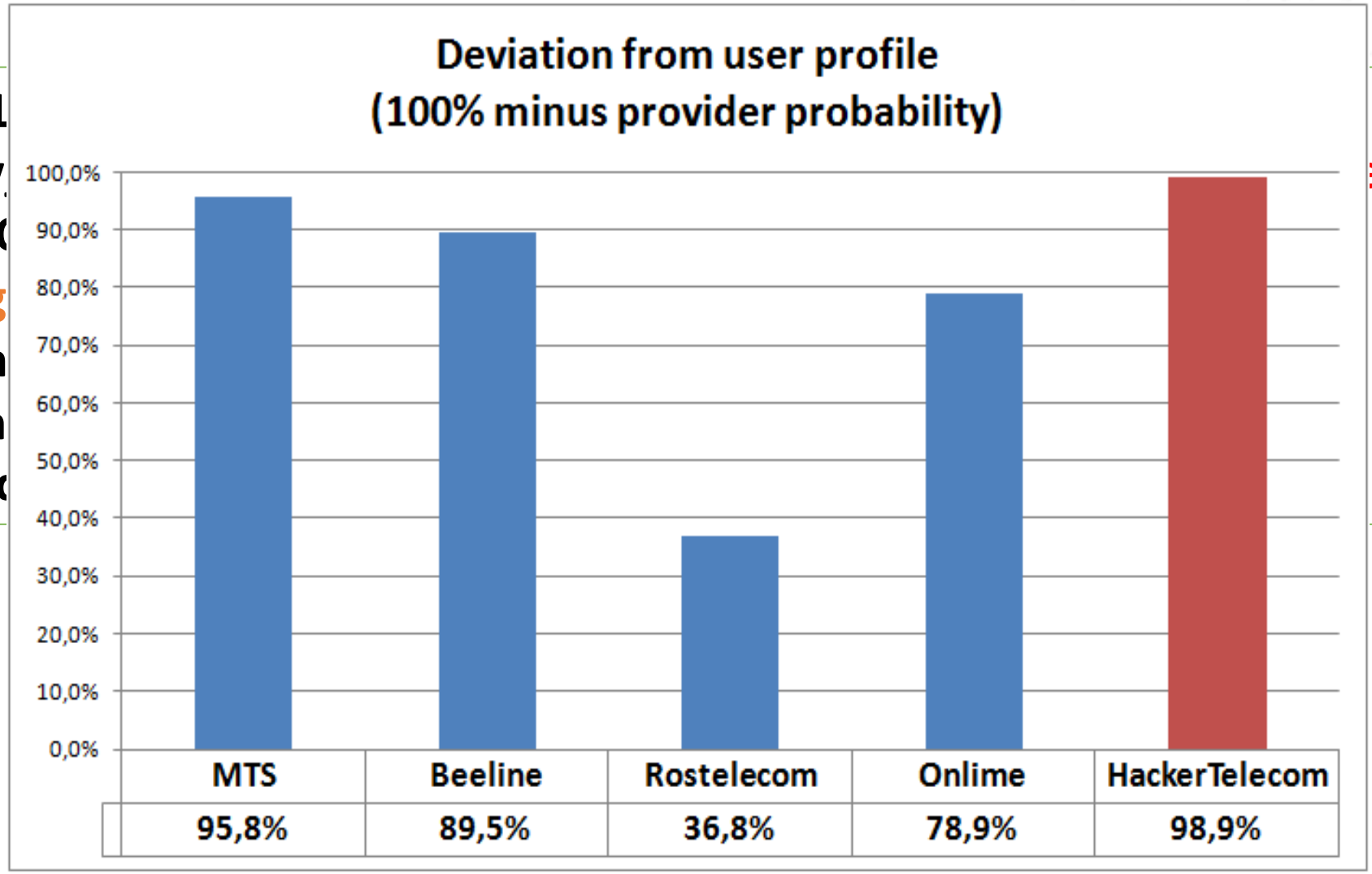
```
create window
loginProfileTotal.win:keepall()
(login string,param string,total long)
```

```
ON EVENT() e
MERGE loginProfileASN p
where p.login=e.login and
p.value=(e.geoip('number')).toString()
when not matched
 then insert select login,'ASN' param,
geoip('number') value,1L v_count
when matched
then update set p.v_count = p.v_count+1
```

```
ON EVENT() e
MERGE loginProfileTotal p
where p.login=e.login
when not matched
 then insert select login,'ASN' param, 1L
total
when matched
then update set p.total = p.total+1
```

Deviation from profile

SEL
v.v
FR(
log
wh
an
and



CorReactive and integration with ELK

Logstash config

```
output {
   redis {
      host => "127.0.0.1"
      db => 0
      data_type => "list "
      batch => true
      batch_events=>500
      key => "events”
      codec => json
   }
}
```

CorReactive config

Collect events

```
"inputs":[
   {
      "type": "redis",
      "config":{
         "host": "localhost",
         "port": 6379,
         "db": 0,
         "queue":"events",
         "batch_count":500,
         "reconnect_timeout":60
      }
   }
]
```

correlation engine)

CorReactive config

Return alerts

```
"outputs":[
   {
      "type":"redis",
      "id":1,
      "config":{
         "host": "localhost",
         "queue":"alerts",
         "port": 6379,
         "db": 0,
         "reconnect_timeout":60,
         "batch_count" :1
      }
   }
]
```

CorReactive configuration steps

1. conf/types:
   Extend base event type “event”, add new fields

2. conf/modules:
   Add new EPL modules (correlation rules)
   If one module depends on another use special directive:
   uses dependent_module; http://goo.gl/9pvlIj

3. Configure inputs and outputs

CorReactive special annotations

Alert generation to output channel
@Alert(name='newalert',outID=1)

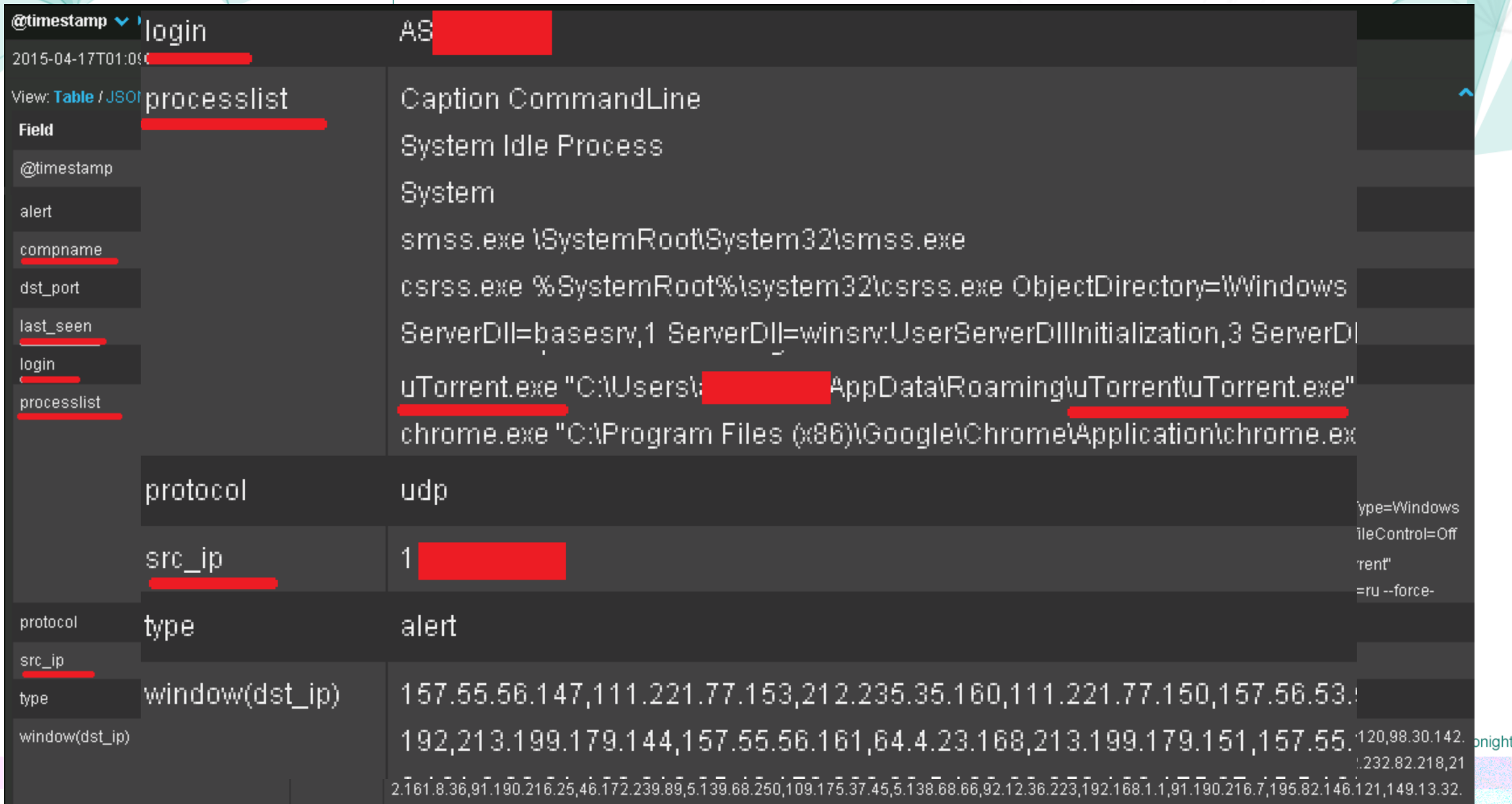
Save data from named window to disk every 5 minutes.
Saved data is automatically restored to named window during loading stage
@Persist

Named window data reloading every 5 minutes from csv file
located in var/winload
@Load(file="data.csv",format="csv",delim="; ")

Dynamically alert enrichment with data from external command output
or on demand query. Enrichment of enrichment is supported.

```
@Enrich(dst="eLogin",type="window", param="select src_ip from loginip where
    login='%{login}'")
@Enrich(dst="nsresult",type="cmd",param="nslookup %{eLogin}")
```

Alert example in Kibana



REST API

Send event in JSON format
POST /api/events

View all registered modules
GET /api/modules/registered

View all registered Esper statements or queries
GET api/modules/statements

Reload data in named window
POST /api/window/reload/{moduleName}/{winName}

Deploy all modules
POST api/modules/deploy

Module deletion
DELETE /api/modules

Module syntax validation
POST api/modules/validate

Do on demand query
POST /api/query

Links

Esper docs
http://www.espertech.com/esper/documentation.php

Solution patterns with description
http://www.espertech.com/esper/solution_patterns.php

EPL editor and debugger
http://esper-epl-tryout.appspot.com/epltryout/mainform.html

CorReactive engine (special for ZeroNights 2015)
http://correactive.sourceforge.net/

Thank you!

Questions?