Un langage pour la configuration de DISCUS, une architecture distribuée de solutions de sécurité

Damien Riquet Gilles Grimaud Michaël Hauspie

Team 2xS
Université Lille 1, France

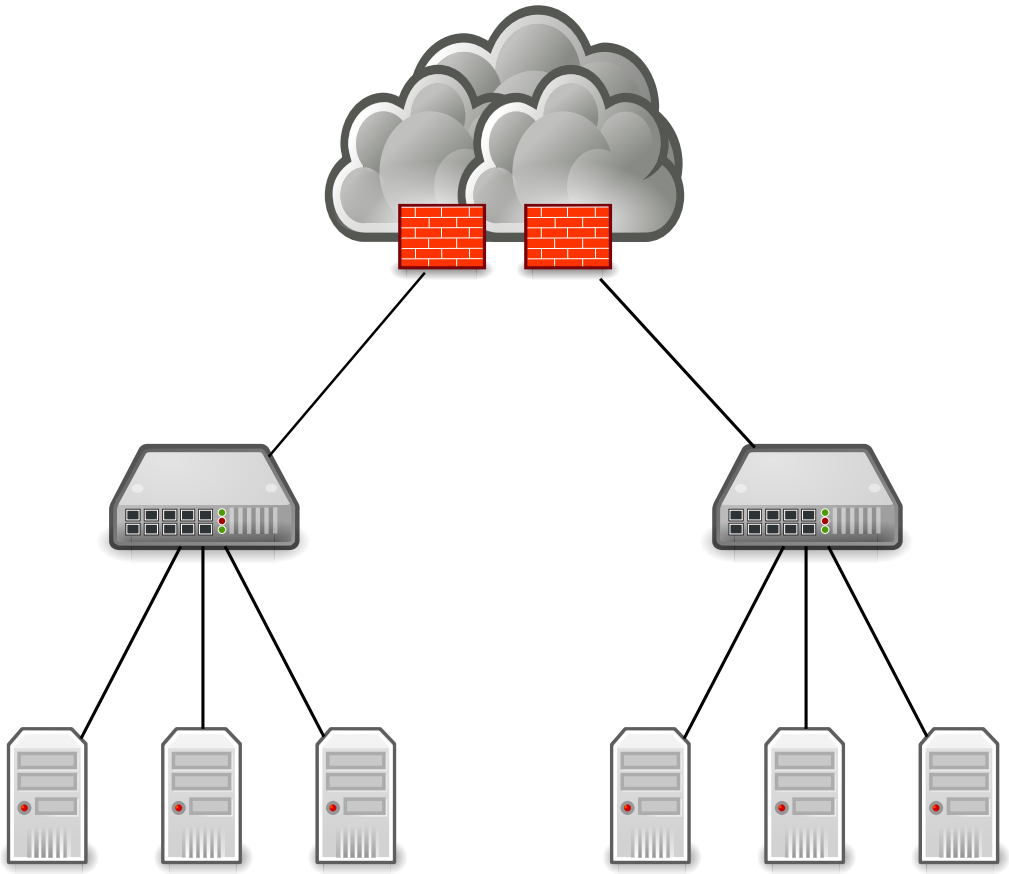
24 Avril 2014

Context

- Cloud Computing: a popular model to process large data set
- Several layers according to the needs of customers
- Store confidential data
- Growing concern about its security

Attacks on the cloud

- Distributed attacks to evade security solutions
- Weaknesses of cloud structure



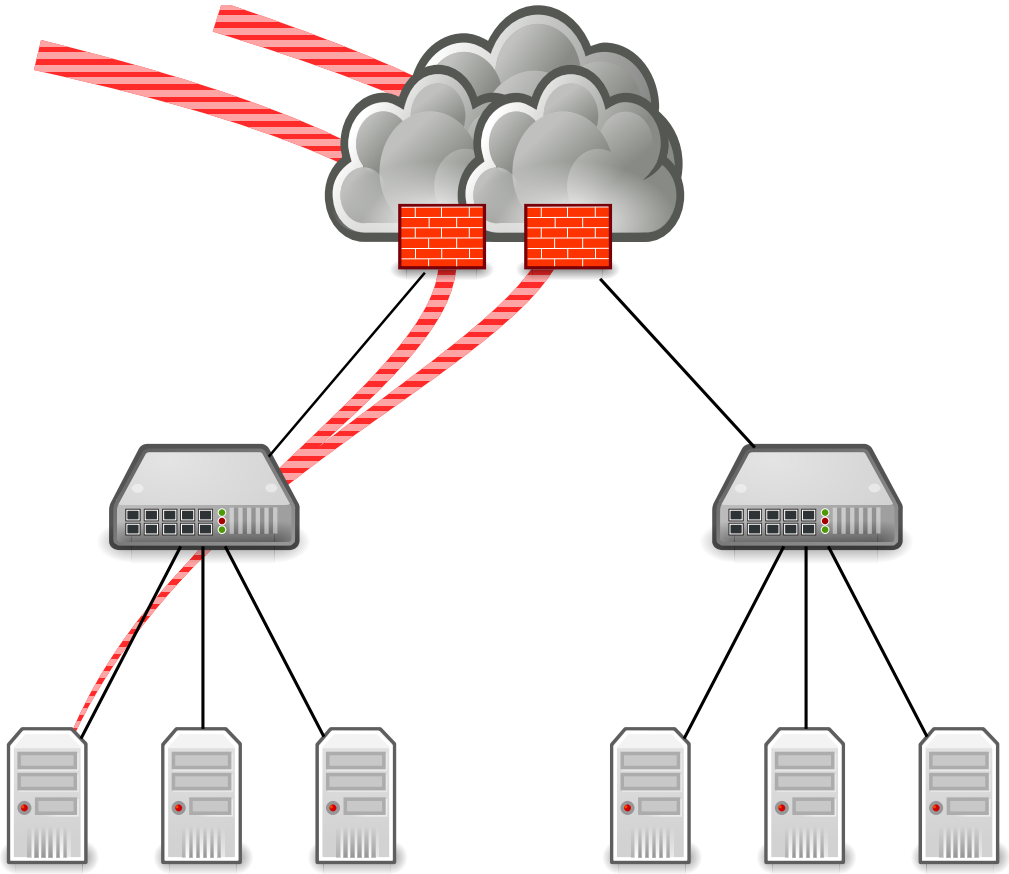
Cloud security - Security solutions commonly used

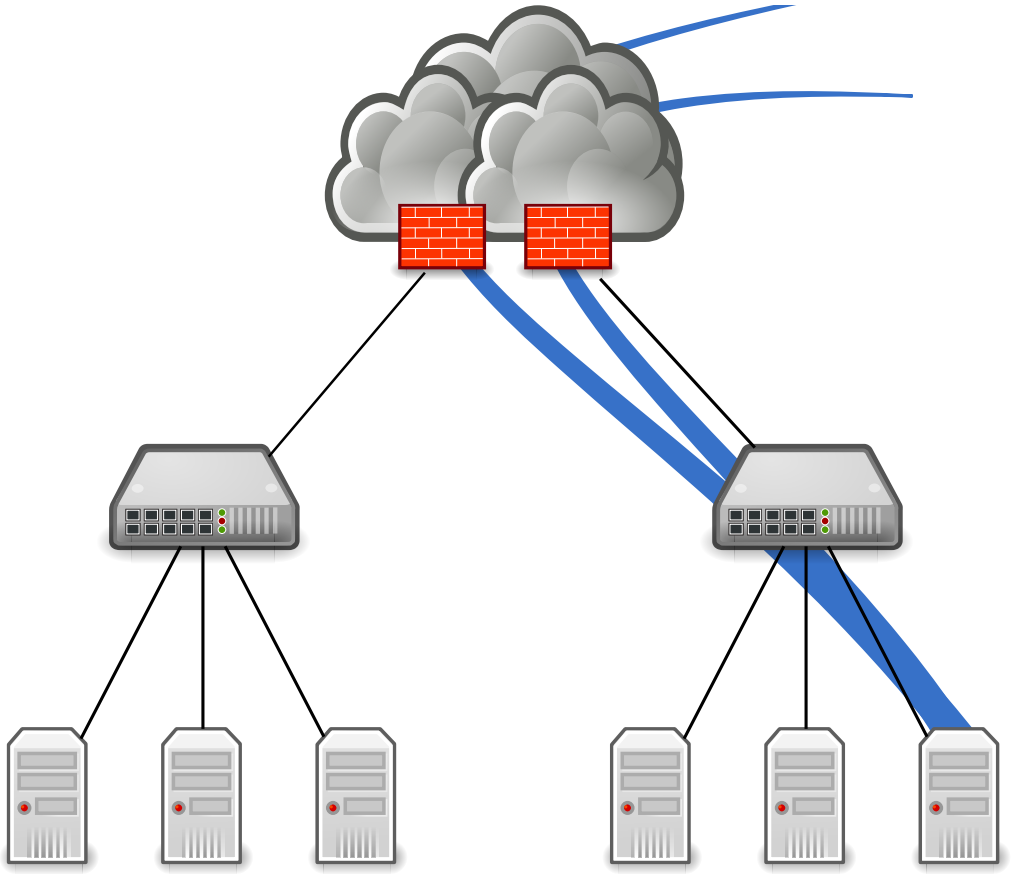
Firewalls

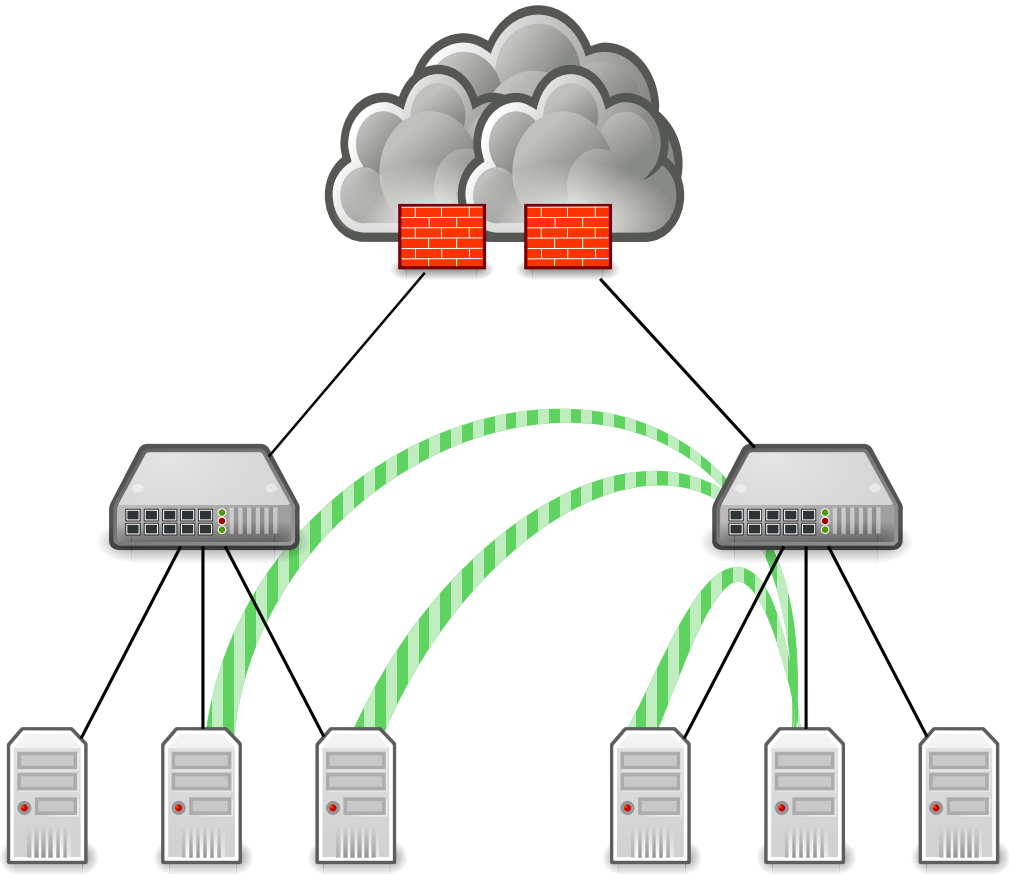
- At the border of the network
- Analyze traffic between two networks
- Security policies

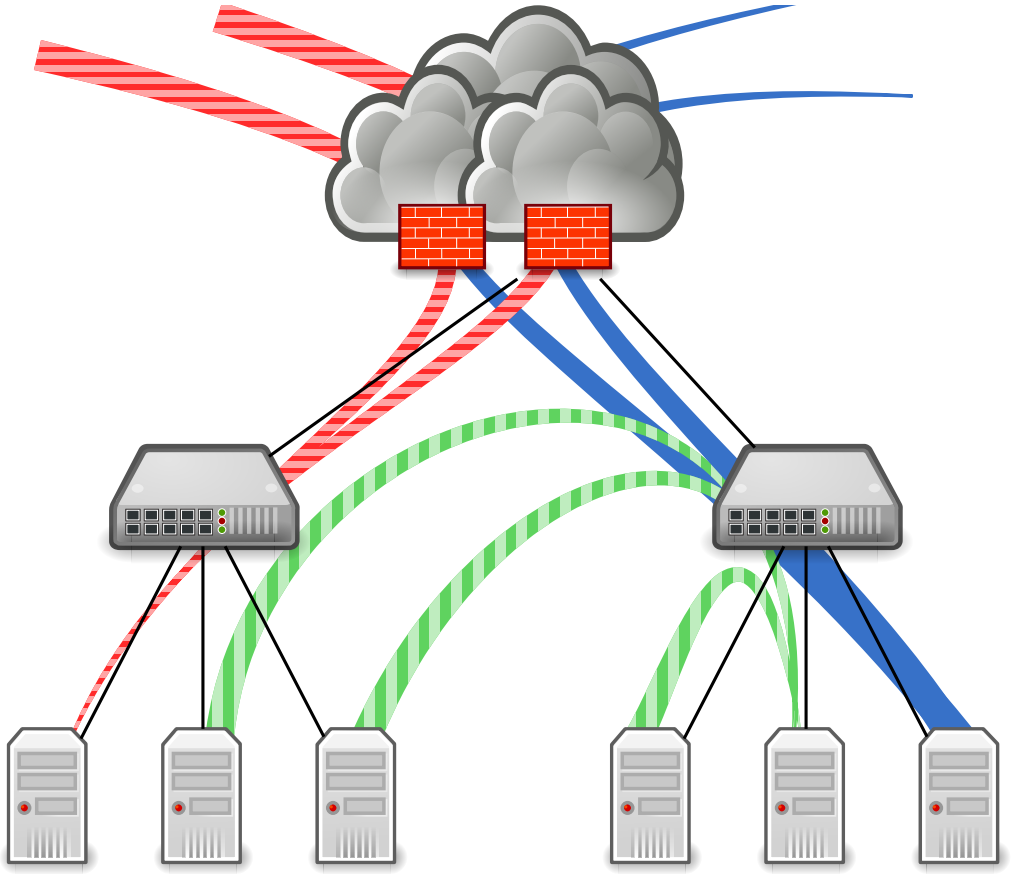
Intrusion Detection System (IDS)

- Network or Host based
- Passive device: raise alarms
- Pattern-matching, analyze traffic









Large scale attacks can easily evade existing security solutions [a] [b]

[a] *Large-scale coordinated attacks : Impact on the cloud security.* IMIS 2012.

[b] *Étude de l'impact des attaques distribuées et multi-chemins sur les solutions de sécurité réseaux.* MajecSTIC 2012.

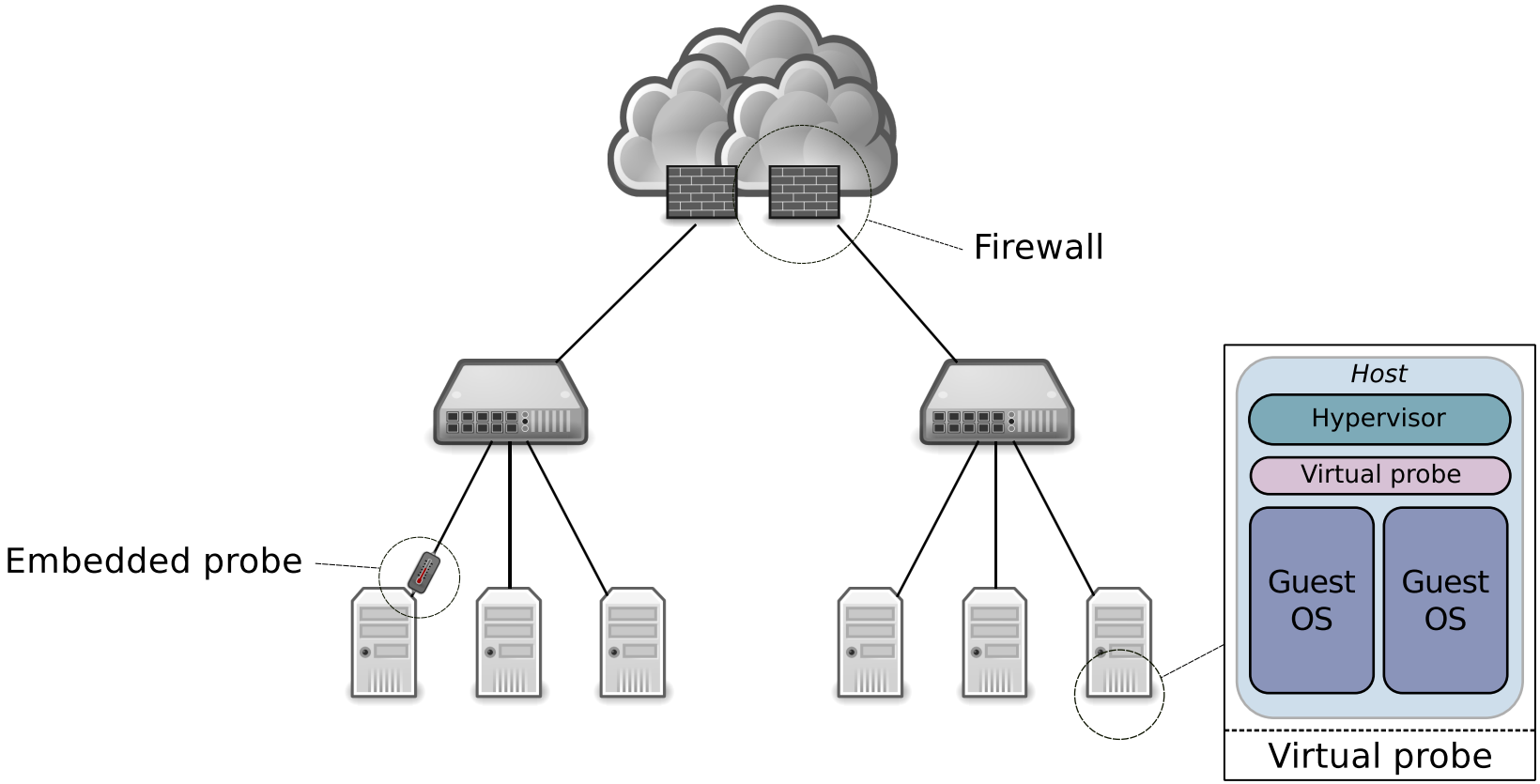
Outline

Outline

Dicus: A massive distributed network of security solutions

- Security probes as close to the hosts as possible
- Probes could either be hardware or software
- Probes collaborate to detect malicious behaviour

Structure overview



But ...

- Difficult to write software for heterogeneous targets
- Collaboration implies sophisticated distributed algorithms
- Hard to deploy (done by the network administrator)

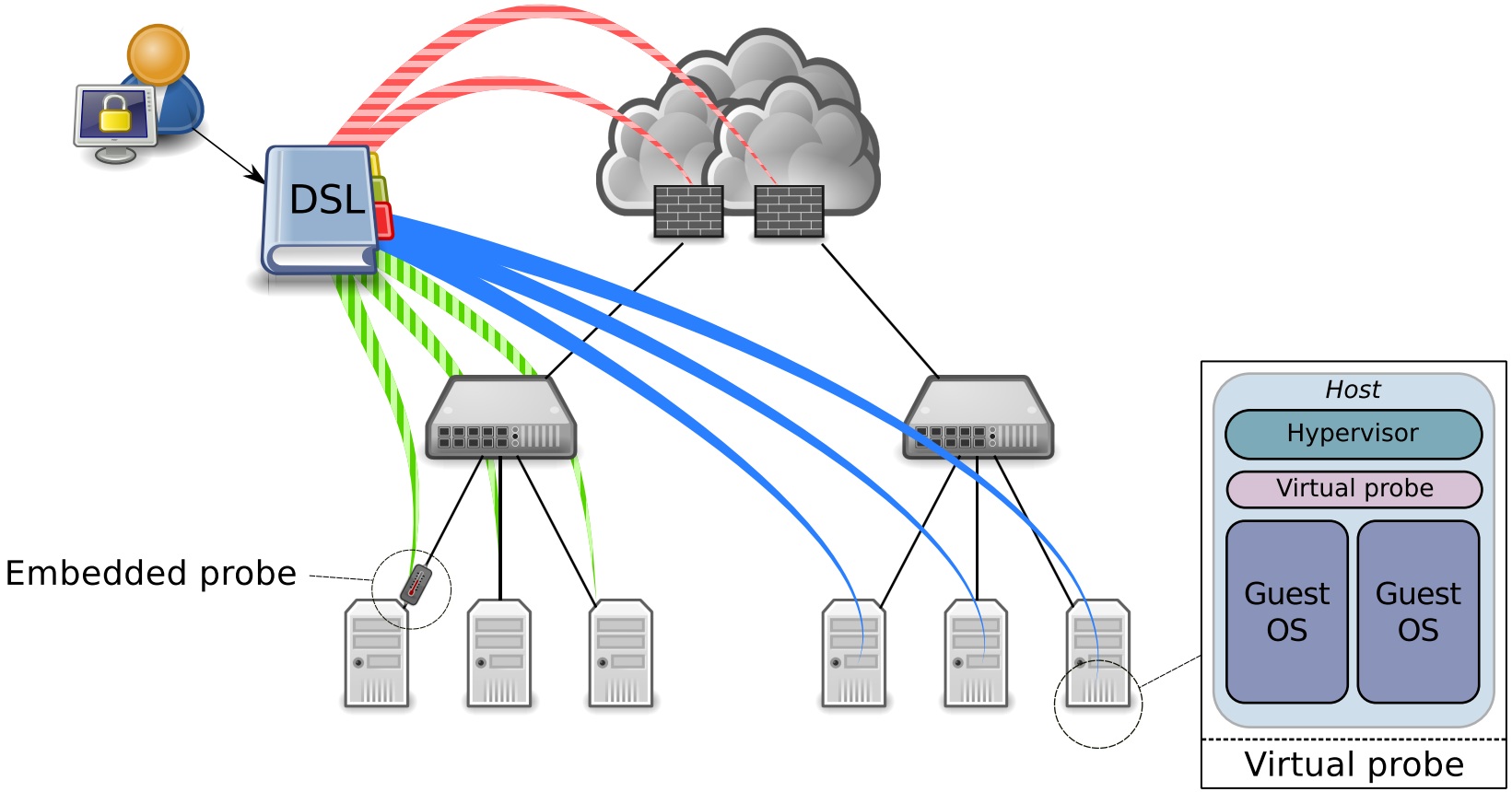
One solution: Discus Script

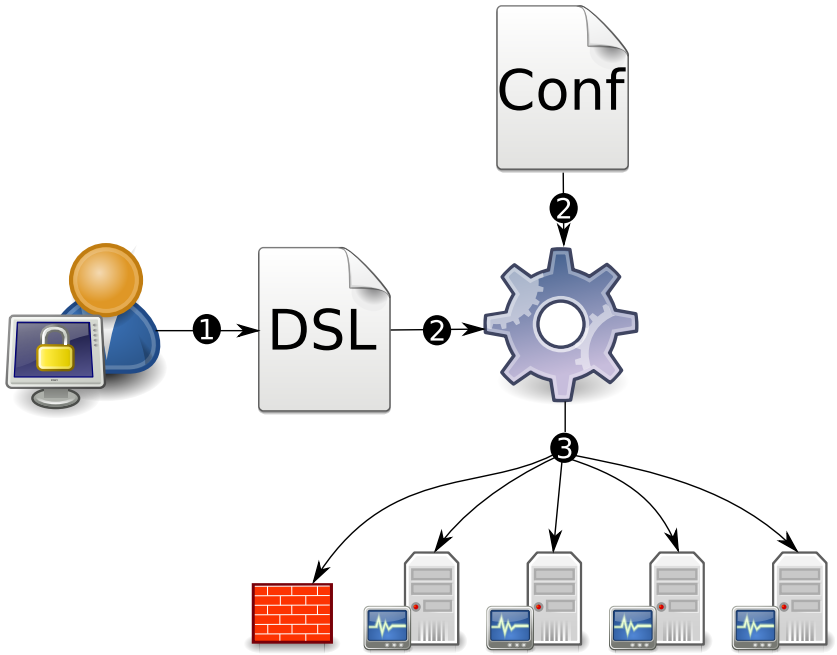
- Must focus on what to detect rather than how to implement it
- Domain Specific Language used to describe security rules
- Discus toolchain deals with implementation details

Outline

Discus Script toolchain

- Provides an abstract layer of heterogeneous security solutions ;
- Frees the developer from focusing on distribution issues ;
- Solution properties:
  - Event-based language,
  - Termination guaranted by construction,
  - Automatic pruning of useless rules,
  - Static typing.


DSL
Embedded probe
Host
Hypervisor
Virtual probe
Guest OS
Guest OS
Virtual probe


Conf
2
1
DSL
2
3

Event-based language

```
on tcp_packet(..., int16 dst_port, ...)
    where dst_port == 80
    raise http_packet(...);
```

Tables

- Distributed database of contextual data ;
- Table entries are aggregates of primary types ;
- Provides a way to collaborate.

```
table tcp_table {
    int32 src,dst;
    int16 p_src,p_dst;
    (...)
};
```

Outline

Basic use case

- Counts number of TCP connections per client
- Alert when a client has opened more than 50 TCP connections

Table declaration

```
table tcp_nb_connection {
    ipaddr client;
    int16 nb_connections;
};
```

New client

```
on tcp_packet(..., ipaddr
    client, ..., int9 flags,
    ...)
    where flags == SYN and
    not exists t in
        tcp_nb_connection with
        t.client == client
    insert into
        tcp_nb_connection {
        client = client;
        nb_connections = 1;
    };
```

Update existing client

```
on tcp_packet(..., ipaddr
    client, ..., int9 flags,
    ...)
  where flags == SYN and
  exists t in
      tcp_nb_connection with
      t.client == client
  update t.nb_connections +=
      1;
  raise check_nb_con(client);
```

Update existing client

```
on check_nb_con(ipaddr client)
    where exists t in tcp_nb_connection with
        t.client == client,
        t.nb_connections > 50
    alert("Client has reach tcp connections limit");
```

Conclusion

- Security solution massively distributed
- Configuration through a dedicated langage
- Abstract layer independent from the target device
- Collaboration through the table mecanisme (distributed database)

Future work

- Deployment of security rules
- Distribution mecanism between the probes

Questions

Damien Riquet - damien.riquet@lifl.fr
Gilles Grimaud - gilles.grimaud@lifl.fr
Michaël Hauspie - michael.hauspie@lifl.fr

http://www.lifl.fr/~riquetd/