Supervision de l'infrastructure et des services à l'Institut Néel.

Stéphanie Zaccaro

Icinga2

- Rappel du contexte : laboratoire / état des besoins / cahier des charges
- Présentation détaillée d'ICINGA 2 en production à Neel
- Evolution envisagée de l'architecture
- Points fort et point faibles de la solution

Contexte

1) Institut Néel : Créé en 2007
Recherche fondamentale en physique de la matière

- 500 agents : (300 permanents, 200 non-permanents)
- manips /serveurs/materiel réseau répartis sur 20 000 m^2 de bâtiments

2) Matériel et services

- Matériel en nombre croissant / nouveaux bâtiments /IoT
  220 switches (310 fin 2018 avec FTTO)
  30 baies de brassage, 14 bornes wifi, 10 VLANs (manips, visiteurs…)

- Une infrastructure virtuelle, des services de gestion, un cloud, un AD, 2 vpn, dhcp, GED plus de 100 sites webs…

Etat des lieux et besoins

La supervision au moment de l'étude (2011):

- munin (quelques serveurs)
- alertes mails : scripts bash / logiciels propriétaires (Dell EM, OMSA)

Besoins

1) Centralisation du contrôle de l'infrastructure

2) Rationalisation et convergence des outils :

- Permettre une meilleure réactivité de l'équipe informatique en cas de problèmes sur l'infra et le parc
- Permettre à l'ensemble de l'équipe d'être proactif et d'avoir un système d'alerte fiable pour résoudre une panne en cas d'absence / congé.

3) Amélioration de la disponibilité des services :

Détecter les failles récurrentes et améliorer le système en conséquence (prévention).

Cahier des charges

1] Critères d'ordre général

- Simplicité d'installation, de paramétrage , maintenance non chronophage

- Dynamisme et réactivité de la communauté (forums, listes etc).

- Projet en développement actif, évolutif.

- UI intuitive et claire (travail des admins facilité pour trouver l'information)

- Accès à une bonne documentation et à des ressources facilement et gratuitement (plugins…)

2] Critères Techniques : supervision des services et de l'infrastructure

Supervision du réseau

Disponibilité du matériel (SNMP), état des liens (switch/cœur de réseau), débit

Supervision des applicatifs

Disponibilité et état des services :
ssh, backup, dhcp, mysql, apache, état des clusters, vpn, etc.
Serveurs : Windows / Linux
ESX, Datastores…

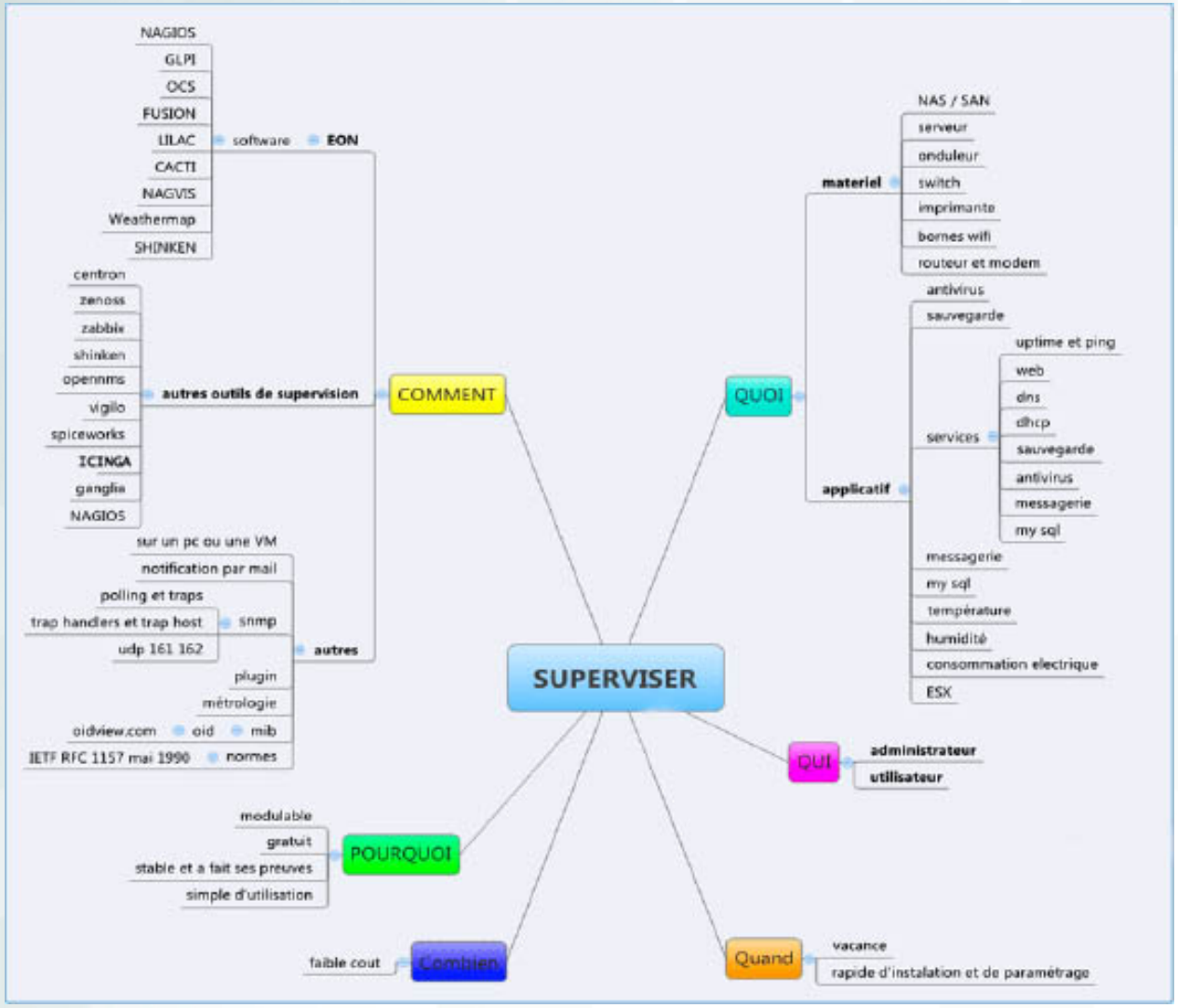
Supervision système et ressources

Serveurs et baies disques : la mémoire, disks, état du stockage, cpu…
Matériel réseau : état des systèmes (alim, fans..)
Technique : Sondes températures dans les salles serveur.

Autres critères

- Notifications
- Métrologie minimum (intégration de plugins possible)

Approche globale pour le choix du logiciel



Répond à tous les critères :
Icinga v1

ICINGA 2 : Plan de la présentation

- Origine du développement de la solution
- Caractéristiques techniques
- Installation / OS
- Configuration (exemples et spécificités)
- Gestion des notifications
- Add-ons installés / disponibles
- Evolution à terme de l'architecture à Néel (clustering + automatisation)
- Conclusion (points fort et points faibles) et perspectives

Solution retenue :

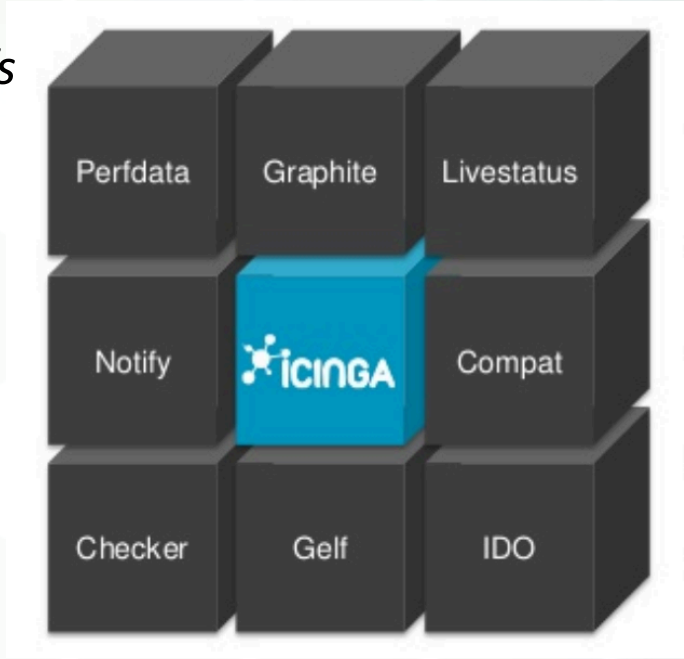
- **2009 : Icinga v1,** premier fork de Nagios, né du mécontentement des développeurs et contributeurs.
- **2013** : développement d'**Icinga2** from scratch en C++, propose des innovations en terme d'architecture.
- Depuis **2014** ~~fork~~ → **core** complètement indépendant mais permet toujours l'usage de plugins nagios *(nombreuses ressources)*
- Licence GNU GPL

Icinga 2 = Inciga2 Core + Icinga web 2

Icinga 2 core : Spécificités techniques

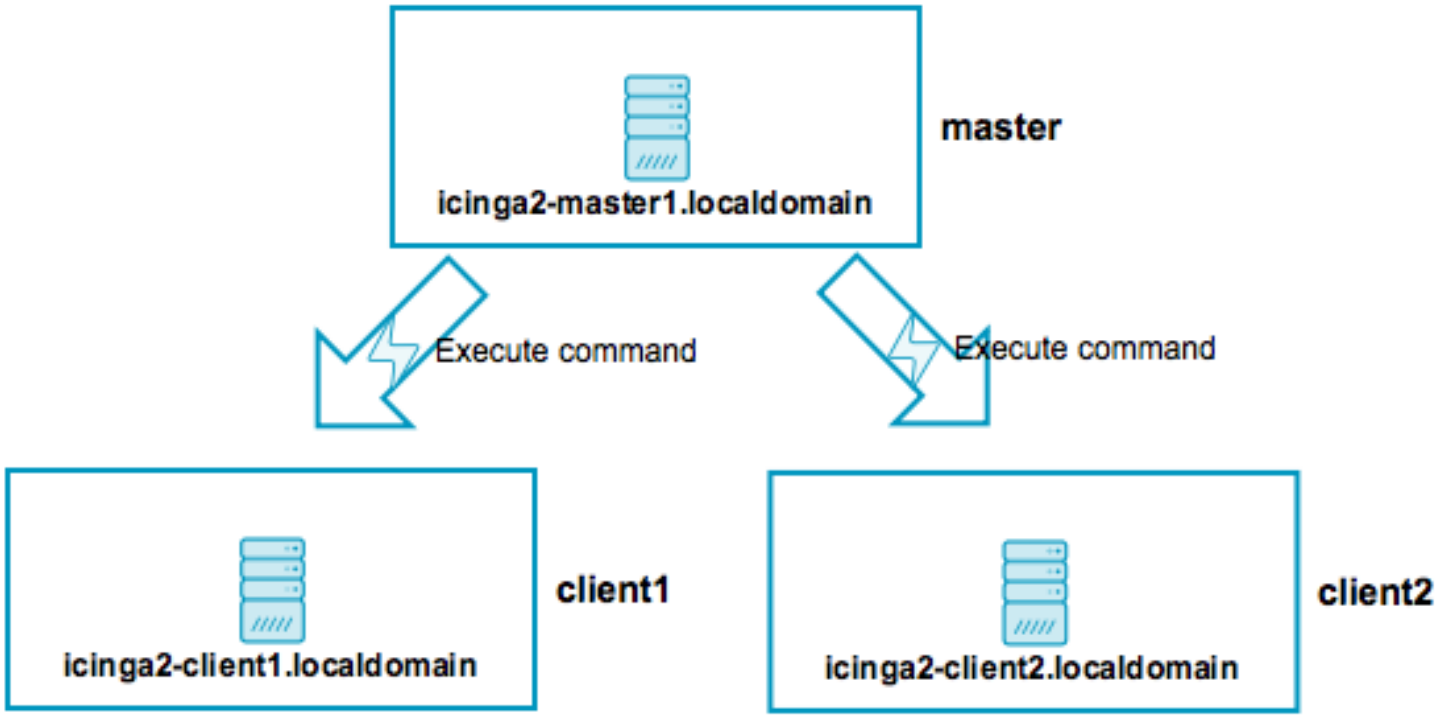
- Multithread : pas de surcharge CPU

  une machine -> 1 million de contrôles actifs pour 60 000 hôtes supervisés.
- Modulaire(≠ rôles sur ≠ machines)
- Visions globales des états (unreachable, up, down)
- Linux / Windows
- Collecte les données de performances)
- Gère les notifications
- Gère les dépendances
- Gère les évènements (event handler : qd état service change)



Architecture de départ : 1 Master / Clients

- Architecture de départ installée avec la version 1 d'Icinga en 2011 et poursuivie avec la 2 (migration en 2014).



On verra plus tard la nécessité de faire évoluer cette infrastructure pour plus de souplesse et de sécurité.

Icinga 2 core :
Installation et fonctionnalités

Installation

- **packages officiels** pour chaque distribution, ajout du dépôt (Debian, Ubuntu, FreeBSD, openSUSE, RHEL, CentOS etc.) / Windows

- **Vagrant box** avec un setup sous VirtualBox ou Parallels Desktop ou **Conteneur Docker** https://hub.docker.com/

- **par défaut 4 features activés :**

- Checker
- Notifications
- Mainlog
- Perfdata

- **Icinga 2 CLI**

- Configurer
- Effectuer des requêtes

Icinga Web 2 :
Spécificités techniques et installation

- **Visualisation et historique** des états des hôtes et services via un dashboard,
- **exportation de graphes et de métriques** grâce à des plugins.

Prérequis

- Icinga2 core avec en backend le modules **DB IDO** (Database Icinga Data Output) -> export configuration + informations d'état dans la bdd.

- **MySQL** ou PostgreSQL

- **Apache** ou Nginx

- **Librairie LDAP PHP** si Authentification Active Directory, LDAP, (possibilité d'utiliser MySQL ou PostgreSQL ou chaînage en fallback)

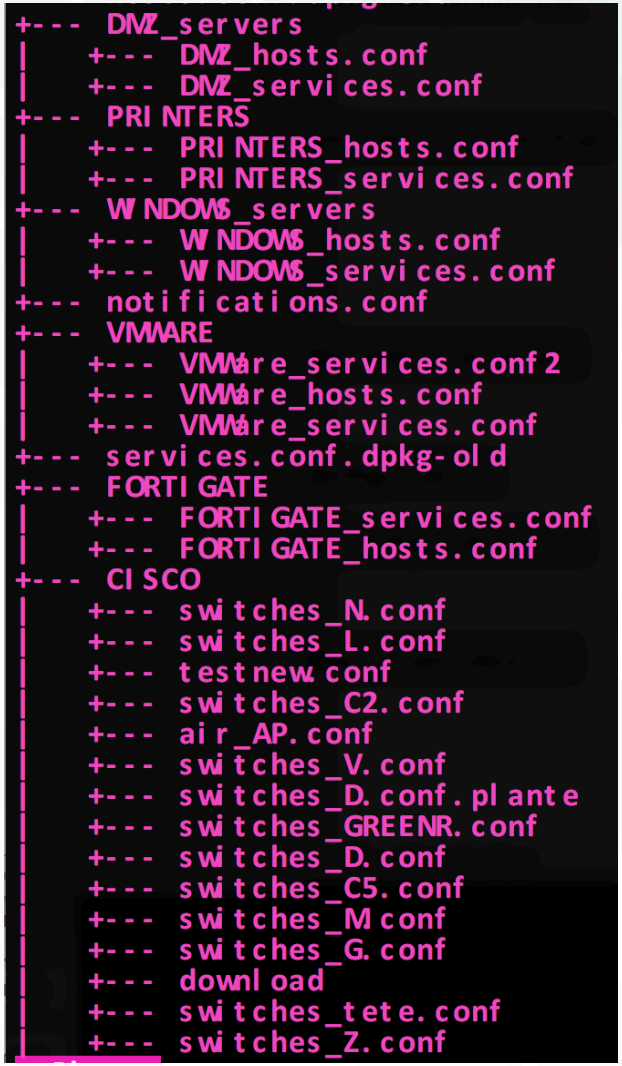
- **API REST** effectuer (check / reschedule checks, notifications) requêtes sur les différents objets

Fichiers de configuration pour le monitoring

Organisation utilisée :

- arborescence hiérarchique basée sur :

- la localisation des objets,
- des attributs communs
  (types de materiel Linux, windows, onduleurs etc…)



GENERALITES : Configuration, langage 1/3

- Langage de configuration **riche** et **intuitif**
- Basé sur **des règles** appliquée à **des objets**

1. Définition d'un objet

- Icinga2 fournit des **templates** que l'on peut adapter à ses besoins qui permettent d'appliquer un set d'attributs génériques à des objets
- Les objets héritent des templates avec la clause **import**
- On peut définir ses propres **variables locales** avec " vars "

```
template Host "linux-server" {
   import "generic-host"
   check_period = "24x7"
   check_interval = 5m
   retry_interval = 1m
   max_check_attempts = 10
import "pnp-hst"
vars.notification["mail"]   = { groups = [
"icingaadmins" ]
}
```

```
object Host "scientific1" {
  import "linux-server"
vars.vlan = "DMZ"
vars.os = "Linux"
display_name = "Scientific1 »
address = "147.173.XX.XXX"
  }
```

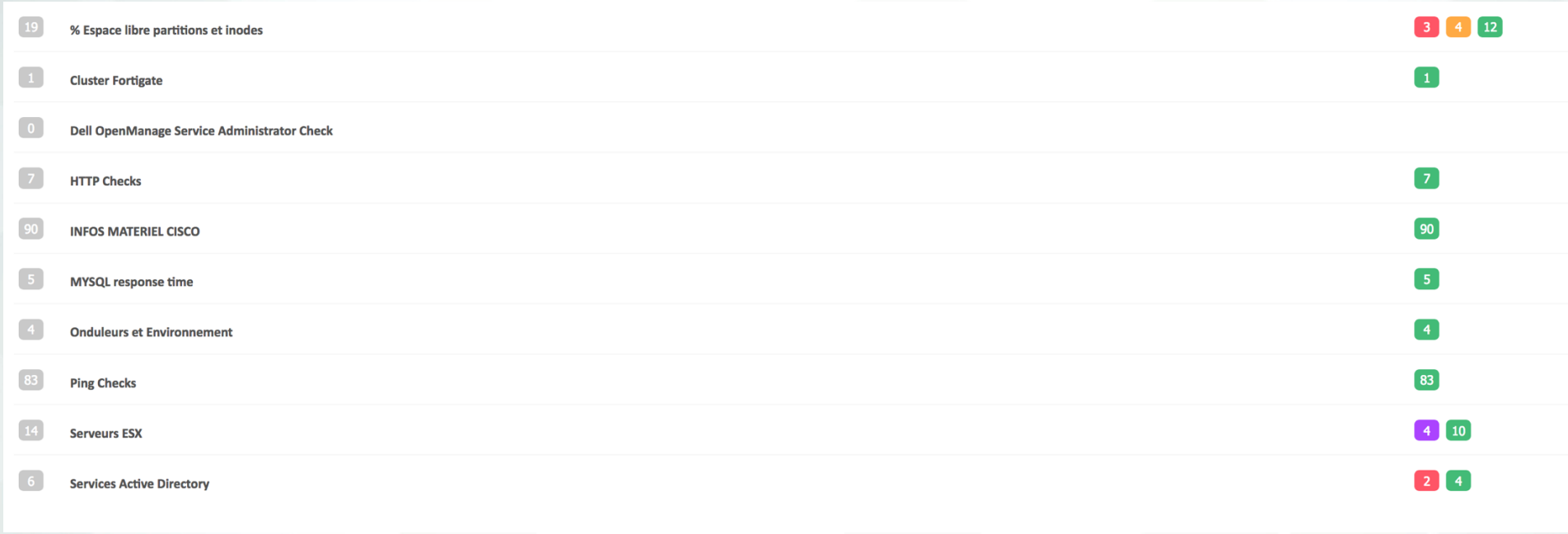
Les groupes 2/3

- Privilégie la gestion par **groupe** pour plus de flexibilité / maintenabilité

2. Définition des groupes

```
object HostGroup "linux-servers" {
  display_name = "Linux Servers"
  assign where host.vars.os == "Linux"
}
```

```
object ServiceGroup "http" {
  display_name = "HTTP Checks"
  assign where match("http*", service.check_command)
}
```



Utilisations d'Expressions pour appliquer des règles aux objets

APPLY RULES 3/3

la clause apply permet d'appliquer un comportement à un type d'objet

3. Définition des règles d'applications

Plusieurs **hôtes partagent** un même **groupe de services**

```
apply Service "MySQL slow queries" {
      ...
  assign where "SQL-servers" in host.groups
    }
```

```
apply Service "ssh" {
....
  assign where host.address  && host.vars.os ==  "Linux"
}
```

Filtres complexes avec conditions multiples : grâce aux variables et fonctions

```
apply Service "Memoire Systeme" (
    ....
  assign where match "windows-servers", host.groups) && !cidr_match("192.168.144.0/24",host.address)
)
```

```
apply Service "web" {
  ...
  assign where match("web*.neel.cnrs.fr", host.name)
  ignore where host.vars.no_web_check
}
```

On peut appliquer des services, notifications et des dépendances aux objets.

Monitoring direct (sans agent local)

Si service / hôte disponible (via port / protocole)

- La bibliothèque de templates Icinga (ITL) : complète
- besoins plus spécifiques : http://exchange.icinga.org ou Monitoring Plugins Project.

Plugins externes :

- Définir objet **CheckCommand**
- Configurer arguments / seuils
- Importe **template** : plugin-check-command.

Exemple de CheckCommand de contrôle de l'état d'un service

```
object CheckCommand "ftp" {
        import "plugin-check-command"
        command = [ PluginDir + "/check_ftp" ]
        arguments =  "-H" = "$ftp_address$"
                     "-p" = "$ftp_port$"
 }
 vars.ftp_address = "$address$"
 }
```

supervision via un agent local (1/4)

Services distants non accessibles directement par le réseau,
Installation d'**agents locaux** pour obtenir les résultats des requêtes check.

SNMP

Configurer une communauté
Nombreux plugins SNMP existants…

```
apply Service "snmp-load" {  import "generic-service"
display_name = "Cisco CPU Load"
check_command = "snmp-load"
       vars.sla = "24x7"
       vars.snmp_warn = "90,80,70"
       vars.snmp_crit = "100,100,100"
 assign where host.vars.snmp_community  != "" && host.vars.brand=="CISCO"
}
```

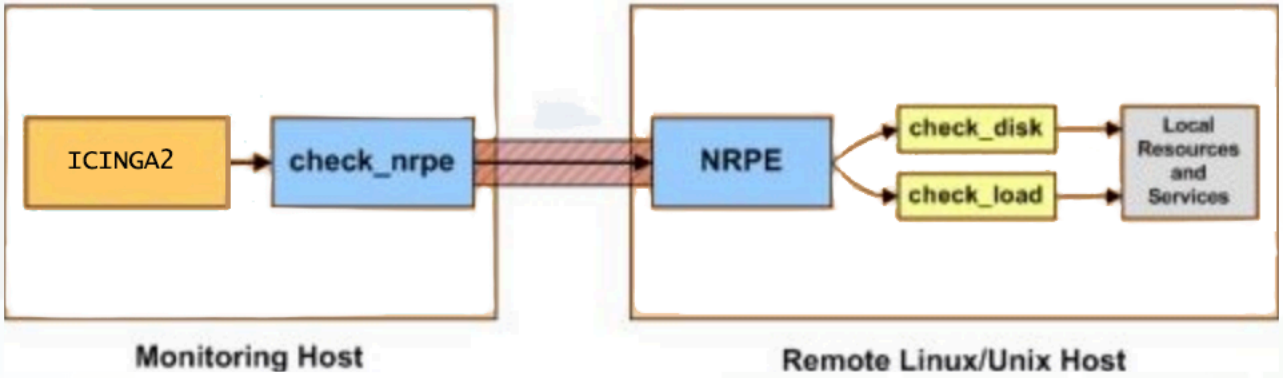
…facile d'appliquer une règle à tous les switches cisco semblables

supervision via un agent local (2/4)

NRPE (Nagios Remote Plugin Executor)

Exécution de commandes de supervision sur des machines distantes (Linux et windows) où daemon NRPE installé et des plugins configurés.

Icinga 2



dernières versions du core : client icinga2 lancé comme un service installé sur clients avec les plugins (remplace nrpe).

supervision via un agent local (3/4)

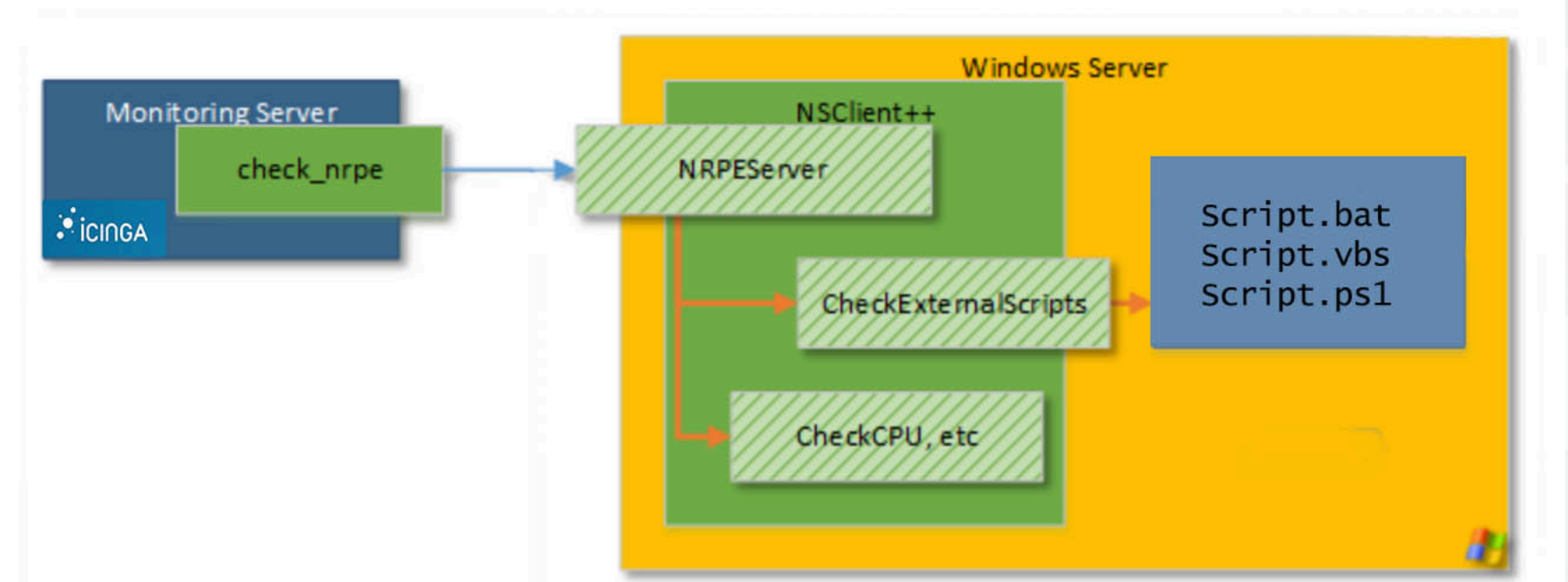
SSH

Exécution plugin sur un serveur distant via ssh (Pas utilisé à Neel)

NSClient++

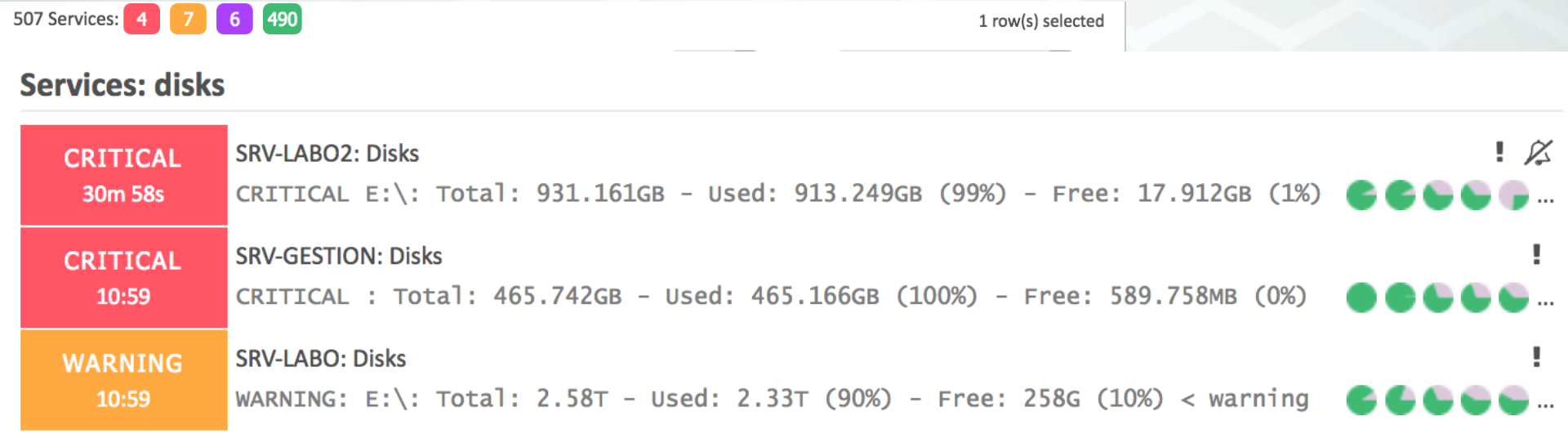
Sur les serveurs Windows :
on installe NSCLient++ (rôle d'agent NRPE) pour exécuter ces checks à distance

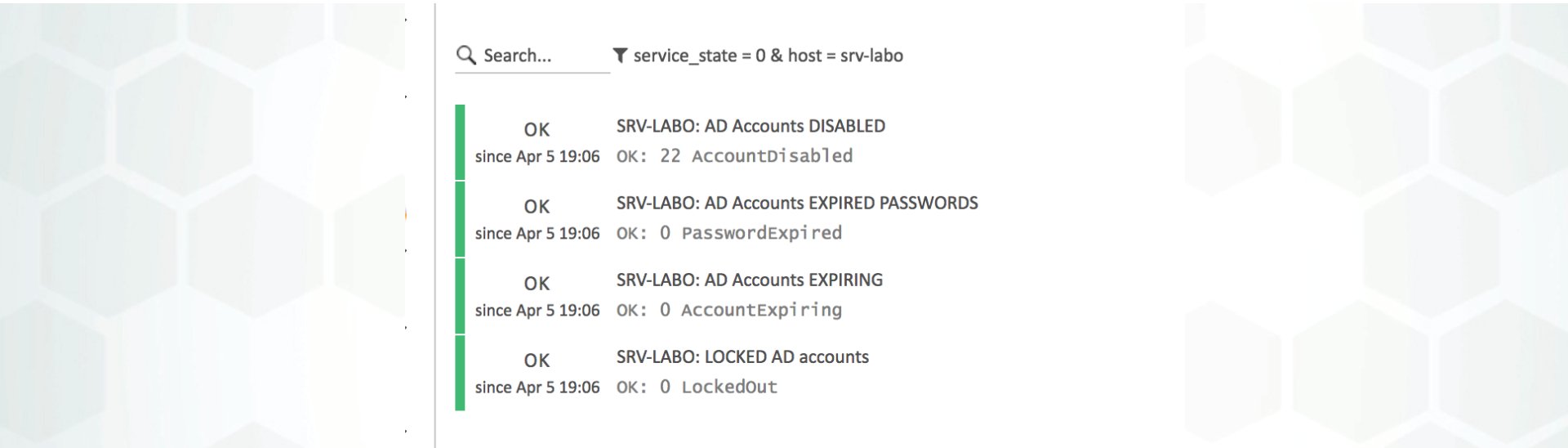
Comment ?
On utilise des commandes internes (CheckSystem) ou des scripts externes exécutés localement
Le plus souvent en **VB** ou en **Powershell**



supervision via un agent local (4/4)

Ex : checker des disques / cpu/ ou grâce à des scripts externes un AD, ou même un serveur dell géré via OMSA





NOTIFICATIONS

Lors de la détection d'un problème :

SOFT	L'état de l'hôte / service a changé récemment: re-checké (retry_interval)	→ **Pas de notifications**
HARD	L'état de l'hôte / service n'a pas changé récemment.	→ **Notifications envoyées**

On peut :

- Appliquer des notifications directement aux objets
- **Best practice** : créer un template notification et utiliser "apply" pour assigner la notification à un nombre d'hôtes / services
- **Escalader une notification** (au bout d'un temps de non résolution) ou la temporiser (intervalle)

```
apply Notification "default-linux" to Host
{
  import "24x7email_host"
  user_groups = [ "icingaadmins" ]
  types = [ Problem, Recovery ]
  states = [ Critical, Warning, Unknown, OK ]
  assign where host.vars.sla == "24x7" && host.vars.os == "Linux"
}
```

NOTIFICATIONS

Optimisation :

- Gérer les intervalles de notifications (criticité du service)
- Utiliser les dépendances
- Programmer « downtimes » (met le service en mode maintenance : coupure d'électricité etc…)
- Cibler les bonnes personnes : (créer des **utilisateurs** et des **groupes d'utilisateurs**)

Ex : ASR reçoivent des **notifications** suivant leur **domaines de responsabilité**

Dépendances et héritage

Disponibilité réseau des services et des hôtes : les objets de dépendance

- **Un service dépend implicitement de son hôte**
  notifications de service supprimées si hôte DOWN ou UNREACHABLE.

- **Dépendances inter-hôtes** déterminées en indiquant le type de relation (hôte / parent).

- **Dépendances inter-services**
  Service parent -> WARNING = objet enfants (hôte ou services) UNREACHABLE -> pas de notifications.

```
object Host "child" {
  import "generic-host"
  address = "10.10.10.50"


  vars.parents = [ "parent1",
"parent2", "parent3" ]
}

apply Dependency "Parent" for
(parent in host.vars.parents) to Host
{
  parent_host_name = parent
  assign where host.address &&
host.vars.parents
}
```

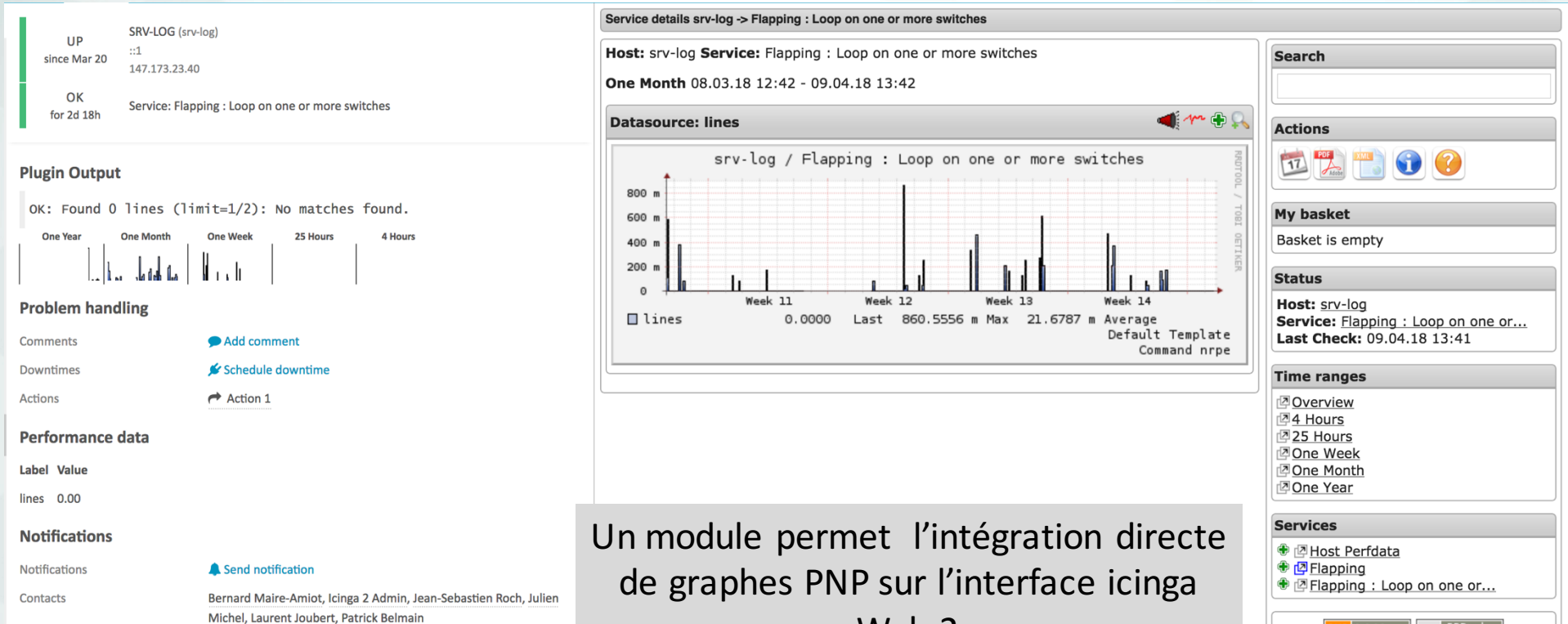
Graphing / données de performances

PerfData :

- peuvent être collectées via des **databases externes** de stockage de métriques
- **visualisées sur des frontend externes** ou intégrées sur Icinga web2.

PNP

PNP4Nagios (add-on) : permet de visualiser sous forme de graphe les données de performances générées par Icinga2.
(feature perfdatawriter + NPCD pour la collecte des data + module Web à configurer).



Un module permet l'intégration directe de graphes PNP sur l'interface icinga

Autres add-ons de graphing disponibles (pas encore en place au labo)

Fonctionnalités pour exporter métriques (appli externes) :

Graphite

- Feature GraphiteWriter, + Graphite en backend

Ajoute des graphes dans l'interface au niveau des hôtes et des services

Autre frontend : grafana

InfluxDB / Grafana en frontend

- Feature InfluxDBWriter permet d'envoyer des métriques en temps réel depuis Icinga2 vers Influx DB

Visualisation

Nagvis

Evolution de l'architecture envisagée (1/3)

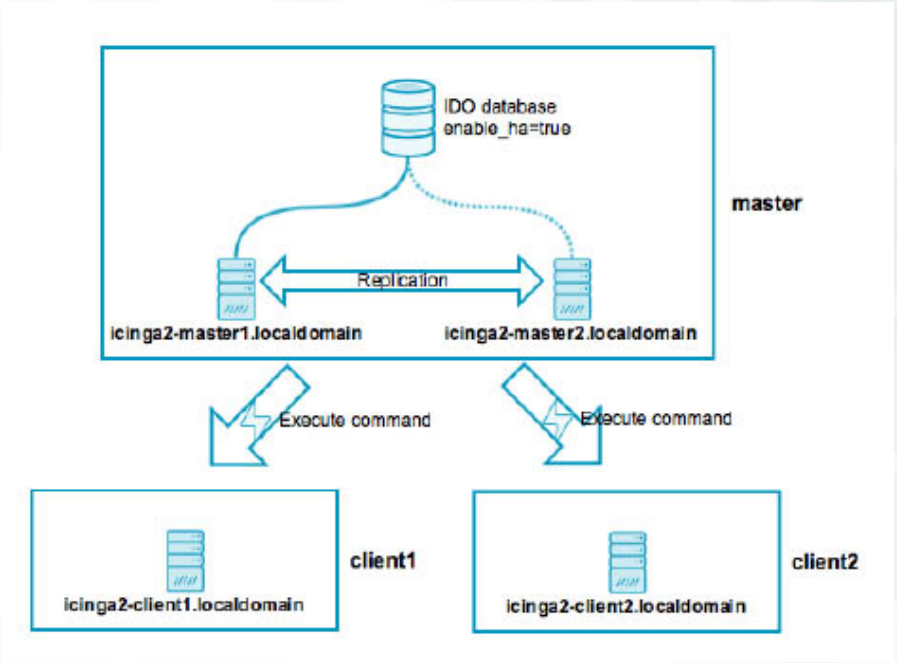
2 changements majeurs dans les dernières versions : l'architecture distribuée et échanges sécurisés

Cluster haute disponibilité et des clients

1. Cluster / clients /zones

- Passer en cluster pour plus de souplesse:
- 2 nœuds **master** au sein d'1 Zone
- Synchronisation automatique entre les 2

- Les **services check** sont envoyés depuis le master qui exécute une commande sur un hôte distant (command_endpoint)



Evolution de l'architecture envisagée (2/3)

2. Clients

- une méthode **intégrée** permet depuis versions récentes d'auto-signer les demandes de certificats des clients ou de les signer à la demande grâce à des tickets envoyés au master

Avantages nouvelles architecture :

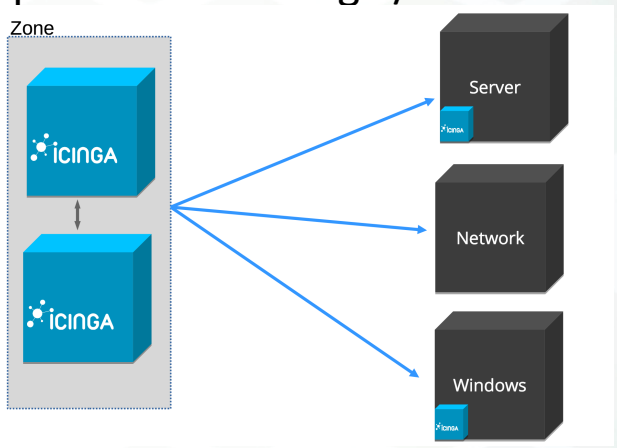
- échanges master / client sont sécurisés et utilisent le **même protocole natif** de part et d'autre **-> plus de NRPE**

- Toutes les commandes sont définies sur le master et les nouvelles (custom_command) synchronisées au sein d'une GlobalZone

- checks et notifcations équilibrés entre les 2 maîtres : répartition de charge / tolérance de pannes

Inconvénients

- **Installation sur tous les clients d' Icinga2 lightweight obligatoire** + plugins désirés pour que le master puisse les utiliser



Evolution de l'architecture envisagée (3/3)

Problèmes :

- actuellement 1 seule personne configure Icinga : lourd à gérer / manque d'automatisation des confs

Solution : Automatiser davantage avec l'outil **Icinga Director (créé en 2017)**

Director :

- Outil de configuration intégré à Icingaweb2 comme un module
- Déployer une conf via interface web
- Plusieurs utilisateurs pour modifier la conf
- Log d'audit pour visualiser les changements effectués
- Possibilité de redéployer d'anciennes configs
- Pré-requis : icinga 2.6, Icinga Web2, une database PHP et php-curl.

Problème : difficile d'importer des définitions existantes créés sous forme de fichiers plats vers Director… Par contre on peut importer des hôtes d'un CMDB existant.

ET/OU Mettre en place un gestionnaire de configuration + inventaire serveurs pour déployer les hôtes et les services (via ansible, puppet…)

Avantages Icinga2

- **Le langage de configuration** et le design orienté « **objets** »qui permet de mettre en place des configurations complexes en un minimum d'efforts

- **L'interface Web** claire et détaillée

- Développeurs et **communauté très réactive** sur les forums, rapidement de l'aide en cas de blocage

- L'usage des **templates** qui facilite l'application des commandes aux hôtes

.

- **L'équipe de développeurs qui** travaille sur des intégrations avec des acteurs phares de l'administration système centralisée (puppet, chef, ansible, Elastic (Icingabeat / logstash plugin + ElasticSearch writer) : **beaucoup de ressources en développement**

- **Roadmap précis**

Inconvénients

- Doc exclusivement en anglais
- Les notifications qui peuvent vite s'emballer si les dépendances n'ont pas toutes été configurées
- Documentation très détaillée parfois sujette à interprétations…

Service Problems

CRITICAL Apr 11	SRV-GESTION: Disks CRITICAL : Total: 465.742GB - Used: 464.539GB (100%) - Free: 1.203GB (0%)
CRITICAL Apr 6	SRV-LABO2: Disks CRITICAL E:\: Total: 931.161GB - Used: 913.944GB (99%) - Free: 17.217GB (1%)
CRITICAL Apr 5	SRV-LABO: AD Accounts EXPIRED CRITICAL: 248 AccountExpired
CRITICAL Apr 5	SRV-LABO: AD Accounts INACTIVE CRITICAL: 1667 AccountInactive
CRITICAL Apr 5	NEEL1: / DISK CRITICAL - free space: / 2302 MB (5% inode=3%):
UNKNOWN Apr 11	ESX1: Datastore Status Welcome to vSphere Management Assistant
UNKNOWN Apr 11	ESX2: Datastore Status Welcome to vSphere Management Assistant
UNKNOWN Apr 11	ESX1: Datastore Status Welcome to vSphere Management Assistant
UNKNOWN Apr 11	ESX2: Datastore Status Welcome to vSphere Management Assistant
UNKNOWN Mar 5	K1-SW144.44 - WS-C2960-24TT-L: Indicateurs Environnement Cisco Unknown option: w

Show More

Recently Recovered Services

OK 32m 6s	Institutional: /home DISK OK - free space: /home 8494 MB (96% inode=99%):
OK 39m 18s	Institutional-Web: /home DISK OK - free space: /home 4097 MB (56% inode=95%):
OK 44m 17s	C1-MEZZ-SW144.192 - WS-C2960S-48TS-L: ping4 PING OK - Paquets perdus = 0%, RTA = 0.82 ms
OK 12:33	E0-SW144.103 - WS-C2950T-24 : ping4 PING OK - Paquets perdus = 0%, RTA = 4.26 ms
OK 12:26	C6-SW144.188 - WS-C2960S-24TS-L : ping4 PING OK - Paquets perdus = 0%, RTA = 1.92 ms
OK 12:20	C3-R+1-SW144.65 - WS-C2960S-48TS-L : ping4 PING OK - Paquets perdus = 0%, RTA = 0.86 ms
OK 12:12	D1-SW144.1 - WS-C2960S-48TS-L : ping4 PING OK - Paquets perdus = 0%, RTA = 2.01 ms
OK 12:12	C6-SW144.188 - WS-C2960S-24TS-L : Indicateurs Environnement Cisco Unknown option: w
OK 11:51	F1-SW144.203 - CISCO WS-C2960-24TT-L : ping4 PING OK - Paquets perdus = 0%, RTA = 1.77 ms
OK 11:24	SRV-GESTION: Memoire Systeme OK: physical: Total: 7.987GB - Used: 4.689GB (58%) - Free: 3.298GB (41%), virtual: Total: 8TB - Used: 116.559MB (0%) - Free: 8TB (99%), committed: Total: 15.973GB - Used: 4.317GB (27%) - Free: 11.656GB (72%), committed: Total: 15.973GB - Used: 4.317GB (27%) - Free: 11.656GB (72%)

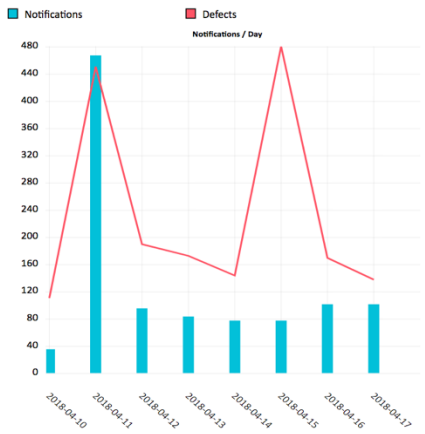
Show More

Host Problems

No hosts found matching the filter.

alert summary

Notifications and Problems



Time to Reaction (Ack, Recover)

ServiceGroups

	Service Group	Service States
19	% Espace libre partitions et inodes	3 2 14
1	Cluster Fortigate	1
0	Dell OpenManage Service Administrator Check	
7	HTTP Checks	7
90	INFOS MATERIEL CISCO	90
5	MYSQL response time	5
4	Onduleurs et Environnement	4
83	Ping Checks	83
14	Serveurs ESX	4 10
6	Services Active Directory	2 4

Technical Overview

120 Hosts UP

5 CRITICAL	5 WARNING
7 UNKNOWN	513 OK