Processi di monitoraggio e Nagios

Svilupparsi una rete complessa e distribuita

di apparati e servizi,

assicurare LdS decenti,

continuare a vivere senza frustrazioni

Di cosa tratteremo

Innalzare la disponibilità dei servizi agli utenti

Gli obiettivi del monitoraggio

Che fa l'industria?

Specificità della rete ninux

Networking e non solo

Lo strumento Nagios

Possibili ulteriori evoluzioni del monitoraggio in ninux

Hands on

Inquadriamo la problematica

Innalzare la disponibilità dei servizi

- ✔ Console unica
- ✔ Proattività nella rilevazione e gestione degli eventi
- ✔ Correlazione tra eventi
- ✔ Identificazione dei punti di debolezza strutturale
- ✔ Emersione dei trend
- ✔ Configuration DB
- ✔ Pianificazione e pubblicazione dei downtime
- ✔ Non spendere una tombola

Gli obiettivi del monitoraggio

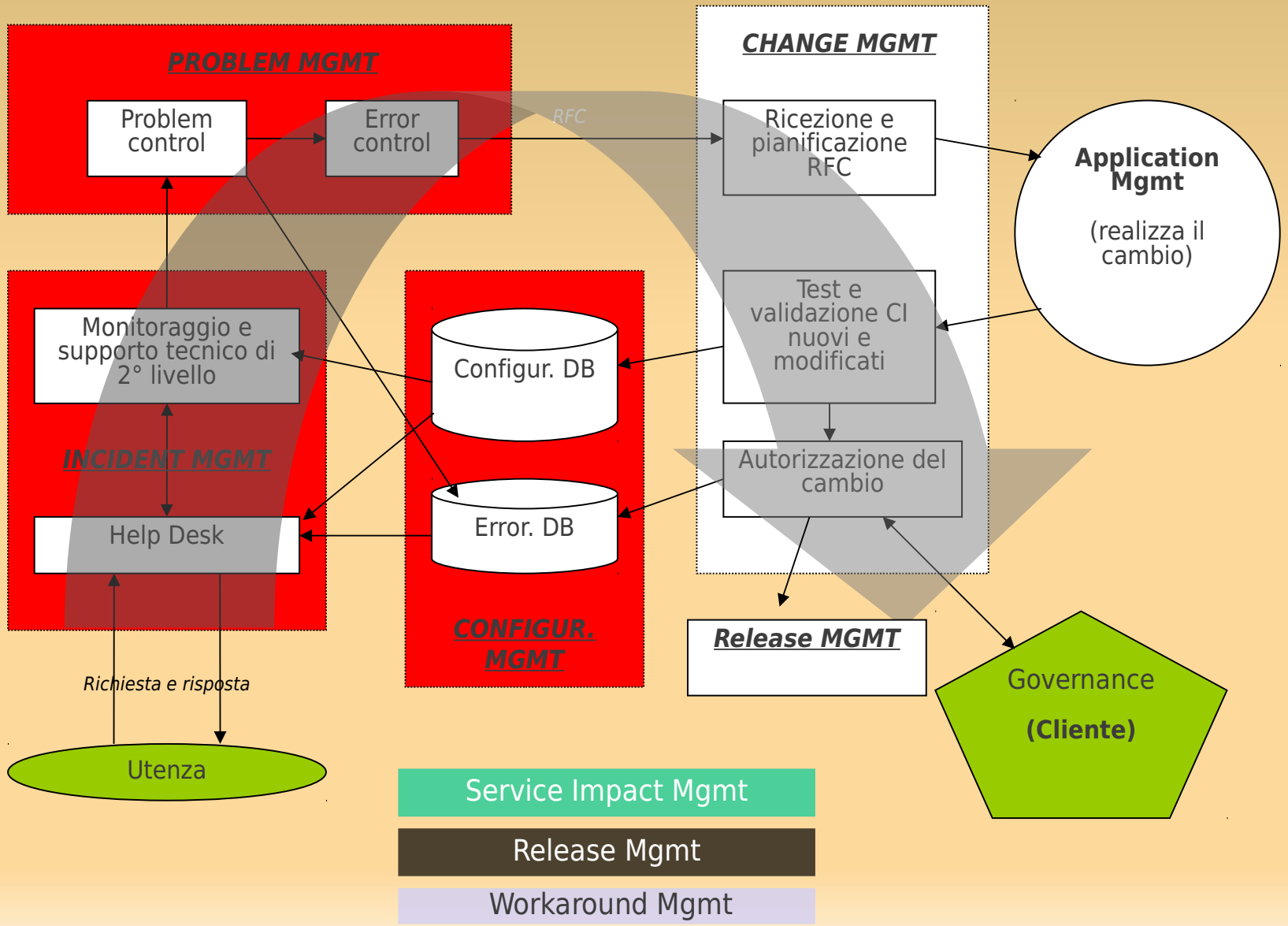
Network & system monitoring conventionally consists of observing an infrastructure's data traffic in order to ***identify_bottlenecks*** *or to* ***troubleshoot*** *hardware and application issues*

Che fa l'industria?

- ✔ Codifica dei processi di monitoraggio, assistenza, supporto, documentazione e change management
- ✔ Adozione delle migliori pratiche (ITIL)
- ✔ Strumenti integrati
- ✔ Organizzazione + Process ownership
- ✔ Policy di esercibilità, promozione controllata
- ✔ Documentazione (knowledge base, configuration DB)
- ✔ Trouble ticketing parossistico di qualsiasi intervento
- ✔ Rigoroso governo dei change (di nuovo!)
- ✔ €, €, €, €, €, ...

Piccolo focus su ITIL

L'80% dei disservizi derivano da un change mal gestito



Specificità della rete ninux

- ✔ Infrastruttura radio e wired in rapido->caotico sviluppo
- ✔ Topologia dinamica
- ✔ Monitoraggio “in-band”
- ✔ Minimi percorsi ridondati
- ✔ Proprietà individuale e gestione “autonoma” degli elementi della infrastruttura
- ✔ Policy zero: decentralizzazione “di tutto”
- ✔ Best effort vs. expectations
- ✔ Accesso WiFi libero al pubblico

Network management e non solo

- ✔ VPN internazionali
- ✔ Streaming radio
- ✔ Hot spot WiFi
- ✔ Asterisk
- ✔ NAS
- ✔ Ping, web, ssh, mail, wiki, ...
- ✔ CPU load nei router
- ✔ Performance e altri aspetti legati alla modalità di trasmissione radio
- ✔ Intrusioni

Lo strumento Nagios

Introduzione a Nagios

- ✔ Monitoraggio sulla disponibilità dei servizi di rete: POP3, HTTP, FTP, SSH, ….
- ✔ Monitoraggio sistemistico: CPU, disco, RAM, processi, …
- ✔ Monitoraggio applicativo
- ✔ Gestione della topologia della rete e della gerarchia dei servizi
- ✔ Possibilità di definire script da attivare automaticamente per particolari eventi
- ✔ Nessun problema nel ridondare il monitoraggio da sedi diverse
- ✔ Definizione di raggruppamenti arbitrari dei monitoraggi: per servizio, tipologia di host, tipo di misura, collocazione geografica, ...
- ✔ Gestione dei fermi pianificati, per evitare allarmi durante le indisponibilità previste
- ✔ Visualizzazione e reportistica multidimensionale delle misure effettuate
- ✔ Gestione delle autorizzazioni e dei profili utente
- ✔ Integrabilità con altri strumenti e protocolli: MRTG, SNMP, cacti, ...

Start-up

- ✔ Scritto in C, è disponibile su linux, Windows, Open-WRT (pkg Nagios2)
- ✔ Si compone di un core + plug-in verticali che eseguono i check attivi + listener passivo (NSCA) che accetta segnalazioni da applicazioni esterne
- ✔ La libreria di plug-in verticali è già molto estesa (1919 presenti in exchange.nagios.org) e copre le necessità più comuni
- ✔ La scrittura di nuovi plug-in è documentata e può appoggiarsi a framework di supporto per Python e perl
- ✔ Sono disponibili molteplici skin CSS e diversi front-end WEB di interrogazione, organizzazione e presentazione dei dati (+ quelle *mobile-oriented*, es. Nagroid)
- ✔ Molto leggero: l'attuale monitoraggio di ninux (ca. 60 nodi) avviene da un portatile ATOM, impegnandone una frazione di CPU
- ✔ Tramite il modulo NRPE può eseguire i plug-in di check direttamente sull'host remoto da monitorare
- ✔ L'esecuzione dei check è IPv6 “aware”

Configurazione di base

- ✔ Nagios è impostato tramite un file `nagios.cfg` principale e un albero di file `.cfg` dedicati
- ✔ Il numero e la posizione dei file `.cfg` dedicati è insignificante
- ✔ Nei `.cfg` sono definiti tutti gli oggetti Nagios: contatti, servizi, host, comandi
- ✔ Ogni oggetto è definito in un file con attributi obbligatori e opzionali
- ✔ Il concetto di “parents” permette di definire gerarchie di servizi/host e topologie di rete
- ✔ Per agevolare il task di configurazione, Nagios permette opzionalmente di specificare per ogni oggetto un oggetto “padre” da cui ereditare valori (con la direttiva `use`)

Meccanismo dei check

- ✔ Nagios utilizza i plug-in per effettuare i test di monitoraggio (`command_line`)
- ✔ Un plug-in è un eseguibile (tipicamente ma non obbligatoriamente uno script) che, quando invocato, ritorna uno tra quattro possibili valori:

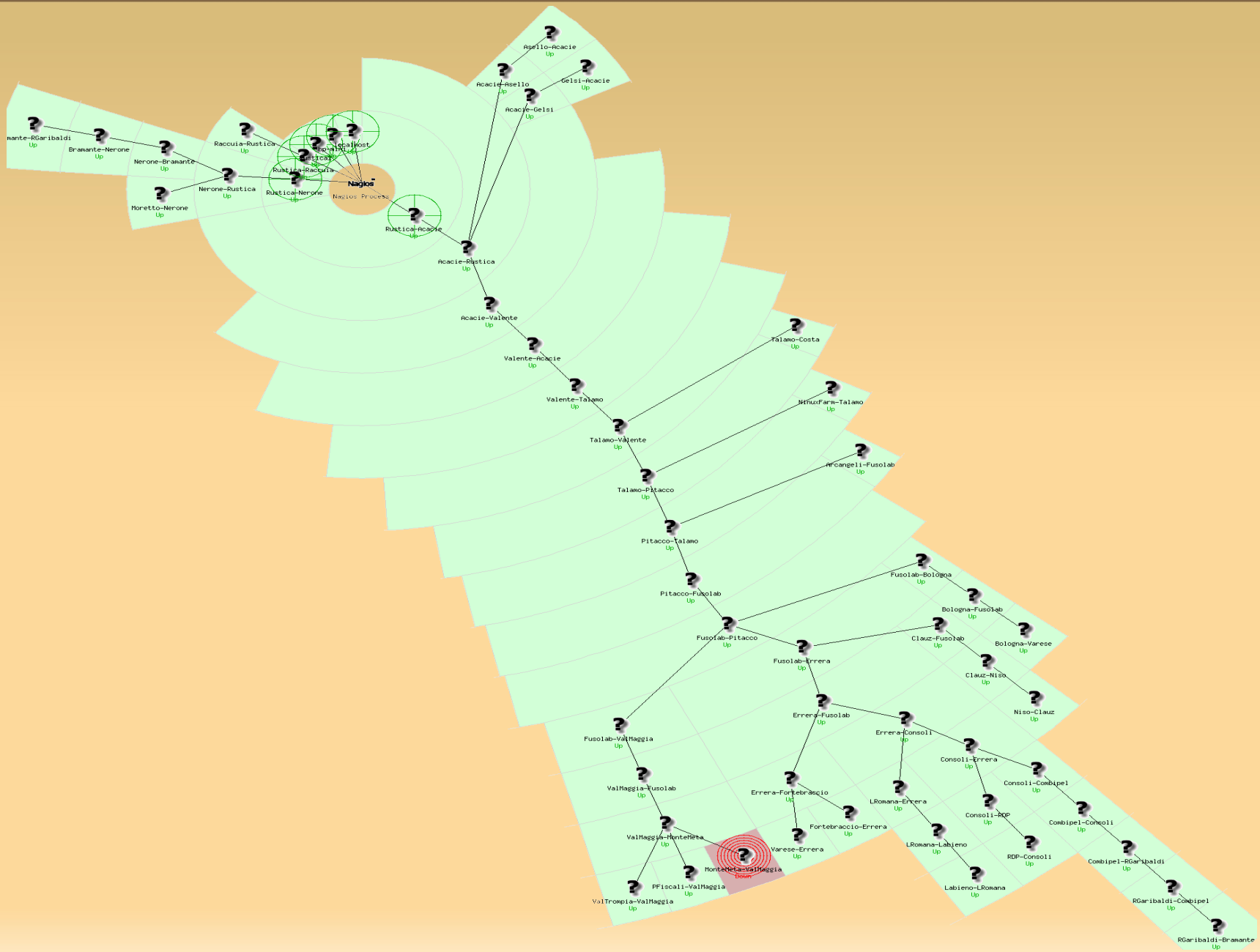
  `0` - `OK`
  `1` - `WARNING`
  `2` - `CRITICAL`
  `3` - `UNKNOWN` (errore nell'esecuzione del plug-in stesso)

- ✔ Ogni test viene ripetuto con periodo `normal_check_interval` (minuti)
- ✔ In caso di risultato `CRITICAL` il test è ripetuto `max_check_attempts` ogni `retry_check_interval` minuti, prima di essere confermato
- ✔ Lo script deve accettare alcuni parametri standard e può accettarne altri dipendenti dal tipo di check stesso
- ✔ Il plug-in, per essere utilizzabile, deve essere specificato con un `command_name` dentro una `define command {}`
- ✔ Il comando è poi invocato, con i suoi parametri e argomenti, dentro una direttiva `define service {}` oppure una `define host {}` tramite la specifica `check_command`

Esempio di definizione: oggetto host

```
define host {
    name                            topolino
    check_command                   check-host-alive
    contact_groups                  Admins
    active_checks_enabled           1
    check_period                    always
    max_check_attempts              3
    event_handler_enabled           1
    process_perf_data               1
    retain_status_information       1
    retain_nonstatus_information    1
    notifications_enabled           1
    notification_interval           120
    notification_period             always
    notification_options            d,u,r
    parents                         minnie
    register                        1
}
```

Screenshot della topologia (ninux Roma)



Meccanismo dei check (esempi)

```
define command {
    command_name     some-command
    command_line     $USER1$/check_something $ARG1$ $ARG2$
}

define service {
    host_name              <lista degli host oppure hostgroup cui applicare il test>
    service_description    some-service
    check_command          some-command!arg-1!arg-2
    [...]
}
```

```
define command {
    command_name    check-DNS
    command_line    $USER1$/check_dns -s $HOSTADDRESS$ -H $ARG1$ -a $ARG2$
}


define service {
    host_name              pippo, pluto, paperino
    service_description    HOST_IS_IN_DNS_SERVER
    check_command          check-DNS!8.8.8.8
}
```

Esecuzione di test remoti

- In caso di servizio remoto affacciato direttamente sulla rete (es. httpd), l'esecuzione del plug-in è locale al server Nagios
- In tutti gli altri casi, occorre eseguire remotamente una qualche azione:
  - Con `check_by_ssh/sshd` eseguo il plug-in sull'host remoto
  - Con `check_nrpe/nrpe` eseguo il plug-in sull'host remoto
  - Con `check_snmp/smnpd` interrogo una MIB sull'host remoto
  - Con NSCA ascolto il risultato di un check locale, trasmesso tramite `send_nsca`

Esempio: il monitoraggio di un relay SMTP

Alcune situazioni riconosciute

- ✔ FLAPPING
  - Questa situazione viene riconosciuta quando Nagios rileva una frequenza di cambio stato superiore a una data soglia massima (parametrica per servizio) sulle ultime 21 misurazioni effettuate
  - Il flapping viene segnalato in console con un messaggio e una icona specifici
  - Un servizio esce dal flapping quando la frequenza dei cambiamenti di stato torna inferiore a una soglia minima (parametrica per servizio)
- ✔ GESTIONE EVENTI
  - Nagios può essere esteso per gestire le transizioni di stati e conservarne traccia in un EventDB (è un modulo separato)
- ✔ MANUTENZIONE PROGRAMMATA
  - L'interfaccia Web di Nagios consente di specificare, per ogni host e ogni servizio definito, un calendario di manutenzioni programmate
  - Gli allarmi provenienti da un host/servizio in manutenzione non vengono considerati ne notificati
  - Il calendario interventi può anche essere prodotto con uno strumento esterno e importato in Nagios da file

Meccanismo di notifica degli alert

- ✔ Ogni oggetto host/service possiede un riferimento (nome) esplicito
- ✔ Uno o più riferimenti possono appartenere a un gruppo di contatto
- ✔ Quando si produce un allarme su un oggetto, Nagios provvede a inviare una notifica (mail, sms) a tutti i referenti previsti per il gruppo di contatto
- ✔ IMPORTANTE: in caso di **flapping** di un oggetto, Nagios smette di inviare notifiche fino alla normalizzazione della situazione.

Definizione dei contatti

```
define contact {
   contact_name                    niccolo@home
   alias                           Niccolò Avico
   host_notifications_enabled      1
   service_notifications_enabled   1
   host_notification_period        nonworkhours
   service_notification_period     nonworkhours
   host_notification_options       d,u
   service_notification_options    c
   host_notification_commands      host-notify-by-email,host-notify-by-SMS
   service_notification_commands   notify-by-email,notify-by-SMS
   email                           niccolo@avico.it
   address1                        3451234567
   can_submit_commands             1
}
```

```
define contact {
   contact_name                    niccolo@work
   alias                           Niccolò Avico
   host_notifications_enabled      1
   service_notifications_enabled   1
   host_notification_period        workhours
   service_notification_period     workhours
   host_notification_options       d,u
   service_notification_options    c
   host_notification_commands      host-notify-by-email,host-notify-by-SMS
   service_notification_commands   notify-by-email,notify-by-SMS
   email                           niccolo@postodilavoro.it
   address1                        33334445555
   can_submit_commands             1
}
```

Dichiarazione del gruppo di contatto

```
define contactgroup {
    contactgroup_name                Admins
    Alias                            ninux administrators
    Members                          niccolo@work,niccolo@home
}
```

```
define host {
    Name                             topolino
    check_command                    check-host-alive
    contact_groups                   Admins
    active_checks_enabled            1
    check_period                     always
    max_check_attempts               3
    event_handler_enabled            1
    process_perf_data                1
    retain_status_information        1
    retain_nonstatus_information     1
    notifications_enabled            1
    notification_interval            120
    notification_period              always
    notification_options             d,u,r
    parents                          minnie
    Register                         1
}
```

Alcuni plug-in utili per ninux

- ✔ AirOS M5 : http://www.omniflux.com/devel/nagios/pnp4nagios-graphs.png
- ✔ AirOS: http://svn.jasonantman.com/public-nagios/check_frogfoot.php
- ✔ AirOS: http://svn.jasonantman.com/public-nagios/check_802dot11.php
- ✔ Asterisk: http://exchange.nagios.org/directory/Plugins/Telephony/Asterisk
- ✔ Intrusion detection SSH: http://exchange.nagios.org/directory/Plugins/Security/check_ssh_faillogin/details
- ✔ NPC: Nagios Plug-in per Cacti, rimpiazza la *user interface* standard Nagios: http://trac2.assembla.com/npc

TO DO

- ✔ Completare il DB degli apparati
- ✔ Inventario e configurazione dei servizi per host
- ✔ Completamento utenze e contatti per apparato
- ✔ Replica del monitoraggio (altri 2 siti)
- ✔ Approfondimento dei plug-in per i servizi specializzati (AirOS, Asterisk, VPN, …)
- ✔ Automazione della configurazione dei router, a partire dal DB dei nodi
- ✔ Integrazione cacti
- ✔ … (l'appetito vien mangiando)