

Prometheus in Small and Medium Businesses

Why You Don't Need to Do Rocket Science
(Kubernetes) to Use It

'nethesis

Matteo Valentini

 @_Amygos



About Nethesis

Nethesis: an example of small medium business

An italian Open Source IT company

~ 30 employees

Creator, main sponsor and contributor of Nethserver, an open source linux distribution

- <https://www.nethserver.org/>
- <https://community.nethserver.org/>

The Nethesis core business is the selling of support to their resellers, on Nethesis's products based on Nethserver distribution.

Nethesis: why adopt Prometheus?

- Not happy with old solution based on Nagios/Adagios
- Launch of a new service based on the immutable infrastructure paradigm
- Try a new thing :)

Nethesis: the initial monitoring scenario

16 static host to monitor:

- System metrics
- CPU/RAM alerts
- UP/DOWN alerts
- Response latency of some service

1 Dynamic system

The infrastructure

Infra: VM Instance

- Hosted in house
- Proxmox Virtual Environment
- Single node instance
 - Centos 7
 - 40 Gb disk
 - 1 Gb ram
 - 1 vCPU
- Service installed:
 - Prometheus
 - Grafana
 - AlertManager
 - Blackbox exporter

Infra: provisioning

- Provisioned using Ansible
 - Most of the roles came from Cloudfalchemistry
- Versioning using git
- Manual apply of ansible playbook

Infra: exporters configuration

- Provisioned with Ansible
- Access policy based on source IP (from our assigned IP range)
 - Cloud firewals
 - iptables ruels

Prometheus configuration

Prometheus: labeling

```
prometheus_targets:
```

```
  node:
```

```
    - targets:
```

```
      - "mail.example.com:9100"
```

```
  labels:
```

```
    env: production
```

```
    system: eshop
```

```
    service: mail
```

```
    server: c1
```

Prometheus: alert rules

Basic alert rules:

- Cpu Load
- Memory usage
- Disk usage
- HTTPS certificate expiration

The alerts are labeled based on severity:

- Information
- Warning
- Critical

Alertmanager: alerting strategy

alertmanager_child_routes:

- match:
 - severity: warning
 - receiver: warning
- match:
 - severity: critical
 - receiver: critical

alertmanager_inhibit_rules:

- target_match:
 - severity: warning
 - source_match:
 - severity: critical
 - equal: ['alertname', 'instance', 'target']

Alertmanager: receivers

```
alertmanager_receivers:  
  - name: warning  
    slack_configs:  
      - send_resolved: true  
        channel: '#prometheus-alerts'  
  - name: critical  
    slack_configs:  
      - send_resolved: true  
        channel: '#prometheus-alerts'  
    email_configs:  
      - send_resolved: true  
        to: "infra-alerts@example.com"  
    webhook_configs: #Telegram channel  
      - send_resolved: true  
        url: http://127.0.0.1:9087/alert/-001234567890
```

Benefits of Prometheus

Visibility

All configurations, of the stack, are stored in a git repository:

- Everyone that have access to the repository can view the configurations
- Pull request workflow for proposed modifications
- Versioning of the changes

Grafana can use LDAP as auth backend:


- Everyone with an account can access to the dashboards,

Local development: Vagrant



Thanks to the pull nature of Prometheus, almost every developer can locally reproduce the production environment:


1. Clone the repository
2. Use the Vagrantfile present in the in the repository to create e provisio a local instance
3. Experimenting and testing
4. Make a pull request with the changes

Social aspects


 **davidep** 9:47 AM ↪ 21
Commented on **AlertManager**'s message: **[FIRING:1] SSLCertExpiringSoon** https://mirrorlist.nethserver.com/?repo=updates&arch=x86_64&nsversion=7.5.1804&usetier=yes <https://mir...>

lo riavvio a mano
dovrebbe diventare green ora
ogni quanto esegue il controllo?


 **amygos** 9:50 AM
ogni 15 secondi
tra poco dovrebbe arrivare il messaggio di risoluzione
 1


 **AlertManager BOT** 9:52 AM

[RESOLVED] SSLCertExpiringSoon https://mirrorlist.nethserver.com/?repo=updates&arch=x86_64&nsversion=7.5.1804&usetier=yes
https://mirrorlist.nethserver.com/?repo=updates&arch=x86_64&nsversion=7.5.1804&usetier=yes (production metrics.nethesis.it http_2xx_check mirror information nethserver.com)

 **AlertManager BOT** 9:52 AM

[RESOLVED] SSLCertExpiringSoon <https://m1.nethserver.com/ping> <https://m1.nethserver.com/ping> (production metrics.nethesis.it http_2xx_check mirror information nethserver.com)

 **davidep** 9:53 AM
E' comodo discutere gli allarmi in questa chat... 😊

 **amygos** 10:00 AM ↪ 21
Commented on **AlertManager**'s message: **[FIRING:1] SSLCertExpiringSoon** https://mirrorlist.nethserver.com/?repo=updates&arch=x86_64&nsversion=7.5.1804&usetier=yes <https://mir...>

resolve_timeout: 3m

Cross companies remote debugging

The problem

One software, a big Java application, that we integrate in Netserver distribution, start to have some problems:

- Some Memory/Resource leak
- Not reproducible
- Not present in all installations

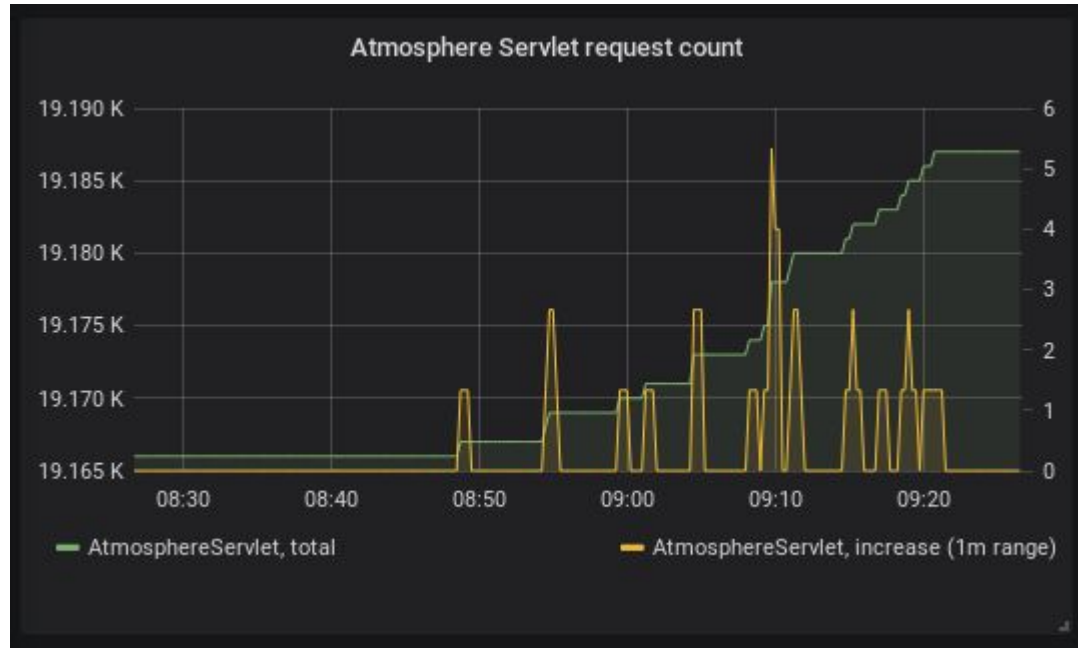
But lucky (or unlucky) the problems was presents in our local production installation

The solution

Thanks to Prometheus and Grafana stack the steps were pretty straightforward:

1. Install the JMX Exporter and configure it in the Prometheus's targets
2. Install the JMX Overview Grafana dashboard
3. Create the users in Grafana for the external developer team.
4. As plus, create a new Mattermost team for discussion and invite the external developers.
5. Have fun! (start debugging)

Custom panel



Grafana alerts



Beyond the metrics

The demo case

We have started to offer to our potential customer a Instance with our products installed as an evaluation demo, the instance must be valid for 30 days.

How can keep track of the expired instances?

1. Install the DigitalOcean exporter
 - a. Actually fork it and patch it for export the Droplet creation date as metric
2. Create the Ansible role for the setup
3. Configure an alert that when the expiration date is meet, an email will be sended to the sales department.

So Prometheus was also used by the sales :)

Conclusions

We have found Prometheus useful?

YES! :) We have found useful uses of Prometheus in many aspects of the company

- Operations
- Development
- Sales

Recommendations

1. Start simple
2. Use Prometheus stack as base
3. Make incremental steps
4. Don't overengineering

Questions?

Thanks for listening!

Who I am?

Matteo Valentini

Developer @ Nethesis (mostly Infrastrutture Developer)

 Amygos

 @_Amygos

 amygos@paranoici.org