



CASSIOPÉE PROJECT 2018-2019: DEVELOPMENT AND DEPLOYMENT OF AN AUTOMATED IT SECURITY AUDIT TOOL IN A VIRTUALIZED ENVIRONMENT.

Aurélien Duboc, Pierrick Gorisse, Lucas Martin

Supervisor: Hervé Debar

Contents

1	Start of the project	1
1.1	Objectives	2
1.2	Requirements	3
1.3	Expected results	7
1.4	Project Expectations	7
1.4.1	Specific competencies (in addition to PRO4501):	7
1.4.2	Learning objectives	7
2	Implementation of the project	8
2.1	Implementation of the project	8
2.2	Network Architecture	8
2.3	Application Architecture	9
2.4	Scoring and standardization	10
2.5	Results	11

1 Start of the project

Large companies as well as SMEs are subject to a security obligation for their information systems. The idea is to propose a tool that allows the management of the main vulnerabilities that security auditors usually look for. This tool will identify several weaknesses and / or configuration vulnerabilities.

The main interest lies in the automated analysis of a large number of machines. It could also propose automated corrections associated with these weaknesses. One could imagine that a tool like this one, to which other features would be added, could be a security audit equivalency for companies if this tool is accredited.

1.1 Objectives

- The establishment of an active infrastructure resulting from the use of various network services (hypervisor, access control, vpn, dns, reverse proxy, monitoring servers...) as well as the use of personal services and data: some users of this infrastructure host their web and ftp servers.
- The deployment of security audit tools for computer systems running Linux at least.
- The management of the logs associated with these audits.
- The development of a web interface allowing data management and visibility processed by the log management tool.
- The documentation in English for the deliverables and for Github in order to get an easier integration by the open source community.

1.2 Requirements

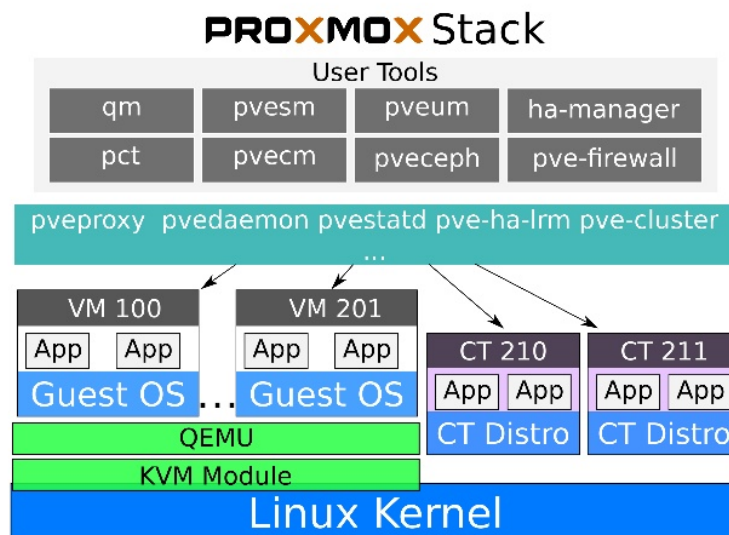
- Setting up a hypervisor for managing virtualized content:

The chosen solution is Proxmox, an open source hypervisor that can provide container based virtual by way of open VZ. It supports guest operating system like Linux (KVM), Windows. It is enabled by the presence of integrated backup service. It delivers full system virtualization by the use of KVM. The Proxmox management interface can function using a normal browser. Proxmox is using the cluster mode, from a single page multiple servers can be managed. and can perform a direct migration between one host to the other. Lastly, Proxmox provides shell access to the KVM directly from its interface, using a Debian system.

Proxmox VE tightly integrates KVM hypervisor and LXC containers, software-defined storage and networking functionality on a single platform, and easily manages high availability clusters and disaster recovery tools with the built-in web management interface.

You may sometimes encounter the term KVM (Kernel-based Virtual Machine). It means that Qemu is running with the support of the virtualization processor extensions, via the Linux KVM module. In the context of Proxmox VE Qemu and KVM can be used interchangeably as Qemu in Proxmox VE will always try to load the KVM module.

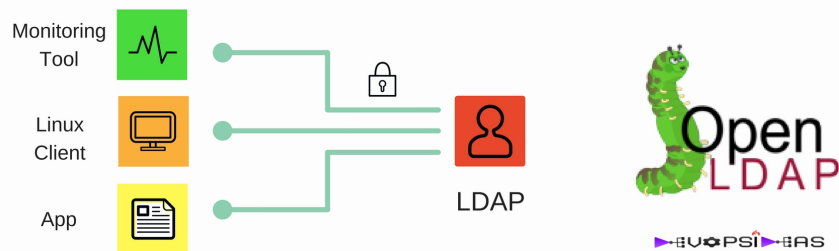
*<https://www.proxmox.com/en/>



- Setting up a directory to manage users and associated access control policies:

We are going to use an LDAP directory as specified in RFC 4510 and following. This allows a standardized representation of information (database - LDAP directory) as well as a standard query protocol widely deployed for this database. The chosen solution is the OpenLDAP software version 2.4.46, because of its maturity and protocol compliance. phpLDAPadmin is the web interface that allows the management of user accounts. one of the most important fields is sshPublicKey because all the servers are configured to do LDAP queries during an SSH connection to list the authorized RSA keys. We could have use a PAM setup with libpam-ldap but it seemed easier to specify in ssh configuration files an AuthorizedKeyCommand that will reach all the RSA public keys on the LDAP server.

*<https://ldap.com/basic-ldap-concepts/>

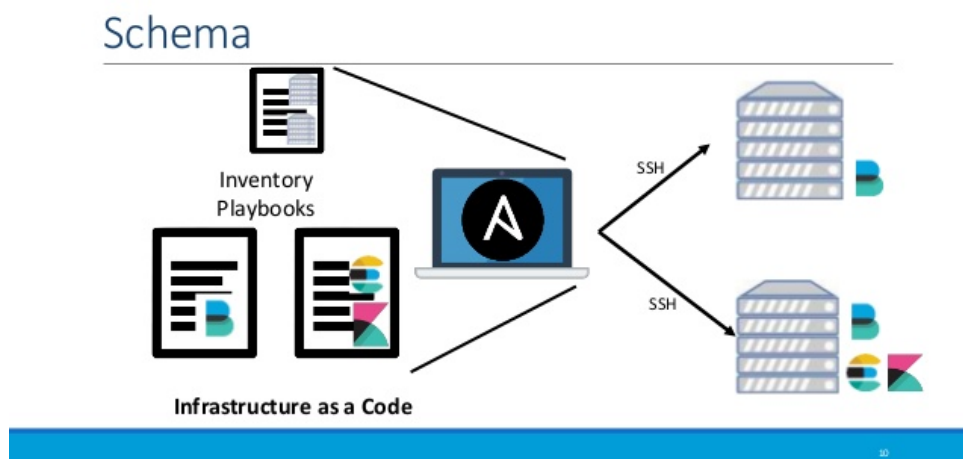


Installing phpLDAPadmin - Web based LDAP Client

- Use of a tool allowing access to all virtualized machines while using the access control protocol cited above:

We are going to use Ansible, an open source software that automates software provisioning, configuration management, and application deployment. Ansible connects via SSH, remote PowerShell or via other remote APIs.

* (<https://www.ansible.com/>)



- Implementation of a list of tools allowing the automated audits added to our personal contributions:

For now, Lynis seems to be the best candidate. We will probably combine several tools later. Lynis is an extensible security audit tool for computer systems running Linux, FreeBSD, macOS, OpenBSD, Solaris, and other Unix-derivatives.

Lynis scanning is opportunistic, meaning it will only use what it can find, like available tools or libraries. The benefit is that no installation of other tools is needed, so you can keep your systems clean. By using this scanning method, the tool can run with almost no dependencies. Also, the more it finds, the more extensive the audit will be. In other words: Lynis will always perform scans that are customized to your system and two audits will never be the same!

* (<https://cisofy.com/lynis/>)

```
[+] Software: firewalls
-----
- Checking iptables kernel module           [ NOT FOUND ]
  Status pf                                 [ NOT FOUND ]
- Checking host based firewall              [ NOT ACTIVE ]

[+] Kernel
-----
- Checking default run level...              [ RUNLEVEL 2 ]
- Checking CPU support (NX/PAE)
  CPU support: PAE and/or NoeXecute supported [ FOUND ]
- Checking kernel version and release        [ DONE ]
- Checking kernel type                       [ DONE ]
- Checking loaded kernel modules             [ DONE ]
  Found 44 active modules
- Checking Linux kernel configuration file... [ FOUND ]
- Checking for available kernel update...    [ OK ]
- Checking core dumps configuration...        [ DISABLED ]
  - Checking setuid core dumps configuration... [ PROTECTED ]

[+] Custom Tests
-----
- Running custom tests...                    [ SKIPPED ]

=====

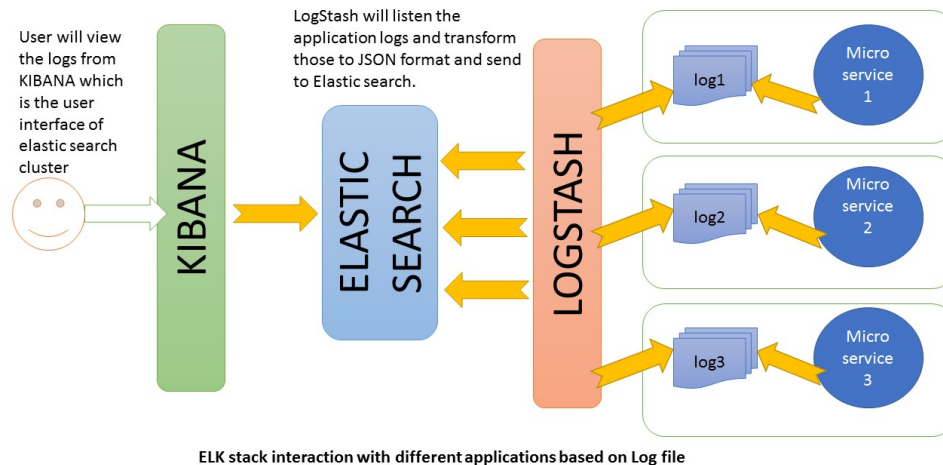
-[ Lynis 1.3.8 Results ]-

Tests performed: 12
```

- Management of the logs:

ELK is the most famous tool for log processing and analysis, so naturally we will use this tool to generate our logs. ELK is the acronym for three open source projects: Elasticsearch, Logstash, and Kibana. Elasticsearch is a search and analytics engine. Logstash is a server-side data processing pipeline that ingests data from multiple sources simultaneously, transforms it, and then sends it to a stash like Elasticsearch. Kibana lets users visualize data with charts and graphs in Elasticsearch. The Elastic Stack is the next evolution of the ELK Stack.

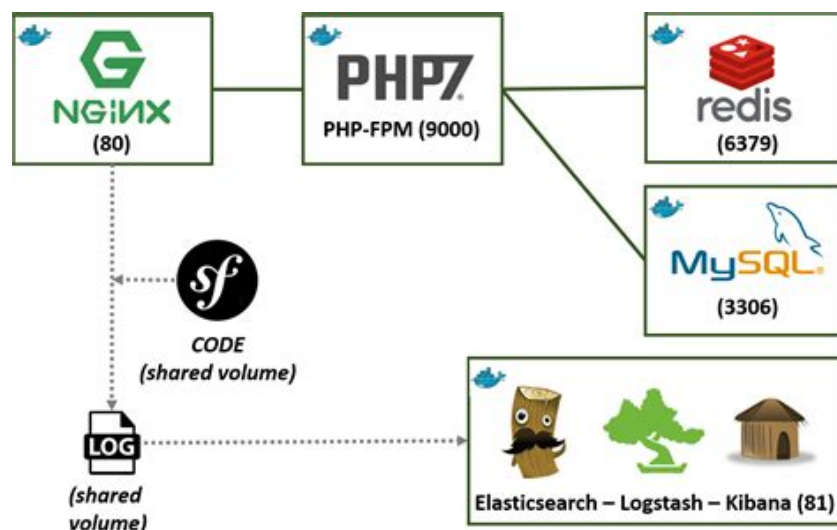
* (<https://www.elastic.co/fr/elk-stack>)



- Web interface development:

The use of a framework associated with a certain number of libraries will allow us to obtain a modular and easy to use application. Symfony is a PHP framework that we used to manipulate, which is why we chose to use this one. Moreover, we were able to verify that there were many libraries usable by this framework allowing us to interface ELK with our web application.

* (<https://pehpkari.cz/blog/2017/10/22/connecting-monolog-with-ELK/>)



1.3 Expected results

- A live demonstration is an option, otherwise, we will record a video of our tool performing an automated audit simulation.
- The presentation of the audit log management and administration platform generated by the tool
- To provide a free and open source tool with clear and explicit documentation allowing the redeployment of this tool in a virtualized environment.

We are still looking for other features.

1.4 Project Expectations

The use of English for our project is primordial. Indeed, all the technical documentation associated with the resources used is in English and we are used to working with resources in English because they are much more complete. In addition, the community providing these resources exchanges mainly in English. In order to offer a tool that is widespread and easy to use, writing the documentation in English then appears as a better choice.

Group size: 3 to 4 students.

1.4.1 Specific competencies (in addition to PRO4501):

UNIX-like platforms (Linux, MacOS) and associated tools, including software development tools (editors, interpreters, etc.). We are working on a server without a graphical environment so a text editor like VIM should be optimized as an IDE with some plugins in order to increase our efficiency on the server side, on the development and deployment of the tool. Coding in JavaScript and / or scripting languages, including referenced frameworks for JavaScript development (e.g. Node.js, Angular.js, etc.) As Elasticsearch and Kibana are working as APIs, the frontend development part should work as a REST API too.

1.4.2 Learning objectives

Processing of accessible textual data (coherence management, etc.) , the presentation and analysis of textual data as well as the understanding of cybersecurity software and integration.

Contact: herve.debar@telecom-sudparis.eu

2 Implementation of the project

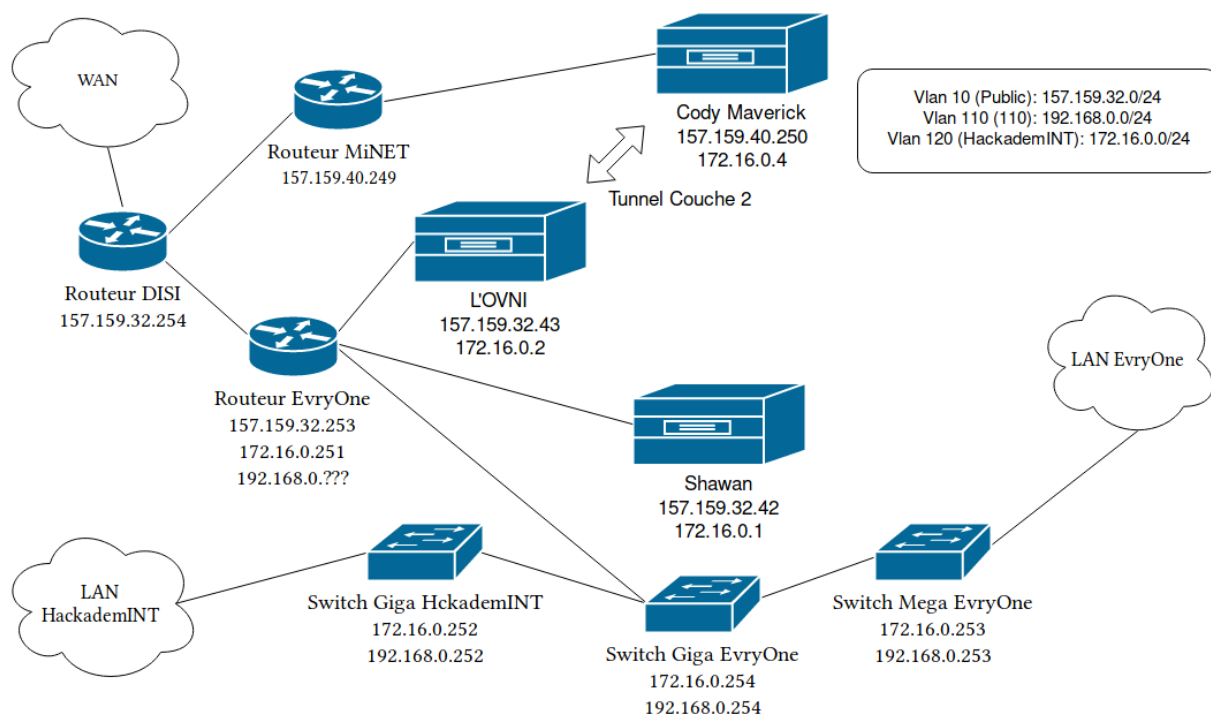
2.1 Implementation of the project

Flask framework imposed after further study of features necessary techniques for the smooth running of the project. The use of Proxmoxer (<https://github.com/swayf/proxmoxer>) was one of the reasons main movement to a python framework.

Proxmoxer is a wrapper around the Proxmox REST API v2. It was inspired by slumber, but it dedicated only to Proxmox. It allows to use not only REST API over HTTPS, but the same api over ssh and pvesh utility. Like Proxmoxia it dynamically creates attributes which responds to the attributes you've attempted to reach.

```
from proxmoxer import ProxmoxAPI
proxmox = ProxmoxAPI('proxmox_host', user='proxmox_admin', backend='ssh_paramiko')
for node in proxmox.nodes.get():
    for vm in proxmox.nodes(node['node']).openvz.get():
        print "{0}._{1} => {2}".format(vm['vmid'], vm['name'], vm['status'])
```

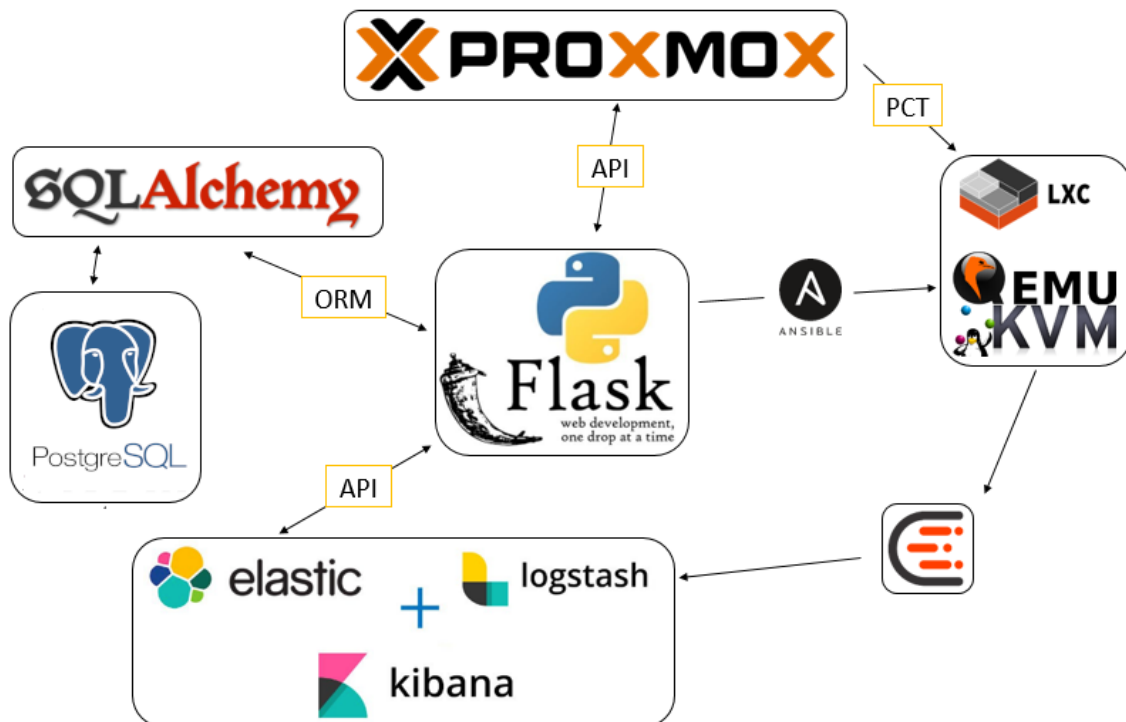
2.2 Network Architecture



2.3 Application Architecture

Here is the detail of the architecture as currently designed for this project:

- Several physical servers (whose names are "Cody-Maverick", "L'OVNI" and "Shawan") have been clustered on which are deployed the Proxmox hypervisor.
- Flask application dialog with this hypervisor through its API, we so use proxmoxer as a wrapper to interface with it.
- It is possible to access containers and virtual machines through pct (Tool to manage Linux Container (LXC) on Proxmox VE) and qemu (Qemu / KVM Virtual Machine Manager).
- To connect to different machines in the infrastructure, the application knows the IP addresses of the different machines thanks to the API of the hypervisor and RSA private keys must be filled in by the administrator system within the application to allow connection to different machines.
- By using the SSH network protocol, the Ansible tool helps with the execution of Lynis security audit tool on the different in order to get a log file which will then parser and embed in the ELK stack, itself interfaced with the application. We thus obtain a complete report of the security of every machine within the application
- User management of the application is performed at a base level postgresql data that the application accesses via the SQLAlchemy ORM.




2.4 Scoring and standardization

The Common Vulnerability Scoring System (CVSS) is a free and open industry standard for assessing the severity of computer system security vulnerabilities. CVSS attempts to assign severity scores to vulnerabilities, allowing responders to prioritize responses and resources according to threat. Scores are calculated based on a formula that depends on several metrics that approximate ease of exploit and the impact of exploit. Scores range from 0 to 10, with 10 being the most severe. While many utilize only the CVSS Base score for determining severity, temporal and environmental scores also exist, to factor in availability of mitigations and how widespread vulnerable systems are within an organization, respectively.

With Lynis, many tests are part of common security guidelines and standards, with on top additional security tests. After the scan a report will be displayed with all discovered findings. Our application will map these vulnerabilities with a CVSS Base, Temporal and Environmental Score

Common Vulnerability Scoring System (CVSS-SIG)

- CVSS v3.0 Calculator
- CVSS v3.0 Specification Document
- CVSS v3.0 User Guide
- CVSS v3.0 Examples
- CVSS v3.0 Calculator Use & Design
- CVSS v2 Archive
- CVSS v1 Archive
- CVSS-SIG participants
- Scores and Calculators
- Identity & logo usage



Common Vulnerability Scoring System Version 3.0 Calculator

Hover over metric group names, metric names and metric values for a summary of the information in the official CVSS v3.0 Specification Document. The Specification is available in the list of links on the left, along with a User Guide providing additional scoring guidance, an Examples document of scored vulnerabilities, and notes on using this calculator (including its design and an XML representation for CVSS v3.0).

Base Score

3.4 (Low)

Attack Vector (AV)

Network (N)
Adjacent (A)
Local (L)
Physical (P)

Attack Complexity (AC)

Low (L)
High (H)

Privileges Required (PR)

None (N)
Low (L)
High (H)

User Interaction (UI)

None (N)
Required (R)

Scope (S)

Unchanged (U)
Changed (C)

Confidentiality (C)

None (N)
Low (L)
High (H)

Integrity (I)

None (N)
Low (L)
High (H)

Availability (A)

None (N)
Low (L)
High (H)

Vector String - CVSS:3.0/AV:A/AC:H/PR:L/UI:R/S:C/L:LL/AN:C/M:TM/MA:MW/MA:MAC/L:MP/L:AU:N/MA:J:U/ML

Temporal Score

3.4 (Low)

Exploit Code Maturity (E)

Not Defined (X)
Unproven (U)
Proof-of-Concept (P)
Functional (F)
High (H)

Remediation Level (RL)

Not Defined (X)
Official Fix (O)
Temporary Fix (T)
Workaround (W)
Unavailable (U)

Report Confidence (RC)

Not Defined (X)
Unknown (U)
Reasonable (R)
Confirmed (C)

2.5 Results

CASSIOPEE
HACKADEMINT

- Dashboard
- CUSTOM SETTINGS
- Settings
- Flask-Admin panel

Containers and Virtual Machines

Show entries

id	name	status	node
lxc/100	reserved	status: stopped	hackademint
lxc/101	reserved	status: stopped	hackademint
lxc/102	ldap	status: running	hackademint
lxc/104	dns	status: running	hackademint
lxc/105	ssllh	status: running	hackademint
lxc/106	revproxy	status: running	hackademint
lxc/107	guacamole	status: stopped	hackademint
lxc/113	Moog-cassiop	status: stopped	hackademint
lxc/114	discordbot	status: running	hackademint
lxc/115	ftp	status: stopped	hackademint
id	name	status	node

CASSIOPEE
HACKADEMINT

- Dashboard
- CUSTOM SETTINGS
- Settings
- Flask-Admin panel

LXC 104: dns | Node: hackademint | CONFIG

JSON: [{"dns": [], "data": [{"cores": 1, "swap": 512, "onboot": 1, "hostname": "dns", "digest": "043b75160840b402a86f6baf8ce4b31c6de21521", "ostype": "debian", "memory": 512, "arch": "amd64", "rootfs": "stockage:104/vm-104-disk-0.raw,size=8G"}, {"interfaces": "OrderedDict([{'net0': {'hwaddr': 'E2:20:9B:1D:D3:7A', 'gw': '10.10.10.1', 'type': 'veth', 'bridge': 'vmbr0', 'name': 'eth0', 'ip': '10.10.10.4/24'}}, {'net1': {'hwaddr': '22:1B:33:AB:0F:AF', 'gw': '10.20.17.1', 'type': 'veth', 'bridge': 'vmbr17', 'name': 'eth17', 'ip': '10.20.17.4/24'}}, {'net2': {'hwaddr': 'FE:92:65:28:A6:A2', 'gw': '10.20.18.1', 'type': 'veth', 'bridge': 'vmbr18', 'name': 'eth18', 'ip': '10.20.18.4/24'}}, {'net3': {'hwaddr': '76:89:34:F0:17:2B', 'gw': '10.20.19.1', 'type': 'veth', 'bridge': 'vmbr19', 'name': 'eth19', 'ip': '10.20.19.4/24'}}, {'net4': {'hwaddr': '16:B8:C1:BA:91:59', 'gw': '10.0.33.1', 'type': 'veth', 'bridge': 'vmbr0', 'name': 'eth33', 'ip': '10.0.33.4/16'}}])"]}]

Network interfaces

name	bridge	ip	gateway	hwaddr	type
eth0	vmbr0	10.10.10.4/24	10.10.10.1	E2:20:9B:1D:D3:7A	veth
eth17	vmbr17	10.20.17.4/24	10.20.17.1	22:1B:33:AB:0F:AF	veth
eth18	vmbr18	10.20.18.4/24	10.20.18.1	FE:92:65:28:A6:A2	veth
eth19	vmbr19	10.20.19.4/24	10.20.19.1	76:89:34:F0:17:2B	veth
eth33	vmbr0	10.0.33.4/16	10.0.33.1	16:B8:C1:BA:91:59	veth

Config data

ostype: debian	arch: amd64	rootfs: stockage:104/vm-104-disk-0.raw,size=8G
cores: 1	memory: 512G	swap: 512G

