

Rapport YDays du 21/10/2020

Addario
Clément
B1 Info

Découverte Root-me.org

Challenge réseau

FTP – AUTHENTICATION

Énoncé

Un échange authentifié de fichier réalisé grâce au protocole FTP. Retrouvez le mot de passe utilisé par l'utilisateur.

Démarrer le challenge

1 ressource(s) associée(s)

■ rfc959 (RFC)

Validation

Entrer le mot de passe

envoyer

Fichier Editer Vue Aller Capture Analyser Statistiques Telephonie Wireless Outils Aide										
Apply a display filter ... <Ctrl-/>										
No.	Time	Source	Destination	Protocol	Length	Info				
1	0.000000	10.20.144.150	10.20.144.151	TCP	74	35974 → 21 [SYN] Seq=0 Win=3264				
2	0.000320	10.20.144.151	10.20.144.150	TCP	78	21 → 35974 [SYN, ACK] Seq=0 Ack=				
3	0.000570	10.20.144.150	10.20.144.151	TCP	66	35974 → 21 [ACK] Seq=1 Ack=1 W:				
4	0.000630	10.20.144.151	10.20.144.150	FTP	106	Response: 220-QTCP at fran.csg.				
5	0.275440	10.20.144.150	10.20.144.151	TCP	66	35974 → 21 [ACK] Seq=1 Ack=37 l				
6	0.275760	10.20.144.151	10.20.144.150	FTP	126	Response: 220 Connection will c				
7	0.276140	10.20.144.150	10.20.144.151	TCP	66	35974 → 21 [ACK] Seq=1 Ack=93 l				
8	4.216600	10.20.144.150	10.20.144.151	FTP	81	Request: USER cdt3500				
9	4.217350	10.20.144.151	10.20.144.150	FTP	91	Response: 331 Enter password.				
10	4.217630	10.20.144.150	10.20.144.151	TCP	66	35974 → 21 [PSH, ACK] Seq=16 Ac				
11	7.639420	10.20.144.150	10.20.144.151	FTP	81	Request: PASS cdt3500				
12	7.843260	10.20.144.151	10.20.144.150	TCP	70	21 → 35974 [PSH, ACK] Seq=114 /				
13	8.184000	10.20.144.151	10.20.144.150	FTP	95	Response: 230 CDT3500 logged c				
14	8.184360	10.20.144.150	10.20.144.151	TCP	66	35974 → 21 [PSH, ACK] Seq=31 Ac				
15	8.185040	10.20.144.150	10.20.144.151	FTP	72	Request: SYST				
16	8.185260	10.20.144.151	10.20.144.150	TCP	70	21 → 35974 [PSH, ACK] Seq=139 /				
17	8.192750	10.20.144.151	10.20.144.150	FTP	147	Response: 215 OS/400 is the re				
> Frame 11: 81 bytes on wire (648 bits), 81 bytes captured (648 bits)										
> Ethernet II, Src: IbmRisc6_9c:14:fe (00:06:29:9c:14:fe), Dst: IbmRisc6_9c:14:ae (00:06:29:9c:14:ae)										
> Internet Protocol Version 4, Src: 10.20.144.150, Dst: 10.20.144.151										
> Transmission Control Protocol, Src Port: 35974, Dst Port: 21, Seq: 16, Ack: 114, Len: 15										
> File Transfer Protocol (FTP)										
0000	00 06 29 9c 14 ae 00 06	29 9c 14 fe 08 00 45 00	..).....).....E.							
0010	00 43 2d 76 40 00 40 06	d7 e9 0a 14 90 96 0a 14	.C-v@. @.							
0020	90 97 8c 86 00 15 01 c1	b9 c6 60 b5 3f 16 80 18?....							
0030	7f 88 bb 15 00 00 01 01	08 0a 62 cc 7b 00 62 c9b{.b.							
0040	d3 50 50 41 53 53 20 63	64 74 73 33 35 30 30 0d	.PPASS c dts3500.							
0050	0a		.							

On devait ici, trouver un mot de passe dans un échange FTP, grâce à WireShark, on a réussi à le trouver.

C'était le premier exercice pour nous initier à l'application.

TELNET – AUTHENTICATION

Énoncé

Retrouvez le mot de passe de l'utilisateur dans cette capture réseau de session TELNET.

55	9.403099	192.168.0.2	192.168.0.1	TCP	66 1254 → 23 [ACK] Seq=2
56	9.464208	192.168.0.1	192.168.0.2	TELNET	75 Telnet Data ...
57	9.483049	192.168.0.2	192.168.0.1	TCP	66 1254 → 23 [ACK] Seq=2
58	10.704378	192.168.0.2	192.168.0.1	TELNET	67 Telnet Data ...
59	10.705366	192.168.0.1	192.168.0.2	TCP	66 23 → 1254 [ACK] Seq=2

> Ethernet II, Src: WesternD_9f:a0:97 (00:00:c0:9f:a0:97), Dst: Lite-OnU_3b:bf:fa (00:a0:cc:3b:bf:fa)
> Internet Protocol Version 4, Src: 192.168.0.1, Dst: 192.168.0.2
> Transmission Control Protocol, Src Port: 23, Dst Port: 1254, Seq: 152, Ack: 210, Len: 9
▼ Telnet
Data: Password:

▼ Telnet	Telnet	▼ Telnet	▼ Telnet
Data: u	Data: s	Data: e	Data: r

Dans ce cas, nous devons également retrouver un mot de passe dans une authentification TELNET, on trouvait les éléments dans chaque ligne « TELNET » après la data « Password ».

Énoncé

Retrouvez les données normalement confidentielles contenues dans cette trame :

```
00 05 73 a0 00 00 e0 69 95 d8 5a 13 86 dd 60 00
00 00 00 9b 06 40 26 07 53 00 00 60 2a bc 00 00
00 00 ba de c0 de 20 01 41 d0 00 02 42 33 00 00
00 00 00 00 00 04 96 74 00 50 bc ea 7d b8 00 c1
d7 03 80 18 00 e1 cf a0 00 00 01 01 08 0a 09 3e
69 b9 17 a1 7e d3 47 45 54 20 2f 20 48 54 54 50
2f 31 2e 31 0d 0a 41 75 74 68 6f 72 69 7a 61 74
69 6f 6e 3a 20 42 61 73 69 63 20 59 32 39 75 5a
6d 6b 36 5a 47 56 75 64 47 6c 68 62 41 3d 3d 0d
0a 55 73 65 72 2d 41 67 65 6e 74 3a 20 49 6e 73
61 6e 65 42 72 6f 77 73 65 72 0d 0a 48 6f 73 74
3a 20 77 77 77 2e 6d 79 69 70 76 36 2e 6f 72 67
0d 0a 41 63 63 65 70 74 3a 20 2a 2f 2a 0d 0a 0d
0a
```

On a ici, une trame codée en Hexadécimal, en le passant en ASCII, on obtient ce resultat :

Paste hex numbers or drop file

6d 6b 36 5a 47 56 75 64 47 6c 68 62 41 3d 3d 0d
0a 55 73 65 72 2d 41 67 65 6e 74 3a 20 49 6e 73
61 6e 65 42 72 6f 77 73 65 72 0d 0a 48 6f 73 74
3a 20 77 77 77 2e 6d 79 69 70 76 36 2e 6f 72 67
0d 0a 41 63 63 65 70 74 3a 20 2a 2f 2a 0d 0a 0d
0a

Character encoding

ASCII

Convert Reset Swap

0s ài00Z00Y` 00@&05 `*% 0pÀp 0AD 0B3 00t P%è}
, Áx000 áİ 000
>i¹0;~ÓGET / HTTP/1.1
Authorization: Basic Y29uZmk6ZGVudGlhbA==
User-Agent: InsaneBrowser
Host: www.myipv6.org

On obtient “Authorization: Basic Y29uZmk6ZGVudGlhbA==”, qui est codé en Base64. En le décodant

on obtient : **confi:dential**, qui est le mot de passe que l’on cherchait !

AUTHENTIFICATION TWITTER

Énoncé

Une session d'authentification twitter a été capturée. Retrouvez le mot de passe de l'utilisateur dans cette capture réseau.

```
keep-alive\r\n
Authorization: Basic dXNlc3Q6cGFzc3dvcmQ=\r\n
Credentials: usertest:password
Connection: keep-alive\r\n
```

Dans ce cas-ci, nous devons trouver le mot de passe d'un compte Twitter.

Wireshark a directement décodé le nom d'utilisateur ainsi que le mot de pas « password »

BLUETOOTH – FICHIER INCONNU

Énoncé

Votre ami travaillant à l'ANSSI a récupéré un fichier illisible dans l'ordi d'un hacker. Tout ce qu'il sait est que cela provient d'un échange entre un ordinateur et un téléphone. A vous d'en apprendre le plus possible sur ce téléphone.

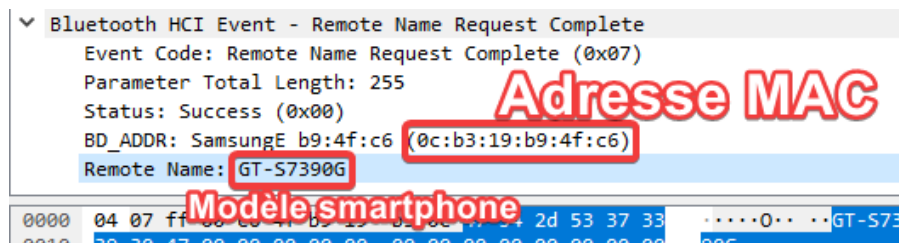
La réponse est le hash SHA1 de la concaténation de l'adresse MAC (en majuscules) et du nom du téléphone.

Exemple :

```
AB:CD:EF:12:34:56monTelephone -> 836eca0d42f34291c5fe91010873008b53c129
```

Dans cet exercice, on devait trouver dans une trame l'adresse MAC ainsi que le modèle du smartphone utilisé pour le transfert de données bluetooth.

Une fois tout récupéré, nous devons le passer dans un hacheur SHA1 pour avoir la réponse à l'exercice.



Voici ce que nous devons hasher : 0C :B3 :19 :B9 :4F :C6GT-S7390G

Résultat du hash

Le hash de votre texte est :



```
c1d0349c153ed96fe2fadf44e880aef9e69c122b
```

CISCO – MOT DE PASSE

Énoncé

Trouvez le mot de passe "Enable".

On avait ici un fichier .txt qui nous donnait plusieurs mot de passe pour plusieurs utilisateurs. Seulement, ces mots de passe sont en type 7 sur le cryptage de cisco, nous devons passer ces mots de passe dans un décrypteur.

On avait donc :

admin:6sK0_admin

guest :6sK0_guest

hub:6sKo_hub

On pouvait donc en déduire, pour l'utilisateur enable que le mot de passe était 6sK0_enable.

```
username hub password 7 025017705B3907344E
username admin privilege 15 password 7 10181A325528130F010D24
username guest password 7 124F163C42340B112F3830
,
```

Type 7 Password:

Plain text:

SIP -AUTHENTICATION

Énoncé

Retrouvez le mot de passe utilisé pour s'authentifier sur l'infrastructure SIP.

Démarrer le challenge

Dans cet exercice, on nous donnait un fichier .txt avec plusieurs informations.

```
172.25.105.3"172.25.105.40"555"asterisk"REGISTER"sip:172.25.105.40"4787f7ce""""PLAIN"1234
172.25.105.3"172.25.105.40"555"asterisk"INVITE"sip:1000@172.25.105.40"70fbfdæ""""MD5"aa533f6efa2b2abac675c1ee6cbde327
172.25.105.3"172.25.105.40"555"asterisk"BYE"sip:1000@172.25.105.40"70fbfdæ""""MD5"0b306e9db1f819dd824acf3227b60e07
```

Il y avait beaucoup d'informations inutiles.

Le but était de trouver un mot de passe.

On avait 2 mots de passe en MD5 et 1 en PLAIN

En testant le mot de passe en type PLAIN qui était 1234, nous avons réussi à passer cet exercice

```
3"4787f7ce""""PLAIN"1234
5.40"70fbfdæ""""MD5"aa533f6e1
```

Challenge Web-Client

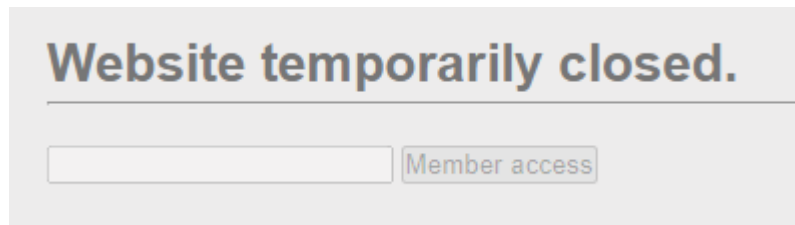
HTML- BOUTONS DESACTIVES

Énoncé

Ce formulaire est désactivé et ne peut pas être utilisé. À vous de trouver le moyen de l'utiliser tout de même.

Démarrer le challenge

En démarrant ce challenge, on nous ammenait dans une page web classique. Seulement, tout était bloqué !

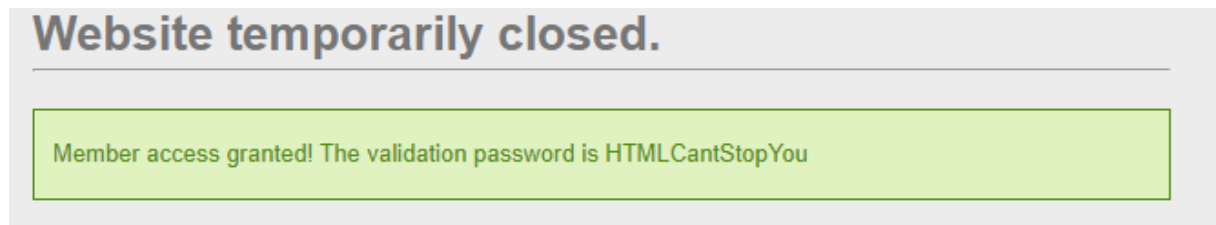


The screenshot shows a web page with a light gray background. At the top, the text "Website temporarily closed." is displayed in a large, bold, dark font. Below this text is a horizontal line. Under the line, there is a disabled login form. It consists of a text input field on the left and a submit button on the right. The submit button has the text "Member access" and is disabled, indicated by a light gray background and a border.

Voici la partie du code qui nous intéresse

```
▼<div>
  <input disabled type="text" name="auth-login" value>
  <input disabled type="submit" value="Member access" name="authbutton">
</div>
```

Juste en remplaçant disable par enable, le site se débloquent !

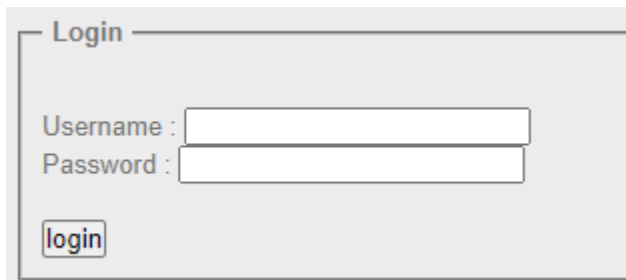


The screenshot shows the same web page as before, but with a success message. The text "Website temporarily closed." is still at the top. Below it, there is a green rectangular box with a black border containing the text "Member access granted! The validation password is HTMLCantStopYou".

Challenge Web-Client

JS - AUTHENTICATION

Nous n'avions pas d'énoncés mais seulement un site web avec un login.



En inspectant le code source, on peut trouver un fichier login.js

```
<script type="text/javascript" src="login.js"></script>
</head>
<body><link rel='stylesheet' property='stylesheet' id='s' type='text/css'
  <fieldset style="margin-top: 10px; padding: 10px;" width="60%">
  <legend><b>Login</b></legend><br/>
  <form name="login" method="POST" action="">
    Username : <input name="pseudo" /><br/>
    Password : <input name="password" /><br/>
    <input type="submit" value="login" />
  </form>
</body>
</html>

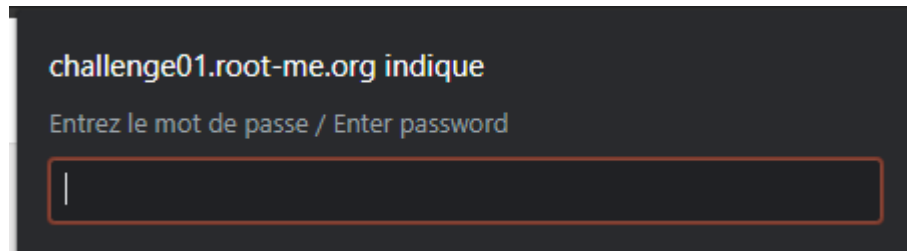
var pseudo=document.login.pseudo.value;
var username=pseudo.toLowerCase();
var password=document.login.password.value;
password=password.toLowerCase();
if (pseudo=="4dm1n" && password=="sh.org") {
  alert("Password accepte, vous pouvez valider le challe");
} else {
  alert("Mauvais mot de passe / wrong password");
}
/* ]]> */
```

Le mot de passe ainsi que le login était explicitement donné dans ce fichier !

JS - SOURCE

Ce challenge ne comporte toujours pas d'énoncés.

En commençant ce Challenge, on nous ramène sur une page avec une alerte qui nous demande un mot de passe, seulement nous avons aucune information.

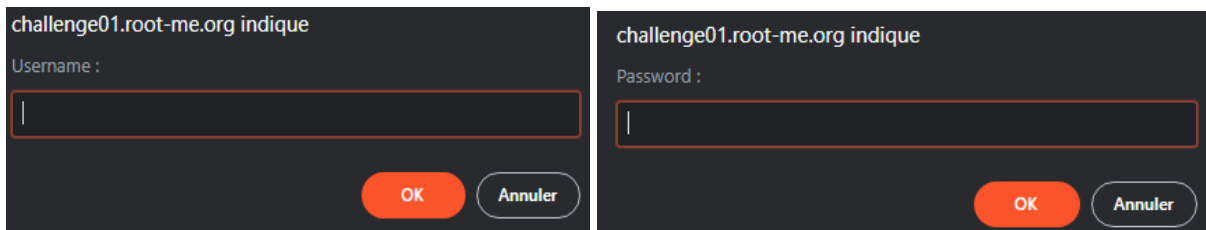


Pour continuer cet exercice, j'ai décidé de regarder le titre et on nous parle de source. En allant dans le code source, on nous donne encore une fois le mot de passe.

```
-----  
<script type="text/javascript">  
/*  */<br/>    function login(){<br/>        pass=prompt("Entrez le mot de passe / Enter password");<br/>        if ( pass == "123456azerty" ) {<br/>            alert("Mot de passe accepté, vous pouvez valider le challenge");<br/>        }<br/>        else {<br/>            alert("Mauvais mot de passe / wrong password !");<br/>        }<br/>    }<br/>/* ]]&gt; */<br/>&lt;/script&gt;<br/>&lt;/head&gt;</pre></div><div data-bbox="113 534 520 550" data-label="Text"><p>Une fois le mot de passe rentré, l'exercice est réussi !</p></div><div data-bbox="113 558 684 699" data-label="Form"><img alt="Screenshot of the challenge01.root-me.org success message. It shows a dark background with the text 'challenge01.root-me.org indique', 'Mot de passe accepté, vous pouvez valider le challenge avec ce mot de passe.', and 'You can validate the challenge using this password.' Below the text is an orange 'OK' button." data-bbox="113 558 684 699"/></div>
```

JS – AUTHENTIFICATION 2

Toujours aucun énoncés, mais dans le site qu'on nous a redirigé, on retrouve seulement un bouton login qui ouvre une alerte avec login et mot de passe. C'était donc à nous de les retrouver !



challenge01.root-me.org indique

Username :

challenge01.root-me.org indique

Password :

OK Annuler

OK Annuler

Toujours dans le même style, on retrouve dans le code source un fichier login.js

```
3 <title>JS Authentication</title>
4 <script language="JavaScript" src="login.js"></script>
5 </head>
6 <body><link rel='stylesheet' property='stylesheet' id='s'
7 </iframe>
8 <div id=EchoTopic>
9 </div></body></html>
```

En entrant dans ce fichier, on trouve plusieurs informations

```
function connexion(){
  var username = prompt("Username :", "");
  var password = prompt("Password :", "");
  var TheLists = ["GOD:HIDDEN"];
  for (i = 0; i < TheLists.length; i++)
  {
    if (TheLists[i].indexOf(username) == 0)
    {
      var TheSplit = TheLists[i].split(":");
      var TheUsername = TheSplit[0];
      var ThePassword = TheSplit[1];
      if (username == TheUsername && password == ThePassword)
      {
        alert("Vous pouvez utiliser ce mot de passe pour valider ce challenge (en majuscule:");
      }
    }
    else
    {
      alert("Nope, you're a naughty hacker.")
    }
  }
}
```

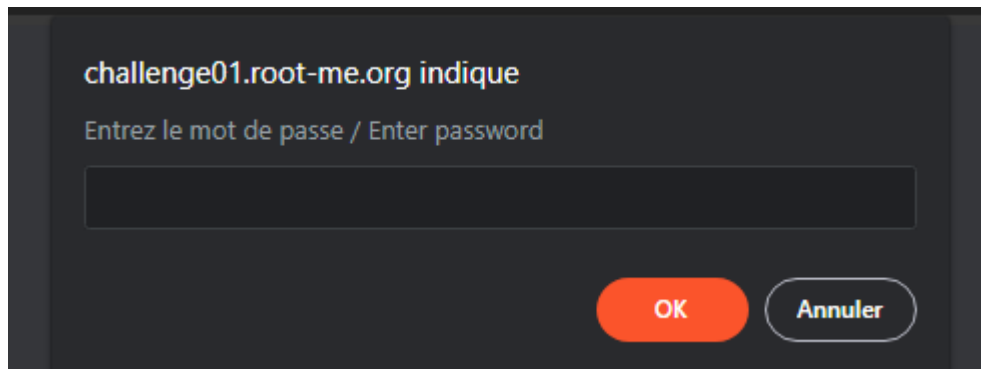
En regardant un peu le code, on peut voir que la variable TheLists contient le mot de passe ainsi que le nom d'utilisateur.

L'Username est donc GOD et le mot de passe est donc HIDDEN

L'exercice est résolu !

JS – OBFUSCATION 1

Toujours dans le même principe, aucun énoncé mais un mot de passe à retrouver.



challenge01.root-me.org indique

Entrez le mot de passe / Enter password

OK Annuler

Dans le code on retrouve des éléments pouvant nous faire penser au mot de passe.

```
<title>Obfuscation JS</title>
<script type="text/javascript">
  /*  */
  pass = '%63%70%61%73%62%69%65%6e%64%75%72%70%61%73%73%77%6f%72%64';
  h = window.prompt('Entrez le mot de passe / Enter password');
  if(h == unescape(pass)) {
    alert('Password accepte, vous pouvez valider le challenge avec ce mo');
  } else {
    alert('Mauvais mot de passe / wrong password');
  }
  // ...
</pre></div><div data-bbox="113 515 400 531" data-label="Text"><p>On a ici un mot de passe mais crypté.</p></div><div data-bbox="113 541 829 557" data-label="Text"><p>Si on passe le mot de passe dans un convertiseur Hexa à ASCII, on nous donne le mot de passe</p></div><div data-bbox="114 565 784 894" data-label="Form"><img alt="Screenshot of a hex-to-ASCII converter tool. The input field contains the hex string '%63%70%61%73%62%69%65%6e%64%75%72%70%61%73%73%77%6f%72%64'. The output field shows the decoded password 'cpasbiendurpassword'." data-bbox="114 565 784 894"/><p>Paste hex numbers or drop file</p><p>%63%70%61%73%62%69%65%6e%64%75%72%70%61%73%73%77%6f%72%64</p><p>Character encoding</p><p>ASCII</p><p>Convert Reset Swap</p><p>cpasbiendurpassword</p></div>
```

Conclusion

Ce premier jour de Ydays m'a permis d'apprendre dans un premier temps WireShark et aussi ça m'a permis de plus chercher par moi-même étant donné qu'aucune réponse n'est donné sur internet.