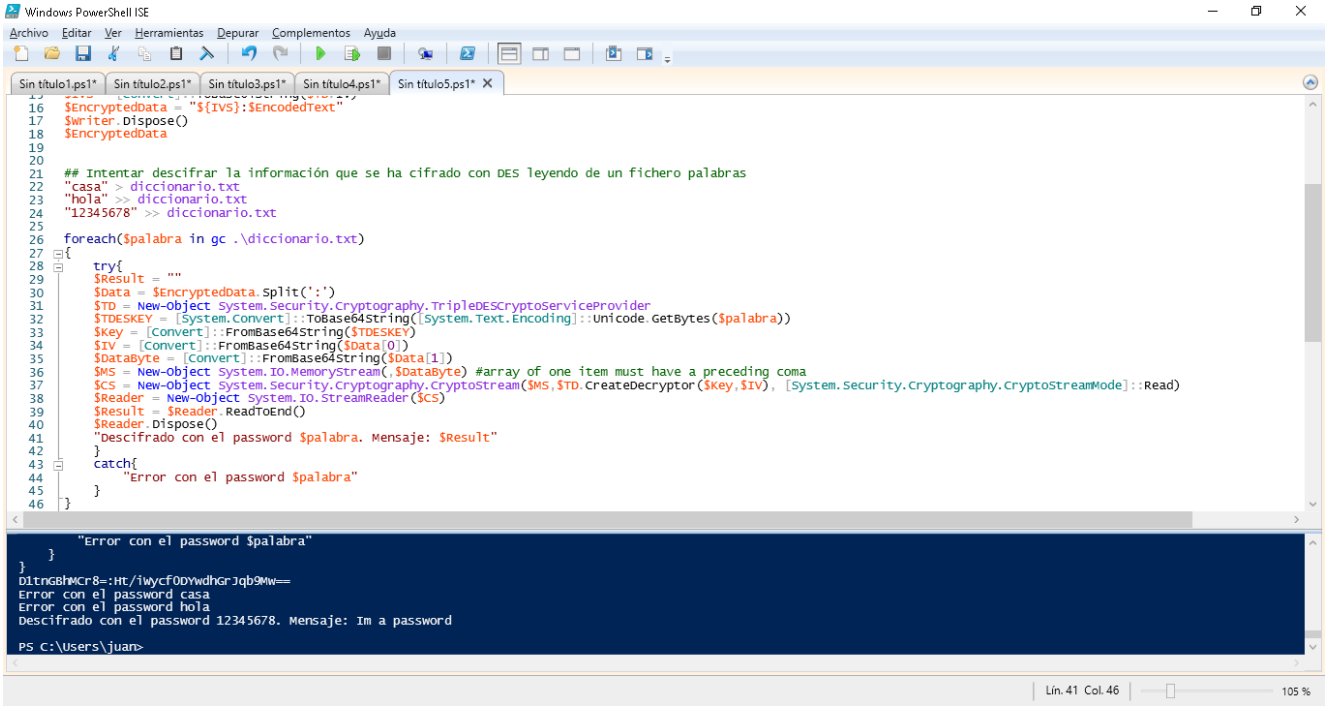
Descifrar un mensaje cifrado con Triple-DES comprobando contraseñas de un diccionario

En criptografía, Triple DES se le llama al algoritmo que hace triple cifrado del DES. También es conocido como TDES o 3DES, en 1998 IBM desarrollo 3DES o Triple DES (TDES) que sería sucesor directo de DES.

Cuando se descubrió que una clave de 56 bits no era suficiente para evitar un ataque de fuerza bruta, TDES fue elegido como forma de agrandar el largo de la clave sin necesidad de cambiar de algoritmo dé cifrado este método de cifrado es inmune al ataque por encuentro a medio camino, doblando la longitud efectiva de la clave (112 bits), pero en cambio es preciso triplicar el número de operaciones de cifrado, haciendo este método de cifrado muchísimo más seguro que el DES. Por tanto, la longitud de la clave usada será de 168 bits (3×56 bits), aunque como se ha dicho su eficacia solo sea de 112 bits. Se continúa cifrando bloques de 64 bits.

El Triple DES está desapareciendo, siendo reemplazado por el algoritmo AES. Sin embargo, la mayoría de las tarjetas de crédito y otros medios de pago electrónicos tienen como estándar el algoritmo Triple DES (anteriormente usaban el DES). Por su diseño, el DES y por lo tanto el TDES son algoritmos lentos. AES puede llegar a ser hasta 6 veces más rápido y hasta la fecha no se ha encontrado ninguna vulnerabilidad.

[crayon-6a9d581fa5cb0417096842/]


Windows PowerShell ISE
Archivo Editar Ver Herramientas Depurar Complementos Ayuda
Sin título1.ps1* Sin título2.ps1* Sin título3.ps1* Sin título4.ps1* Sin título5.ps1*
$EncryptedData = "${IVS}:$EncodedText"
$Writer.Dispose()
$EncryptedData
Intentar descifrar la información que se ha cifrado con DES leyendo de un fichero palabras
"casa" > diccionario.txt
"hola" >> diccionario.txt
"12345678" >> diccionario.txt
foreach($palabra in gc .\diccionario.txt)
{
try{
$Result = ""
$Data = $EncryptedData.Split(':')
$TD = New-Object System.Security.Cryptography.TripleDESCryptoServiceProvider
$TDESKEY = [System.Convert]::ToBase64String([System.Text.Encoding]::Unicode.GetBytes($palabra))
$Key = [Convert]::FromBase64String($TDESKEY)
$IV = [Convert]::FromBase64String($Data[0])
$DataByte = [Convert]::FromBase64String($Data[1])
$MS = New-Object System.IO.MemoryStream(,$DataByte) #array of one item must have a preceding coma
$CS = New-Object System.Security.Cryptography.CryptoStream($MS,$TD.CreateDecryptor($Key,$IV), [System.Security.Cryptography.CryptoStreamMode]::Read)
$Reader = New-Object System.IO.StreamReader($CS)
$Result = $Reader.ReadToEnd()
$Reader.Dispose()
"Descifrado con el password $palabra. Mensaje: $Result"
}
catch{
"Error con el password $palabra"
}
}
"Error con el password $palabra"
}
}
D1tnGBhMCr8=:Ht/iWycfODYwdhGrJqb9Mw==
Error con el password casa
Error con el password hola
Descifrado con el password 12345678. Mensaje: Im a password
PS C:\Users\juan>
Lín. 41 Col. 46
105 %