

Diffie-Hellman + OnetimePad

Kryptografische Verfahren: Diffie-Hellman-Schlüssel-Vereinbarung Hafendorn Okt 2011

erg = diff([127.56,10.77]) • $\begin{array}{l} \text{"Sie verabreden p = 127"} \\ \text{"Sie verabreden g = 56"} \\ \text{"Anton sendet " 30} \\ \text{"Berta sendet " 90} \\ \text{"Antons Schlüssel " 37} \\ \text{"Bertas Schlüssel " 37} \end{array}$ isPrime(erg[1,2]) • true

Diese Datei zeigt das **Diffie-Hellman-Verfahren**.

Die Umwandlung von Zahlen in Ziffernlisten und zurück

zah2li([192837465]) • {1,9,2,8,3,7,4,6,5} **li2zah({1,9,2,8,3,7,4,6,5})** • 192837465 und das **OneTimePad-Verfahren** für Ziffernlisten mit verschlüsseln und entschlüsseln.

onetimepad([{1,2,3,4,5,6,7},{1,9,2,8,3,7,4,6,5}]) • {2,1,5,2,8,3,1} **onetimeinv([{1,2,5,2,8,3,1},{1,9,2,8,3,7,4,6,5}])** • {1,2,3,4,5,6,7}

Man kann auch alles kombinieren:

diffiehell - diff(kryxnpempre(randint(1000000000,10000000000000000000),randint(10000000,10000000000000000000))
random(10000000000000000000) • randint(10000000,10000000000000000000))

* $\begin{array}{l} \text{"Sie verabreden p = 96166013137767"} \\ \text{"Sie verabreden g = 8671702491"} \\ \text{"Anton sendet " 40899640805182} \\ \text{"Berta sendet " 33737227662276} \\ \text{"Antons Schlüssel " 43662590709547} \\ \text{"Bertas Schlüssel " 43662590709547} \end{array}$

sli = diffiehell[3,2] • 43662590709547 sli = zah2li(s) • {4,3,6,6,2,5,9,0,7,0,9,5,4,7} **otp = onetimepad(zah2li([112233445566,sl]))** • {5,4,8,8,5,8,3,4,2,5,5,1} **otli = onetimeinv(otp,sli)** • {1,1,2,2,3,3,4,4,5,5,6,6} Ende: li2zah(otli) • 112233445566

Die Message 112233445566 kommt am Ende wieder heraus.

```

diffi[13,2,5,4] + [{"Sie verabreden p " : 13, "Sie verabreden g " : 2}, {"Anton sendet " : 6, "Bertha sendet " : 3}, {"Antons Schlüssel " : 9, "Bertas Schlüssel " : 9}]

diffi[123457,5678,1023,5001] + [{"Sie verabreden p " : 123457, "Sie verabreden g " : 5678}, {"Anton sendet " : 42332, "Bertha sendet " : 80229}, {"Antons Schlüssel " : 68591, "Bertas Schlüssel " : 68591}]

diffi[kry,nextprime] 7524738843686849139, {55665526616813868, 10248276867348633, 57657657001 }

+ [{"Sie verabreden p " : 7524738843686849139, "Sie verabreden g " : 55665526616813868}, {"Anton sendet " : 4812236671797693468, "Bertha sendet " : 8210090613061314686}, {"Antons Schlüssel " : 69377184492155382014, "Bertas Schlüssel " : 69377184492155382014}]

10^14 ist maximale Größe für randint, 10^14 funktioniert nicht als Eintrag (Fehler gemeldet Octet. 2011)
diffi[kry,nextprime] randint(10000000000,10000000000000000000) ]|randint(10000000,10000000000),
randint(10000000,10000000000) ]|randint(10000000,10000000000)

+ [{"Sie verabreden p " : 6758801582209, "Sie verabreden g " : 3849525901}, {"Anton sendet " : 6597997620719, "Bertha sendet " : 6331204062781}, {"Antons Schlüssel " : 6631609941997, "Bertas Schlüssel " : 6631609941997}]

Spielweise

erg = diffi[127,56,10,77] + [{"Sie verabreden p " : 127, "Sie verabreden g " : 56}, {"Anton sendet " : 30, "Bertha sendet " : 90}, {"Antons Schlüssel " : 37, "Bertas Schlüssel " : 37}]

isPrime(erg[1,2]) * true

```

1.1

1.2

<pre> • onetimepad Define LibPub onetimepad[mess,key]= Func © (message, key) -> {cryptogramm } © alias als Ziffernlisten Local df,k k:=key Loop df:=dim(mess)-dim(k) If df=0 Then Return mod(mess,k,10) Elseif df>0 Then k:=left(k,df) Elseif df<0 Then k:=augment(k,k) Endif EndLoop EndFunc </pre>	<p>8/14</p> <p>onetimepad • {2,5,0,3,4,8},{1,9,2,8,3,7,4} • {3,4,2,1,7,5}</p> <p>MessageLength=Schlüssellänge</p> <p>onetimeinv [{3,4,2,1,7,5},{1,9,2,8,3,7}] • {2,5,0,3,4,8}</p>
	<p>Wichtig für das Programm ist,</p> <p>mod(mess,k,10)</p> <p>Die Modulo-Funktionist "listable", sie arbeitet auf den Elementen der Liste einzeln.</p> <p>Ebenso kann man Listen elementweise mit + addieren.</p> <p>{1,2,3} + {10,20,30} • {11,22,33}</p> <p>mod({11,22,33},7) • {4,1,5}</p> <p>Die anderen Programmteile dienen dazu, die Schlüssellänge an die Länge der Message anzupassen.</p>
onetimepad • {2,5,0,3,4,8},{1,9,2,8,3,7,4,6} • {3,4,2,1,7,5}	Schlüssel länger
onetimeinv [{3,4,2,1,7,5},{1,9,2,8,3,7,4,6}] • {2,5,0,3,4,8}	
onetimepad • {2,5,0,3,4,8},{1,9,2} • {3,4,2,4,3,0}	Schlüssel kürzer
onetimeinv [{3,4,2,4,3,0},{1,9,2}] • {2,5,0,3,4,8}	

1.3

1.4

```

"onetimemeinv" erfolgreich gespeichert
Define LibPub onetimemeinv(cryp,kay)-
  Func
    ©(cryptogramm, key) ->message
    © alles als Ziffernlisten
    Local dif,k
    k:=key
    Loop
      df:=dim(cryp)-dim(k)
      If df=0 Then
        Return mod(cryp-k,10)
      Elseif df<0 Then
        k:=k(dim(k)+df)
      Elseif df>0 Then
        k:=augment(k,df)
      EndIf
    EndLoop
  EndFunc
}

onetimemeinv({3,4,2,1,7,5},{1,9,2,8,3,7}) • {2,5,0,3,4,8}
OneTimePadInvers Kryptogramm und Schlüssel als
Ziffernlisten.
onetimemeinv({3,4,2,1,7,5},{1,9,2,8,3,7,4,6}) • {2,5,0,3,4,8}
Schlüssel länger

```

```

zahlfli
Define LibPub zahlfli(zahl)=
Func
© zahl->Ziffernliste
Local i,z,h
h:={}{ }
z:=zahl
While z>0
h:=augment({{ mod(z,10)},h})
z:=floor(  $\frac{z}{10}$  )
EndWhile
Return h
EndFunc

zahlfli(112233445566 ) • { 1,1,2,2,3,3,4,4,5,5,6,6 }
li2zahfli({ 1,1,2,2,3,3,4,4,5,5,6,6 }) • 112233445566

```

1.5

1.6

```

li2zahl 0/8
Define LibPub li2zahl(hste)-
| Func
| © {Ziffernliste }-> Zahl
| Local z,h,i
| h:=hste; z:=0
| For i,1,dim(hste)
| z:=2*10-#h[1]
| h:=mid(h,2)
| EndFor
| Return z
| EndFunc

```

li2zahl{2,5,0,3,4,8} • 250348
zahlli(250348) • {2,5,0,3,4,8}