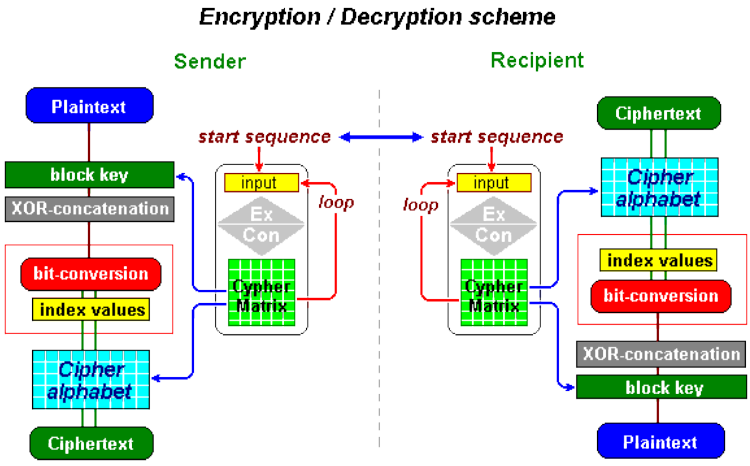
„Verschlüsseln mit Unicode"

(Ernst Erich Schnoor)

CypherMatrix® ist ein neues Verschlüsselungs-Verfahren, das vom Autor in mehrjähriger Arbeit in Python-Technik entwickelt worden ist [#1].



Das Verfahren arbeitet als Blockfolge. Der Klartext wird in definierten Blöcken seriell abgearbeitet. Das Verfahren verwendet einige Bereiche, die erst bei der Entwicklung des Verfahrens entdeckt worden sind. Die Bereiche sind bisher weder dokumentiert noch wissenschaftlich analysiert worden.

„crypto incognito"

Das sind insbesondere folgende Teilbereiche:

„Codepage" Technologie
digitaler „Matrix-Generator"
„one-time-chain"
„Bitkonversion"

„Codepage" Technologie

Im Computer werden Texte, Wörter und Zeichen durch Zahlen repräsentiert. Daher ist es notwendig, eine Zuordnung von Zahlen und Zeichen herzustellen. Der Computer benötigt eine Zuordnungstabelle, allgemein als **Codepage** bezeichnet [#2]. Dort wird verzeichnet, welche Zahl für welchen Buchstaben stehen soll. Diese Tabelle ist in beide Richtungen eindeutig, aus einem Buchstaben wird also exakt eine Zahl, und aus dieser Zahl wird umgekehrt wieder exakt dieser Buchstabe. Die Position eines Zeichens in der Tabelle und seine Codierung ist mit den Werten von 0 bis 255 (= 1 Byte) immer gleich. Diese Eigenschaften sind in das CypherMatrix Verfahren integriert.

Ende der 80er-Jahre entstand eine universielle Zeichentabelle für alle Sprachen der Welt: **Unicode** [#3]. Sie lassen sich digital darstellen und mit Zuordnung der binären Ziffern '0' bis '255' zu einer geeigneten Codpage gestalten [#4]**.** Die binären Zeichen werden zum Empfänger transferiert ('Schicht 1 OSI-

Modell') und dort mit der gleichen Codepage in die ursprünglichen Zeichen zurück- verwandelt.

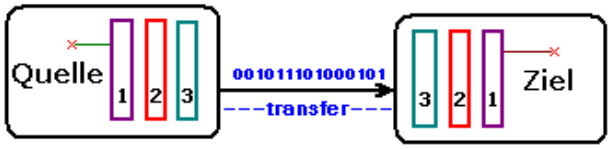
Vom Programm beispielsweise sind folgende 256 Unicode-Zeichen vorgegeben:

```
t1  = ΣΤΥΦΧΨΩΙΥάέήίϋα                                  # grichisch
t2  = ৫লঙজগচছঝঢ৩১৪মঢ৫৶৪৬৩৷৷ল                          # indisch
t3  = 뀨낲냬낄냶낔넣낱낳내끺냎끕냋뀾                        # koreanisch
t4  = અોઅોકખગઘઙચછજઝઞટઠડ                              # gujaratisch
t5  = じすずせぜそぞただちぢっつづてでとどなに            # japanisch
t6  = DEFGHIJKLMNPRSTUVabcdefghkprs                      # ascii-Code
t7  = ЖЗИИКЛМНОПРСТУФХЦЧШЩ                               # kyrillisch
t8  = 俵倕倩併倆俓俓側御栁倆傷俣俸俣俤俗悠做俾俥       # chinesisch
t9  = ጶጷጸጹጺጻጼጽጾጿፀፁፂፃፄ                              # ätiopisch
t10 = ฟภมยรฤลฦวศษสหฬ                                    # thailändisch
t11 = ჟუსტᲝᲠᲡᲢᲣᲤᲥᲦᲧᲨᲩᲪᲫᲬᲭᲮᲯᲰᲱᲲᲳᲴᲵᲶᲷᲸᲹᲺ᲻᲼Ჽ   # georgisch
t12 = ཁཀཅཆཇཉཏཐདདྷནཔཕཙ                               # tibetisch
t13 = ঔকখগঘঙচছজঝঞটঠডঢ                                 # bengalisch
t14 = ಪಫಬಭಮಯರಱಲಳವಶಷಸಹ                                 # kannadaisch
t15 = කඛගඝඩඡජඣඤටඨඩඪණඳඵබභමඹයරලවශෂසහළෆ          # singalesisch
```

Für die Verschlüsselung kommt hier der 'Matrix-Generator' ins Spiel [#5].

Digitaler „Matrix-Generator"

Der Generator hat die Aufgabe, den Verlauf des Verfahrens in allen Einzelheiten zu steuern.



1. Codepage (CypherMatrix)
2. XOR-Verknüpfung
3. Bit-Konversion

Der Generator generiert aus dem vorgegebenen Zeichensatz eine dynamische „Runden-Matrix" mit 256 Zeichen, aus der jeweils ein Runden-Schlüssel für die XOR-Verknüpfung mit dem Klartextblock und ein Matrix-Schlüssel als Startphrase für die nächste Runde erzeugt werden. Als Test kann das Programm: telecypher-Demo.py herunter geladen werden.

>teleCypher-Demo<
Verschlüsselungsprogramm
CypherMatrix-Verfahren

Auswahl: *Transfer ..E.. (europäische Zeichen)*
..G.. (globaler Unicode)
..I.. (indisch, bengalisch)

..q.. (beenden)

Transfer wählen > G
Startphrase: „schwarzer Holunder am Roten Main in der goldenen Abendsonne"
Umfang: 59 Zeichen

Wird für das Verfahren als Transfer: „globaler Unicode“ gewählt, generiert das Programm folgende „Codepage“ (256 Zeichen) [#4], hier als Runden-Matrix bezeichnet.

0	づ Ï 낱 ᧚ ८ᨷᨸᨹ ᧡ᨻᨽ° TLá ᨾ Ꮒ ᨶᨺᨻᨼ Π ᨿ OP 令 ᨽᨿ ᨿ に CᏀ	31
32	¸ K 냶 ´ ち ᨻ ᨿ αN 仟 s ᨸ U ᨾ 낯 ᨿ b ᨽ 任 ᨻ て mᨿ ήᨾ つ MИ せ f « ᨶ	63
64	ЖЩЦ ᨾ じ ᨾ 내 ᨾ 낊 ᨾ 全 ᨾ 낊 ᨶ ᨻ 仕 ᨾ b 꾲 TJ ᨾ ᨸᨾ- です ᨾ ᨾ ᨾ V	95
96	냎 i ᨸ ± rd だ 仜 そ な ᨾ r ᨿ Yc ᨾ 낳 μ 낳 ず 仚 Ω 끕 ᨸ 仗 ¸ ᨸ Π ᨸ 냮 ᨾ I	127
128	ᨿ Ÿ と ᨾ ᨸ ᨾ ᨾ r И 仔 ᨾ ϕ 꾹 仟 Ω ¬ ᨶ R ᨾ 佐 ᨾ ᨾ 넣 ᨾ ᨾ ᨾ 3 仓 ᨾ 仞 D ᨿ	159
160	仙 ᨾ ᨾ p ᨿ Y ᨾ ぞ ᨾ ᨾ e ᨾ E ぢ S ᨾ ᨾ G ² h ᨾ ᨾ ど ᨾ た H ᨿ 代 C ᨾ r ᨾ Ψ	191
192	付 ᨾ ᨾ ᨾ ᨾ ᨾ Φ ᨾ ぜ T ᨾ kg ᨾ K ᨾ ᨾ ᨾ ᨾ X ᨾ ᨾ ᨾ M ᨾ ᨾ ® 令 以 ᨾ ¶	223
224	aU ᨾ ᨾ ᨾ ᨾ P ᨾ ³ ᨾ ᨾ ¯ H ᨾ ᨾ ᨾ ¹ Σ ᨾ 仞 ᨾ 끲 ᨾ ᨾ X ᨾ F ¸ · 他 ¹	255

In das Programm „teleCypher-Demo.py“ werden als Klartext eingegeben:
„Jeden Morgen geht die Sonne auf und scheint den ganzen Tag so hell und klar in meine kleine Stube (nach: H.Claudius)“
Nach der Eingabe generiert das Programm den Runden-Schlüssel und den für die nächste Runde als Startphrase erforderlichen Matrix-Schlüssel wie folgt:

Runden-Schlüssel
ᨾ r ᨿ Yc ᨾ 낳 μ 낳 ず 仚 Ω 끕 ᨸ 仗 ¸ ᨸ Π ᨸ 냮 ᨾ I ᨿ Ÿ と ᨾ ᨸ ᨾ ᨾ r И 仔 ᨾ ϕ 꾹 仟 Ω ¬ ᨶ R ᨾ 佐 ᨾ ᨾ 넣 ᨾ ᨾ ᨾ 3 仓 ᨾ 仞 D ᨿ 仙 ᨾ ᨾ p ᨿ Y ᨾ ぞ ᨾ
Umfang: 63 Zeichen
Der Runden-Schlüssel wird für das folgende „one-time-pad“ benötigt. Aus den Bestimmungsfaktoren: Länge Schlüssel (63) und Menge Zeichen (256) ergibt sich ein Schlüsselraum von $256^{pow}63$ = 5.237e+163. In jeder Runde wird ein neuer Schlüssel erzeugt und kein früherer Schlüssel wird wiederholt.

Matrix-Schlüssel
I ᨿ Ÿ と ᨾ ᨸ ᨾ ᨾ r И 仔 ᨾ ϕ 꾹 仟 Ω ¬ ᨶ R ᨾ 佐 ᨾ ᨾ 넣 ᨾ ᨾ ᨾ 3 仓 ᨾ 仞 D ᨿ 仙 ᨾ ᨾ p ᨿ Y ᨾ ぞ ᨾ e ᨾ E ぢ S ᨾ ᨾ G ² h ᨾ ᨾ ど ᨾ た H ᨿ 代 C ᨾ r
Umfang: 63 Zeichen
Der Matrix-Schlüssel wird zur Initialisierung der nächsten Runde auf den Beginn der Schleife zurückgeführt (loop).

„one-time-chain“

Der zu verschlüsselnde Text wird in gleicher Länge mit einem aus der CypherMatrix entnommenen Schlüssel **XOR**-verknüpft [#6]. Das Ergebnis als Bitfolge holt mit den Zuordnungswerten der Elemente das entsprechende Zeichen aus der Runden-Matrix und verbindet es zur Folge der Bit-Konversion .

„Jeden Morgen geht die Sonne auf und scheint den ganzen Tag so h
(1. Block, Umfang: 63 Zeichen)

Runden-Schlüssel
ᨾ r ᨿ Yc ᨾ 낳 μ 낳 ず 仚 Ω 끕 ᨸ 仗 ¸ ᨸ Π ᨸ 냮 ᨾ I ᨿ Ÿ と ᨾ ᨸ ᨾ ᨾ r И 仔 ᨾ ϕ 꾹 仟 Ω ¬ ᨶ R ᨾ 佐 ᨾ ᨾ 넣 ᨾ ᨾ ᨾ 3 仓 ᨾ 仞 D ᨿ 仙 ᨾ ᨾ p ᨿ Y ᨾ ぞ ᨾ (Umfang: 63 Zeichen)

XOR --> one-time-pad:
¸ á □ ᨾ 仕 fC づ ᨾ ᨾ ᨾ □ に ᨾ á す P ᨾ ᨾ V ᨾ □ ¯ H □ ᨾ P Σ □ ¹ ᨾ ᨾ S · ¯ ᨾ 仞 F · a ど Σ Σ 끲 ᨾ · ᨾ Σ PF ¹ ᨾ ᨾ ᨾ ᨾ 山 ぜ r 付 (Umfang: 63 Zeichen)

Da Klartext und Schlüssel immer die gleiche Länge haben, entsteht ein partielles **„one-time.pad“** [#7]. Der Schlüssel wird auch nicht wiederholt. In jeder Runde wird ein anderer Schlüssel aus der jeweiligen Runden-Matrix entnom- men. Das ergibt für den gesamten Vorgang eine Kette zusammenhängender „one- time-pad“ Funktionen, gewissermaßen als eine **„one-time-chain“**.

Bit-Konversion

Bisher werden Umwandlungen von Bitfolgen nur im Verfahren „*Coding Base64*“ vorgenommen [#8]. Dabei werden Bytes im Bitsystem zur Basis 8 in eine Folge von 6-bit Sequenzen umgewandelt. Die dezimalen Werte dieser Sequenzen sind Indizes für ein statisches Chiffre-Alphabet von 64 Zeichen.

Bit-Konversion ist die Umwandlung einer Bitfolge von einem Bitsystem in ein anderes Bitsystem, im vorliegenden Verfahren von 8-bit in 7-bit. Dabei bleiben die Anzahl der Bits und ihre Reihenfolge gleich. Kein Bit wird hinzugefügt und kein Bit wird weggelassen. Nur die Anzahl der Bits in einer Einheit ändert sich. Die dezimalen Werte der neuen Einheiten sind Indexwerte für das zugeordnete Alphabet. Die Bit-Konversion von Basis 8 zur Basis 7 geschieht im Einzelnen wie folgt:

Bitfolge im Original:

011000100110100101110100011001100110111101101100011001100

98 105 116 102 111 108

Bitfolge nach Konversion:

011000100110100101110100011001100110111101101100011001100

49 26 46 70 51 61 88

Im vorliegenden Beispiel wird das Ergebnis der XOR-Verknüpfung (8-bit) in eine 7-Bitfolge umgewandelt. Die numerischen Zuordnungen holen die ent- sprechenden Zeichen aus der Runden-Matrix (Codepage) und verbinden sie zum Chiffretext.

XOR --> one-time-pad:

¸ ȧ□ □ゔ仕fC ゔ ចঌঀઓ□に ঌȧ す P ចঀV ភ□ ¯H□ УPΣ□ ¹ตว S · ¯」 仢F · a ど ΣΣ 뀔
ঝ· ฬΣPF ¹ঘঠ ฦঔ ঘШ ぜ r 付 (Umfang: 63 Zeichen)

Chiffretext (8-bit --> 7-bit):

ঙf ɲ ず넊 ঘY ចЩU ´ ศঔ৩৩ せ ŋ¸ ঙr ঘTФ 任 লক্ষn 냎 ল P□ づ仕 I 仜な ঔ 넊s ศศЖКЛ r ยέÏ 깞
r ลI ঌ 깗냎Ï ち I ず മЩU ɵ □ ঞ্ɥ ষছ なてず (Umfang: 72 Zeichen)

Entschlüsseln

Die Entschlüsselung geschieht in der umgekehrter Reihenfolge. Alle Steuer- ungsparameter werden mit derselben Startsequenz erzeugt. Der vom Sender zum Empfänger transferierte Chiffretext wird eingelesen:

Chiffretext an den Empfänger: Nachricht.txt.ctx

ঙf ɲ ず넊 ঘY ចЩU ´ ศঔ৩৩ せ ŋ¸ ঙr ঘTФ 任 লক্ষn 냎 ল P□ づ仕 I 仜な ঔ 넊s ศศЖКЛ r ยέÏ 깞
r ลI ঌ 깗냎Ï ち I ず മЩU ɵ □ ঞ্ɥ ষছ なてず Ш 仡μ □ ঔЦ だ Щ ា ঽRή ঙ ঽঔR だ 巭□ ঢ 仞
কRŸЦH 付 ఠঀঌ১ক¸ ఠঽ□ঀঔ 任 Щ μ D ঔGRচ৩η ঽঘ´ ঢ´ ঌចη p ´ ঞЙ

Die Bitkonversion wird durchgeführt und das Ergebnis mit dem Runden-Schlüssel XOR-verknüpft, so dass auch hier ein **one-time-pad** entsteht. Am Ende ergibt sich der ursprüngliche Klartex.

Schlussfolgerung

Das Zusammenspiel von „**Codepages**" mit der „**one-time-chain**" und der „**Bit-Konversion**" ergeben für das Verfahren eine absolute Sicherheit. Nach derzeitigem Stand der Technik erscheint der erzeugte Chiffretext:

nicht brechbar.

Weitere Informationen zu Python-Modulen können unter www.telecypher.net/ abgerufen werden. Für Fragen und Erläuterungen steht der Autor unter

eschnoor@multi-matrix.de

jederzeit zur Verfügung.

München, im August 2020

[#1] Algorithmus-CypherMatrix,telecypher.net/Algorithmus-CypherMatrix.pdf
[#2] de.wikipedia.org/wiki/Zeichensatztabelle
[#3] de.wikipedia.org/wiki/Unicode
[#4] wiki.selfhtml.org/wiki/ Zeichencodierung, wikipedia.org/wiki/Codepage
[#5] Matrix-Generator, http://www.telecypher.net/Matrix-Generator.pdf
[#6] de.wikipedia.org/wiki/Exklusiv-Oder-Gatter
[#7] de.wikipedia.org/wiki/One-Time-Pad
[#8] de.wikipedia.org/Base64