
-[PGP]-

ein Erklaerungsversuch
[exklusiv fuer WOOOS]
[by Kaneda ~ v1.1]

PGP - stehend fuer 'Pretty Good Privacy' - ist das weitverbreiteste Hybrid-Codierungssystem (fuer kleinere Datenmengen) auf der Welt.
Es benutzt unter anderem die Verschluesselungs-Algorithmen IDEA und RSA.

Theoretisch funktioniert PGP wie folgt: Mal angenommen, AlSimsa und Nory stehen in Kontakt zueinander und tauschen Dokumente ueber ein unsicheres Netz aus. Nory moechte von AlSimsa einen Brief bekommen und generiert dafuer ein Schluesselpaar.
Als erstes seinen sog. PublicKey Pn(), den er frei im Netz verfuegbar macht, wo AlSimsa aber auch der spionierende Raven sich den abholen koennen. Der zweite ist der PrivateKey (sagen wir mal S fuer Secret) Sn(), den er nicht weitergibt. Die message M soll verschluesselt und als Code C an Nory gesandt werden. Dazu verwendet AlSimsa Pn(). Die Sache saehe - auch fuer Nichtmathematiker - in etwa so aus: C=Pn(M)
Der Code (oder String) - ein Haufen ASCII-Geduenst - wird also von AlSimsa zu Nory geschickt und unterwegs auch von Raven kopiert, was aber nix macht, denn dieser ist nicht im Besitz von Sn - und somit unfaehig, C zu entschluesseln. Nory dagegen kennt seinen Sn und kann C entschluesseln: M=Sn(Pn(C)).

Aber dass der Inhalt von C wirklich von AlSimsa kam, ist nicht voellig sicher. (Raven koennte sie ausgetauscht haben - denn schliesslich ist Pn frei zugaenglich, so kann auch er nur von Nory zu oeffnende Daten schicken)
Wir brauchen also eine Garantie, dass C auch wirklich von AlSimsa kommt. Also generiert AlSimsa sich ebenfalls ein Schluesselpaar - Pa() und Sa() - und verschluesselt M mit seinem SecretKey Sa(). Nory kann dann das Dokument mit AlSimsa seinem PublicKey decodieren. Das macht in sofern wenig Sinn, denn schliesslich ist es Raven ebenfalls moeglich, das Dokument zu decodieren. Also verschluesselt AlSimsa dies Dokument nochmals wie oben beschrieben: C= Pn(Sa(M)). Dieses Paket wird an Nory geschickt, der es mit seinem SecretKey oeffnet; zum Vorschein kommt Sa(M). Wenn nun Nory versucht, die Message mit AlSimsa seinem PublicKey zu entschluesseln, wird die Authenzitaet der Message garantiert, denn das Dokument wird nur Sinn ergeben, wenn sie von AlSimsa verschluesselt worden ist. Schliesslich ist nur AlSimsa der SecretKey bekannt. So kann AlSimsa quasi eine digitale Unterschrift dem gesendeten Dokument beigeben.

Der echte Algorithmus ist selbstredent etwas komplizierter, aber obiges Schema soll lediglich die Idee von PGP veranschaulichen.

Nun, der Versuch, die Funktionsweise von PGP (auf dem Codierungsverfahren 'RSA', 1977 von Rivest, Shamir and Adleman erfunden, beruhend) zu erklaeren: Als erstes werden zwei grosse Primzahlen ausgesucht, p und q, dann daraus eine Nummer n errechnet: n=pq. Nach einigen anderen Berechnungen sind zwei Schluessel dabei rausgekommen:

PublicKey (n,e)
SecretKey (d)
(wobei e und d Exponenten darstellen, urspruengliche p und q gehoeren zerstoert)

Diese zwei Schluessel werden zur Ver- und Entschluesselung von Daten gebraucht. [Aus einem Dokument uebernommene, geltende Zahlen koennten sein:]

Publickey (2537, 47)
Secretkey (311)

Buchstaben werden bei dem Input des Passwortes in Ziffern
umgewandelt (A=01, B=02, usw), welche dann als Integralwert in Bloecke
zersplittet werden. Jeh groesser die zuvor ausgewaehlte Primzahl ist,
desto groesser sind diese Bloecke. Wenn nun das Passwort 'ME' benutzt
wuerde, erhielten wir den Integralwert 1305. Eine dann vollzogene Rechnung
ist dann 1305^47.
Wir reden hier aber nicht von so popeligen, sondern _sehr_ grossen
Primzahlen, deren Exponent um ein vielfaches groesser als die eigentliche
Zahl ist. (2^2048 !). Wie dem auch sei. Laut Formel - wir wissen jetzt,
was die Variabeln zu sagen haben - wird wie folgt verschluesselt:
C=M^e mod n
Das ist, wenn das Passwort etwas intelligenter als im Beispiel und lang genug
gewaehlt ist, nahezu unknackbar. Nahezu desshalb, weil es nicht bewiesen ist,
dass es _keine_ Formel fuer das Faktorisieren von grossen Zahlen gibt.
Die eigentlichen Gruende, warum RSA mehr und mehr unsicher wird, liegt in dem
fortschreiten der Faktorisierungs-techniken, der ansteigenden Rechenleistung
von Computern (mehr Versuche in weniger Zeit) und der Fakt, dass Computer einfach
billiger werden.
Es gibt nur mehr oder weniger zwei Methoden, die dem Neugierigen bleiben:
Zum einen 'Bruteforce'-maessig zu raten, was in etwa 'bis zum Ende des
Universums und noch ein bisschen mehr' dauern durfte. Und zum anderen,
und so wird es praktiziert - ist das eben angesprochene Faktorisieren.
Es werden grosse Zufallszahlen generiert, danach darauf ueberprueft, ob
es Primzahlen sind. Sind sie es, wird ausprobiert, ob zwei von ihnen
n ergeben. Dann muss noch der String, sprich das Passwort errechnet und
ueberprueft werden, ob dieser passt. Dieses Faktorisieren dauert 'extremly long'
und die restlichen Prozeduren mindestens genauso lang.
Das alles bleibt demjenigen erspart, der d kennt:
M=C^d mod n

Zum Abschluss noch eine kleine - stumpf uebernommene Tabelle, welche aus
dem Jahre 2000 stammt:

KeySize	MIPS-years required to factor
512	30,000
768	200,000,000
1024	300,000,000,000
2048	300,000,000,000,000,000,000

bleibt nur zu hoffen, dass die NSA bislang
keinen 'polynomial-time-factoring-algorithm'
gefunden und an ECHOLON weitergegeben hat ;-)

_____
credits:
zwanderer [aus einem Dokument vom Jan/2002]
W. Unruh [aus einem Artikel im axion.physics anno 2000]
Phil R Zimmermann [Erfinder von PGP]

Kommentare, Kritik, Verbesserungen, Ergaenzungen, Belehrungen und/oder aber
eine kleine Hilfestellung bezueglich dem fehlenden IDEA-Teil per mail an:
kaneda@wooos.de
v1.1 - 01.2002
