# goldbug-manual-de

Manual / Benutzerhandbuch des GoldBug Crypto Messengers (Deutsch / German)

View on GitHub

## GoldBug-manual

Manual of the GoldBug Crypto Messenger (Deutsch / German)




https://compendio.github.io/goldbug-manual/

& https://compendio.github.io/goldbug-manual-de/

Edited by Scott Edwards (2018, Review at Github).

Inhaltsverzeichnis


1 Was ist GoldBug?
- 1.1 Warum ist es wichtig, dass Internet-Nutzer Kommunikation verschlüsseln
- 1.2 Woher kommt der Name "GoldBug"?

2 Verschlüsselung: GoldBug hat Alternativen zu RSA
- Der erste NTRU & McEliece Messenger
- 2.1 Asymmetrische Verschlüsselung mit PKI: RSA, Elgamal und insbesondere NTRU und McEliece im Vergleich
- 2.2 Anwendung der Block Cipher Modi & Encrypt-then-MAC
- 2.3 Hybrides Verschlüsselungs-System
- 2.4 Symmetrische Verschlüsselung mit AES

3 Was ist das Echo-Protokoll?
- 3.1 Volles Echo
- 3.2 Halbes Echo
- 3.3 Echo Accounts
- 3.4 Das Echo-Grid
  - 3.4.1 Beispiele des Schlüssel-Austausches von Alice, Bob, Ed & Maria
- 3.5 Adaptives Echo (AE) und seine AE-Tokens
  - 3.5.1 Hänsel und Gretel – Ein Beispiel für den Adaptiven Echo Modus
- 3.6 Einige Beispiele, wie das Echo-Protokoll funktioniert

4 Cryptographisches Discovery

5 Einen ersten Setup einrichten – z.B. mit dem Wizard
5.1 Zwei Login-Methoden
5.2 Generierung von 10 Schlüsseln für die Verschlüsselung
5.3 Aktivierung des Kernels
5.4 Einen Nachbarn mit der IP-Adresse verbinden

6 Die Chat-Funktion
6.1 Freund hinzufügen durch Tausch und Einfügen der Schlüssel
6.1.1 Besonderheit: Repleo
6.2 Einen ersten sicheren Chat beginnen
6.3 Erg. Sicherheitsmerkmal: MELODICA: Calling mit einem Gemini
6.3.1 Asymmetrisches Calling
6.3.2 Instant Perfect Forward Secrecy (IPFS)
6.3.3 Symmetrisches Calling
6.3.4 2-Wege-Calling
6.4 Erg. Sicherheitsmerkmal: Socialist Millionaire Protocol
6.4.1 SMP-Calling
6.5 Erg. Sicherheitsmerkmal: Forward Secrecy (asymmetrisch)
6.5.1 Forward Secrecy Calling
6.6 Übersicht der verschiedenen Calling-Arten
6.7 Emotikons
6.8 Datei-Transfer im Chat-pop-up-Fenster

7 Gruppen-Chat im IRC-Stil

8 Smoke Mobiler Chat Client
8.1. Smoke Android Client
8.2. Fire to Buzz Chat

9 Die E-Mail-Funktion
9.1 POP3
9.2 IMAP
9.3 P2P E-Mail: ohne Vorratsdatenspeicherung
9.4 C/O und E-Mail Institutionen einrichten
9.4.1 Care-Of-Methode (c/o)
9.4.2 Virtuelle E-Mail Institution ("VEMI") - Methode
9.5 Zusätzliche Verschlüsselung: Ein „GoldBug“ auf ein E-Mail setzen
9.6 Forward Secrecy
9.7 Secret Streams

10 POPTASTIC - Verschlüsselter Chat und E-Mail über POP3 & IMAP
10.1 Chat über POPTASTIC
10.2 E-Mail über POPTASTIC
10.3 Einrichtung von POPTASTIC
10.4 Weiterentwicklung von POPTASTIC

11 FileSharing: mit StarBeam (SB)
11.1 StarBeam-Magneten erstellen mit Verschlüsselungswerten
11.1.1 Option „Nova“: Datei vor Versand verschlüsseln?
11.1.2 Nutzung eines One-Time-Magneten
11.1.3 Magnet-URI-Standards für kryptographische Werte

11.1.4 Rewind Funktion
11.1.5 Vergleich mit Turtle-Hopping
11.2 StarBeam-Upload: Eine Datei übertragen
11.3 StarBeam-Downloads
11.3.1 Werkzeug: StarBeam-Analyzer
11.3.2 Ausblick auf Crypto-Torrents

12 Web-Suchmaschine mit URL-Datenbank
12.1 Datenbank Setup
12.1.1 SQLite
12.1.2 PostgreSQL
12.2 URL-Filter
12.3 URL-Community
12.4 Pandamonium Webcrawler
12.5 RSS-Reader und URL-Import

13 Chat/E-Mail-Server einrichten
13.1 Chat-/E-Mail-Server über einen Listener einrichten
13.1.1. Server Broadcast
13.1.2. Sicherheitsoptionen
13.1.3. Proxy- & Firewall-Anmerkungen: z.B. Betrieb über Tor
13.1.4. GoldBug als Lan Messenger
13.2 Server/Listener-Erstellung Zuhause hinter einem Router/Nat
13.3 Nutzung von GoldBug im TOR-Netzwerk
13.4 Spot-On Server
13.5 Spot-On Lite Server als Deamon
13.6 SmokeStack Server unter Android
13.7 Bluetooth Server
13.8 UDP Server
13.9.SCTP Server
13.10 NCat Verbindung

14 Werkzeuge
14.1 Werkzeug: Verschlüsselung von Dateien (FileEncryptor)
14.2 Werkzeug: Das Rosetta CryptoPad
14.3 Werkzeug: Echo Public Key Share (EPKS)
14.4 Pass-Through-Funktionalität
14.5 Anzeige Analysen und Statistiken

15 BIG SEVEN STUDY: Crypto-Messenger-Audit

16 Die digitale Verschlüsselung der privaten Kommunikation im Kontext von ...
16.1 Principles of the protection of private speech, communication and li
Universal Declaration of Human Rights, 1948 (Art. 12)
16.2 International Covenant on Civil and Political Rights, 1966 (Art. 17)
16.3 European Convention on Human Rights, 1950 (Art. 8)
16.4 Charter of Fundamental Rights of the European Union, 2000 (Art. 7, 8
16.5 Basic Law e.g. Federal Republic of Germany, 1949 (Art.2 Abs.1 & Art.
16.6 Art. 10 - Privacy of correspondence, posts and telecommunications
16.7 § 206 - Verletzung des Post- oder Fernmeldegeheimnisses
16.8 U.S. Constitution: Search and Seizure (Expectation of Privacy, US Su

17 Historie der Programm-Veröffentlichungen

18 Webseite

19 Offener Quellcode
19.1 Informationen zur Kompilierung

20 Schriftenverzeichnis

Manual-Dateien: https://github.com/compendio/goldbug-manual-de

Manual Deutsch/German: https://github.com/compendio/goldbug-manual-de

PDF: https://sf.net/projects/GoldBug/files/goldbug-manual-de.pdf

Website: http://goldbug.sourceforge.net/

Download: https://sourceforge.net/projects/goldbug/files/

Source: https://github.com/textbrowser/spot-on

# 1 Was ist GoldBug?

GoldBug ist ein sicherer Instant Chat Messenger und verschlüsselnder E-Mail-Client, der darüber hinaus auch noch weitere Funktionen beinhaltet wie Gruppenchat, Dateitransfer sowie auch eine URL-Suche auf Basis einer implementierten URL-Datenbank.

Damit sind die von einem regulären Internet-Nutzer häufig genutzten drei Grundfunktionen im Internet - Kommunikation (Chat/E-Mail), Web-Suche und Datei-Transfer - in einer verschlüsselnden Umgebung sicher und umfänglich abgebildet.

Darüber hinaus sind in GoldBug auch eine Reihe an nützlichen Werkzeugen implementiert, z.B. die Server-Funktionalität für verschlüsselten Chat, proxyfähige Durchleitungen, Pads zur Wandlung von Text bzw. Dateien in Ciphertext und umgekehrt, ein Feedreader und ein WebCrawler, oder auch Übersichten für Freunde von Statistiken und Analysen, und einiges mehr.

Mit der Nutzung von GoldBug - kurz GB - kann der Nutzer daher aufgrund der Verschlüsselung relativ sicher sein, dass kein unerwünschter Dritter die Gespräche belauschen bzw. E-Mails oder Dateiübertragungen öffnen kann. Auch die URL-Suche geschieht auf der lokalen Maschine, so dass Suchanfragen geschützt und sicher bleiben.

Die Nutzer-zu-Nutzer Kommunikation soll mit dieser Anwendung auch über das Internet im privaten, geschützten Raum verbleiben.

Dafür nutzt GoldBug starke Vielfach-Verschlüsselung, auch hybride Verschlüsselung genannt, mit verschiedenen Ebenen von moderner Verschlüsselungs-Technologie basierend auf etablierten Verschlüsselungs-Bibliotheken - wie libgcrypt (bekannt von OpenPGP bzw. GnuPG und OpenSSL).

Zum Beispiel werden damit für jede Funktion eigene und voneinander verschiedene öffentlich/private Schlüssel zur Verschlüsselung und für die Signaturen erstellt - basierend auf den Verschlüsselungsalgorithmen RSA, oder wahlweise auch Elgamal und NTRU. Neben NTRU ist auch der Verschlüsselungsalgorithmus McEliece quelloffen implementiert.

Diese beiden letztgenannten Algorithmen gelten als besonders sicher gegenüber Angriffen, die aus dem Quantum Computing bekannt sind und zukünftig aufgrund von schnellen Quantum-Computern zunehmend relevanter werden. GoldBug ist damit weltweit (einer) der erste(n) Messenger, der diese beiden Algorithmen implementierte und damit die Abkehr bzw. Alternativen vom dem seit 2016 offiziell als gebrochen geltenden RSA-Algorithmus einleitete (siehe NIST zit. n. Adams/Maier 2016).

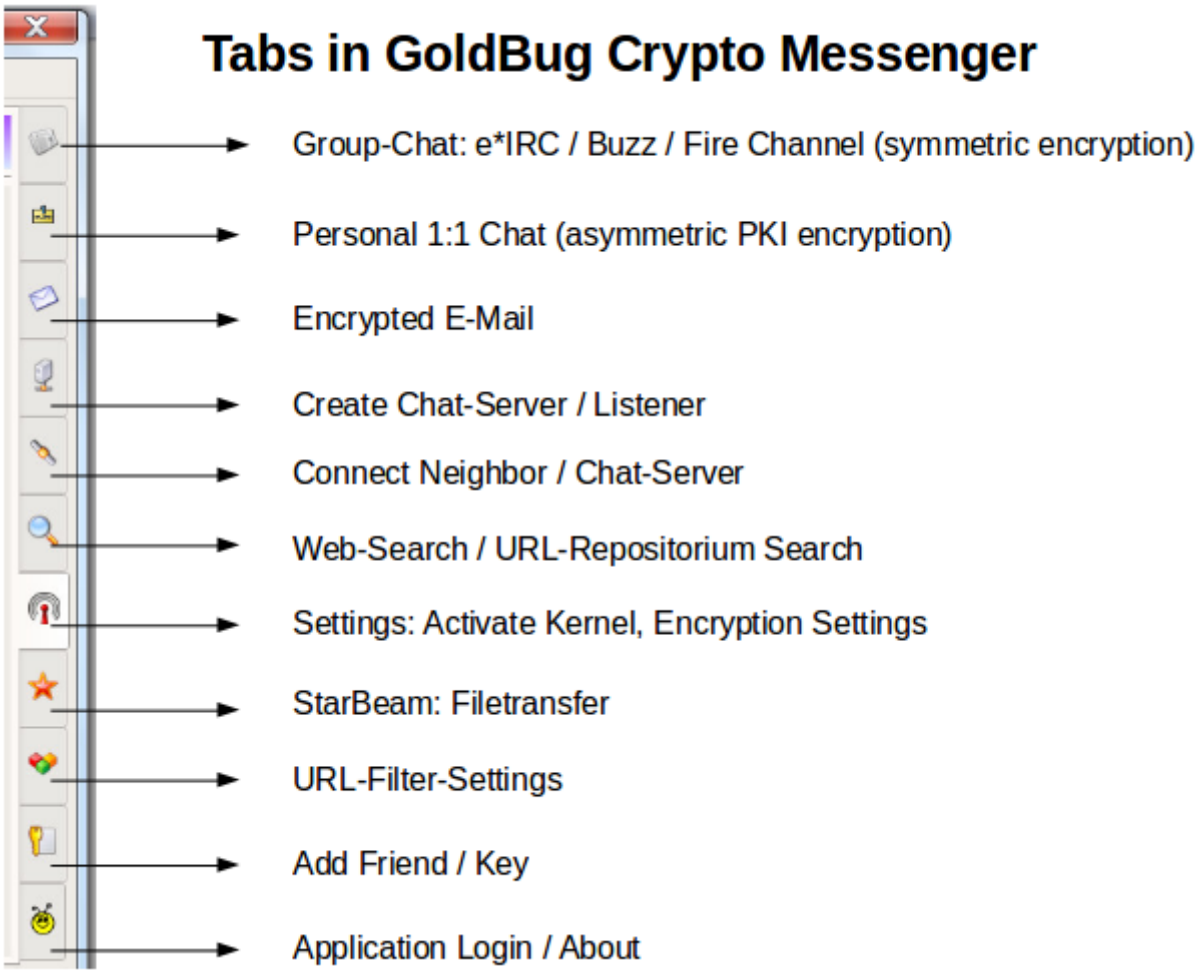
Weiterhin bietet die Applikation auch dezentrales und verschlüsseltes E-Mail und auch dezentralen öffentlichen Gruppen-Chat im IRC-Stil an. Schließlich besteht auch die Funktion, eine URL-Websuche in einem dezentralen Datenbank-Repositorium umzusetzen: Nutzer können zu ihren thematischen RSS-Feeds somit URLS und Inhalte der Webseite - wie bislang Booksmarks im Browser und dessen Cache - in einer komfortabel durchsuchbaren Datenbank abspeichern bzw. dort importieren, die entweder auf SQL oder PostGres basiert und p2p netzwerkfähig ist.

Beim E-Mail kann IMAP, POP3 und drittens, P2P E-Mail genutzt werden - GoldBug ist somit auch ein voll funktionsfähiger E-Mail-Klient. Alsbald verschlüsselte E-Mails gesandt werden, ist es erforderlich, dass auch der Freund diesen Klienten nutzt. Das hat den Vorteil, dass der Verschlüsselungs-Key nur einmalig zu tauschen ist, dann aber nicht mehr bei jedem einzelnen E-Mail angewandt werden muss. Diese Funktion, den Schlüssel auf direktem Weg verschlüsselt zurück zu übertragen, wird in GoldBug “Repleo” genannt und ist später dann auch in eingigen anderen Projekten unter der Bezeichnung Autocrypt bzw. KeySync automatisiert übernommen worden.

Wie in jedem Nachrichtenprogramm können auch Dateien geteilt und versandt werden. Der Versand ist per sé immer verschlüsselt. Mit den Werkzeugen “Rosetta CryptoPad” und dem “File-Encryptor” kann der Nutzer auch Text und/oder Dateien nochmals zusätzlich sicher verschlüsseln oder auch zurück konvertieren. Diese Werkzeuge zur Verschlüsselung sind somit auch für andere Übertragungswege (wie unverschlüsselge Wege außerhalb von GoldBug) nutzbar.

Mit all seinen Ausstattungen ist GoldBug daher eine sogenannte “Communication Suite” - ein Programm mit zahlreichen Funktionen für die sichere Kommunikation, das die Übertragung der verschlüsselten Pakete mit dem sogenannten Echo-Protokoll realisiert und besonders absichert, wie es weiter unten noch erläutert wird.

Abbildung: Erklärung der Tabs in GoldBug Crypto Messenger




## 1.1 Warum ist es wichtig, dass Internet-Nutzer Kommunikation verschlüsseln

Heutzutage sind fast alle kabellosen Wifi-Internetzugänge mit einem Passwort geschützt (Freifunk-Aktivitäten versuchen gerade, diese Überregulierung wieder zurückzunehmen durch Passwort- und Account-freie Zugänge zum Internet). In wenigen Jahren werden Klartext-Nachrichten oder E-Mails an Freunde (im Folgenden gelten alle Begriffe immer für alle Geschlechter) über das Internet ebenso verschlüsselt sein - sein müssen. Um diesen Wandel zu festigen, wird manchmal auch von C-Mail (für Crypto-Mail) statt E-Mail als neuem Begriff gesprochen.

Verschlüsselung ist keine Frage, ob man etwas zu verbergen hat oder nicht, es ist die Frage, ob wir selbst unsere Kommunikation kontrollieren - oder sie durch Andere, Dritte kontrolliert wird.

Es ist letztlich auch eine Frage des Angriffs auf das freien Denken und eine Frage der Streichung der Annahme einer Unschuldsvermutung (“Im Zweifel für den Angeklagten” - wenn jeder Bürger überhaupt auf eine Anklagebank gehört!).

Demokratie erfordert das Denken und die Diskussion von Alternativen im Privaten wie auch in der Öffentlichkeit.

Die Kommunikation und Datenübertragungen über das Internet soll so geschüzt werden, wie Eltern auch ihre Lieben oder eine Vogel-Mutter ihre Jungen gegenüber Unbekannten schützen würde: Jeder soll seine Datenschutz- und Menschenrechte mit modernen Verschlüsselungsfunktionen selbstverteidigen.

Starke Multi-Verschlüsselung (sogenannte “hybride Verschlüsselung”) sichert also letztlich auch die Erklärungen der Menschenrechte in ihrem breit konstituiertem Konsus und ist eine digitale Selbstverteidigung, die jeder erlernen und nutzen sollte - um letztlich auch zur Demokratie beitragen und diese stützen zu können.

Der GoldBug Messenger versucht, ein einfach zu nutzendes Werkzeug für diesen Anspruch zu sein. Ähnlich der Sicherheitsentwicklung beim Automobil wird sich auch die E-Mail- & Chat-Verschlüsselung entwickeln: ist man beim Automobil zunächst ohne Sicherheitsgurt gefahren, fahren wir heute hingegen mit verpflichtenden Sicherheitsgurten (z.B. seit 1975 in Deutschland) und zusätzlichen Airbags oder drittens noch ergänzenden elektronischen Sicherheits-Informationssystemen.

GoldBug ist dabei ein einfach zu bedienendes, jedoch in gewissem Umfang auch zu erlernendes Programm; es erfordert - wie beim Auto-Führerschein - die Kenntnis der verschiedenen Bedienelemente und Optionen. GoldBug ist somit eine bereits vereinfachte Benutzeroberfläche gegenüber der originalen Benutzeroberfläche, die aus dem “Spot-on”-Projekt kommend auch Spot-on genannt wird. Ähnlich einem Cockpit eines Flugzeuges sind in dieser originalen Benutzeroberfläche noch wesentlich mehr Bedienelemente vorhanden. In GoldBug sind diese bereits reduziert und auch eine weitere Minimal-Ansicht wird für Einsteiger in Software für kryptographische Prozesse angeboten. Insofern gilt: Erlernen, was noch unbekannt ist und beachten, dass es bereits ein reduzierter Umfang ist. Dieses vorliegende Manual kann dabei helfen, sich die einzelnen Funktionen zu erschließen. Und Nutzer, die erst lesen und dann ausprobieren, sind wie immer klar im Vorteil. :-)

Die unverschlüsselte Plain-Text-E-Mail oder Chat-Nachricht sollte daher eigentlich ausgedient haben, nachdem man im Jahr 2013 mit den Snowden-Papieren feststellte, dass private E-Mails in großem Umfang von vielen Interessierten global abgehört und dabei systematisch erfasst, gespeichert und ausgewertet werden.

Nebenbei bemerkt: Der Schriftzug des GoldBug-Logos ist in der Schriftart “Neuland” geschrieben - ein Font, der im Jahre 1923 vom Schriftkünstler Rudolf Koch entwickelt wurde. Der Logo-Schriftzug ist seit dem ersten, zeitgleichen Release von GoldBug im Jahre 2013 interessanter Weise eine Allusion an den “Satz des Jahres” 2013, in dem die deutsche Bundeskanzlerin Angela Merkel - im Zusammenhang mit der Überwachungs- und Spionageaffäre 2013 und dem Abhören ihres Telefons - in einem Gespräch mit dem amerikanischen Präsidenten Barack Obama den Satz prägte: „Das Internet ist für uns alle Neuland.” ..

.. - Wie lange Verschlüsselung für die nachfolgenden Schülergenerationen “Neuland” bzw. eine im wahrsten Sinne des Wortes “Geheimwissenschaft” bleibt – oder eine Art „Gurtpflicht“ auch E-

Mails zu C-Mails wandeln wird, entscheiden die Lernenden, Lehrenden sowie die Medien und Techniker - in jedem Falle aber jeder selbst (z.B. als Leser dieses Manuals) mit den Freunden, mit denen die Verschlüsselungssoftware angewandt wird.

Abbildung: GoldBug Logo (Smiley)



## 1.2 Woher kommt der Name “GoldBug”?

Der Gold-Käfer (“The Gold-Bug”) ist eine Kurzgeschichte von Edgar Allan Poe: In der Handlung geht es um William LeGrand, der kürzlich einen gold-farbenen Marienkäfer entdeckte.

Sein Kumpel, Jupiter, erwartete nun, dass LeGrand sich weiterentwickeln wird in seiner Suche nach Erkenntnis, Reichtum und Weisheit, nachdem er mit dem Goldkäfer in Kontakt gewesen ist

- und geht daher zu einem weiteren Freund von LeGrand, ein mit Namen nicht weiter benannter Erzähler, der es für eine gute Idee hält, seinen alten Freund mal wieder zu besuchen. Nachdem LeGrand sodann auf eine geheime Botschaft gestoßen ist und diese erfolgreich entschlüsseln konnte, starten die drei ein Abenteuer als Team.

Der Gold-Käfer - als eine der wenigen Stücke in der Literatur - integriert Verschlüsselungstexte als Element der Geschichte. Poe war damit der Popularität von Verschlüsselungstexten seiner Zeit weit voraus als er im Jahre 1843 “The Gold-Bug” schrieb, in dem der Erfolg der Geschichte sich z.B. um ein solches Kryptogramm und metaphorisch um die Suche nach der Erkenntnis aus dem Stein der Weisen drehte.

Der Gold-Käfer war eine viel gelesene Geschichte, äußerst populär und von den Literaten das meist untersuchte Werk von Poe während seiner Lebenszeit. Seine Ideen halfen ebenso das Schreiben verschlüsselter Texte und so genannter Kryptogramme weiter bekannt zu machen (vgl. auch Wikipedia).

Über 170 Jahre später hat Verschlüsselung im Internet-Zeitalter mehr Gewicht denn je. Verschlüsselung sollte ein Standard sein, wenn wir Kommunikation über das unsichere Internet senden - Grund genug daher, mit dem Namen der Applikation an die Ursprünge des verschlüsselten Schreibens zu erinnern.

GoldBug ist somit eine historische Hommage. Die ggf. eine Gewöhnung an den Begriff erfordert, denn unter „Bug“ wird in der IT-Sprache oftmals eine Fehlerkorrektur verstanden. Die Auffassung, einen Goldkäfer genauso Wertzuschätzen wie ein anderes Kuscheltier, erfordert daher – je nach Mensch – ggf. eine starke kognitive Reorganisation eines bislang geprägten Weltbildes oder die routinierte Ausweitung der Wertschätzung von Bug-Funden als interessante Forschungsfunde.

Wer gern Neues erkundet, offen an das, was vorgefunden wird, herangeht, wird auch bei kryptographischen Prozessen mit der Applikation GoldBug viele Dinge lernen und vertiefen können, wenn bislang noch kein Zugang zu diesem “Neuland” ermöglicht wurde. Für Lehrpersonen ist die Software daher ein interessantes Lehrinstrument, das Kryptographie in praktischer Implementierung und in Übungen mit spielerischem Austesten vorstellen und erproben kann und an die Anfänge der populären Kryptographie seinerzeit errinnert.

## 2 Verschlüsselung: GoldBug hat Alternativen zu RSA - Der erste NTRU & McEliece Messenger

Verschlüsselung ist immer nur so gut, wie die mathematischen Berechnungen nicht durch die Automation von Computern in Windeseile berechnet werden können. Daher wird mathematisch gesehen die Primfaktorzerlegung genutzt, da sie zum Teil jahrelangen Rechenaufwand benötigt.

Es werden grundsätzlich zwei Methoden zur Verschlüsselung unterschieden. Zum einen die symmetrische Verschlüsselung: Beide Nutzer wenden das gleiche Passwort an, z.B. ein

sogenanntes AES mit 32 Zeichen, was noch weiter unten ausführlicher erläutert wird, oder zum anderen die a-symmetrische Verschlüsselung. Bei der a-symmetrischen Verschlüsselung hat jeder Nutzer zwei Schlüssel, einen privaten und einen öffentlichen Schlüssel. Die Nutzer tauschen jeweils den öffentlichen Schlüssel und können dann Daten mit Anwendung des privaten Schlüssels in Kombination mit dem öffentlichen Schlüssel verschlüsseln. Nach Übertragung ist auch die Gegenseite in der Lage, mit dem eigenen privaten Schlüssel die Nachricht zu entziffern. Das asymmetrische Verfahren wird auch PKI genannt: Pubic Key Infrastructure genannt, die auf verschiedenen Algorithmen für die Schlüsselerzeugung aufbauen kann.

Dennoch ist Verschlüsselung - sei es über AES oder PKI - nicht unknackbar und die Prozeduren und Bibliotheken müssen auch gut angewandt werden, damit sie sicher sind. RSA gilt dabei „heute als ein wesentlicher, vielfach untersuchter und noch nicht knackbarer Verschlüsselungsstandard - wenn auch die Weiterentwicklung der schnellen Computer eine andere Zukunft bringen mag“, – so wurde es noch 2014 in diesem Manual vermerkt. Im Jahr 2016 gab dann das offizielle NIST Institut bekannt, dass RSA als gebrochen gilt im Zeitalter des Quantum Computings (siehe NIST zit. n. Adams/Maier 2016).

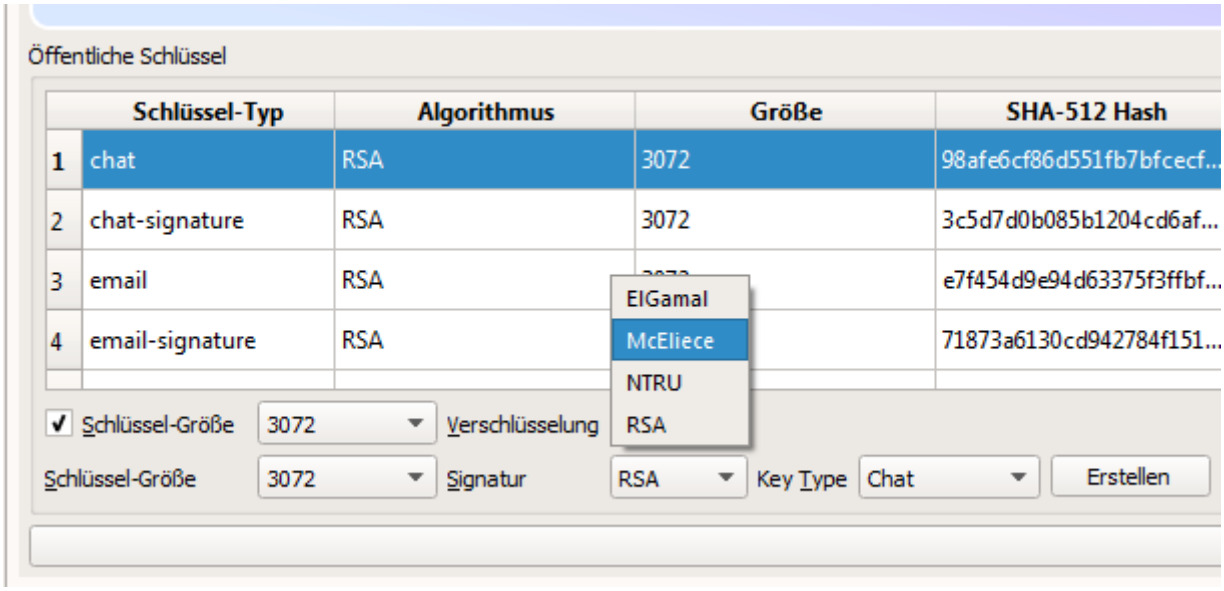
Die Medien haben es kaum aufgegriffen, da wahrscheinlich jeder einstimmen wird, dass man einen Quanten-Computer nicht im nächsten Supermarkt kaufen könne, und das Problem daher nicht relevant sei. Es hat den Charme von Kindern, die sich die Hand vor Augen halten und das Problem oder eine Gefährdung somit nicht an ihre Realitätswahrnehmung heranlassen. Dennoch gilt offiziell als bestätigt: RSA kann – zwar mit speziellen Mitteln – gebrochen werden. Die Sicherheit ist dahin. Das bedeutet auch Auswirkungen auf unsere Internet-Ökonomie und das Online-Banking, denn bislang sind auch die sicheren Verbindungen auf RSA aufbauend und ein TLS zur Absicherung der Verbindung bei Online-Banking oder -Shopping basierend auf McEliece oder NTRU gibt es bislang noch nicht.

## 2.1 Asymmetrische Verschlüsselung mit PKI: RSA, Elgamal und insbesondere NTRU und McEliece im Vergleich

GoldBug Messenger hat daher schon frühzeitig ergänzend verschiedene Alternativen zu RSA eingebaut - falls dieser Verschlüsselungs-Algorithmus-Standard einmal unsicher würde: Ggf. kann RSA mit entsprechend großer Größe des Schlüssels (mind. 3072 Bytes) von nicht spezialisierten technischen Administrationskräften noch als zeitliche Hürde betrachtet werden, zumal GoldBug auch mit der Mehrfach-Verschlüsselung weitergehende Absicherungen vorhält.

Neben RSA hat GoldBug dennoch die Verschlüsselungsalgorithmen Elgamal und auch NTRU und McEliece implementiert. Die beiden letzt-genannten gelten auch als besonders resistent gegen die Angriffe, die aus dem Quantum Computing bekannt sind.

Abbildung: McEliece Algorithmus zukunftsweisender Schutz gegen Angriffe aus dem Quantum Computing




GoldBug nutzt die libgcrypt-, libntru- und McEliece-Bibliotheken für die Erzeugung der dauerhaften privaten und öffentlichen Schlüsselpaare. Derzeit generiert die Anwendung Schlüsselpaare für jede der einzelnen zehn Funktionen während der Initialisierung. Die Schlüsselerzeugung ist optional. Folglich erfordert GoldBug zwingend keine Public Key-Infrastruktur. Auch nachgelagert können die gewünschten Algorithmen ausgewählt werden und Schlüssel generiert werden.

Bei den bei Verschlüsselung optional verfügbaren Signaturverfahren besteht ebenso eine umfassende Auswahl: DSA, ECDSA, EdDSA, Elgamal und RSA. Signatur bedeutet, dass der erstellte Schlüssel für die Verschlüsselung nochmals mit einem Schlüssel signiert wird, um nachweisen zu können, dass eine Nachricht auch von einem bestimmten Teilnehmer und niemand anderem kommt. Die OAEP und PSS Schemata sind mit der RSA-Verschlüsselung und RSA-Signatur entsprechend verwendet.

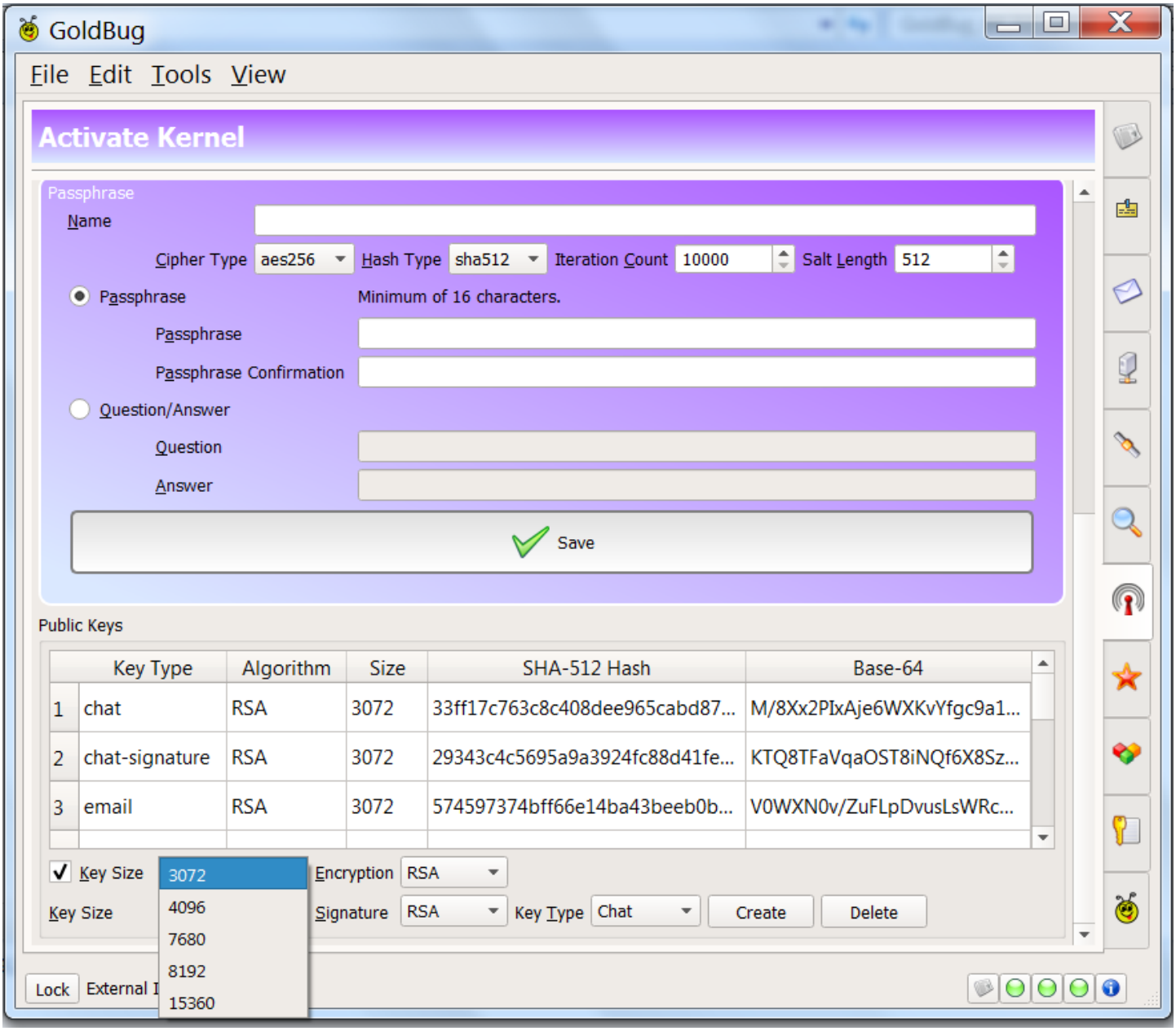
Abbildung: RSA und seine Alterntiven in GoldBug

| **McEliece-Kryptosystem** |
| --- |
| Das McEliece-Kryptosystem ist ein asymmetrischer Verschlüsselungsalgorithmus. Es wurde 1978 vom Kryptographen Robert J. McEliece vorgestellt. Es ist selbst unter Verwendung von Quantencomputern keine effiziente Methode bekannt, mit der das McEliece-Kryptosystem gebrochen werden kann, was es zu einem vielversprechenden Algorithmus für Post-Quanten-Kryptographie macht. |
| **NTRU** |
| NTRU ist ein asymmetrisches Verschlüsselungsverfahren, das 1996 von den Mathematikern Jeffrey Hoffstein, Jill Pipher und Joseph Silverman entwickelt wurde. Es basiert lose auf Gitterproblemen, die selbst mit Quantenrechnern als nicht knackbar gelten. Allerdings ist NTRUEncrypt bisher nicht so umfänglich untersucht wie gebräuchlichere Verfahren (z.B. RSA). NTRUEncrypt ist durch IEEE P1363.1 standardisiert (vgl. NTRUEncrypt). |

| Elgamal |
| --- |
| Das Elgamal-Verschlüsselungs-Verfahren oder Elgamal-Kryptosystem ist ein im Jahr 1985 vom Kryptologen Taher Elgamal entwickeltes Public-Key-Verschlüsselungsverfahren, das auf der Idee des Diffie-Hellman-Schlüsselaustauschs aufbaut. Das Elgamal-Verschlüsselungsverfahren beruht, wie auch das Diffie-Hellman-Protokoll, auf Operationen in einer zyklischen Gruppe endlicher Ordnung. Das Elgamal-Verschlüsselungs-Verfahren ist beweisbar IND-CPA-sicher unter der Annahme, dass das Decisional-Diffie-Hellman-Problem in der zugrundeliegenden Gruppe nicht trivial ist. Verwandt mit dem hier beschriebenen Verschlüsselungsverfahren (aber nicht mit diesem identisch) ist das Elgamal-Signaturverfahren (Das Elgamal-Signatur-Verfahren ist in GoldBug bislang nicht implementiert). Elgamal unterliegt keinem Patent (vgl. Elgamal-Verschlüsselungsverfahren). |
| **RSA** |
| RSA (nach den Personen Rivest, Shamir und Adleman) ist ein asymmetrisches kryptographisches Verfahren, das sowohl zur Verschlüsselung als auch zur digitalen Signatur verwendet werden kann. Es verwendet ein Schlüsselpaar, bestehend aus einem privaten Schlüssel, der zum Entschlüsseln oder Signieren von Daten verwendet wird, und einem öffentlichen Schlüssel, mit dem man verschlüsselt oder Signaturen prüft. Der private Schlüssel wird geheim gehalten und kann nur mit extrem hohem Aufwand aus dem öffentlichen Schlüssel berechnet werden (vgl. RSA-Kryptosystem). |

Die Verschlüsselung von GoldBug ist derart gestaltet, dass jeder Nutzer mit jedem Nutzer kommunizieren kann, egal, welchen Verschlüsselungsalgorithmus ein Nutzer gewählt hat. Kommunikation zwischen den Nutzern mit verschiedenen Schlüsseltypen ist somit gut definiert, wenn die Knoten gemeinsame Versionen der libgcrypt und libntru Bibliotheken nutzen: Wer einen RSA-Schlüssel gewählt hat, kann also auch mit einem Nutzer verschlüsselt chatten und e-mailen, der einen Elgamal-Schlüssel gewählt hat. Dieses liegt daran, dass jeder jeden Algorithmus unterstützt und die Bibliothek dieses unterstützt. Wer das Programm mit einem Freund testen will, nutzt am besten die jeweils aktuellste Version von GoldBug.

Abbildung: Anpassbare Crypto




Natürlich kann jeder Nutzer in GoldBug seine

- individuelle Schlüsselgröße einstellen,
- die "Cipher",
- den "Hashtype",
- ferner "Iteration Count",
- und die kryptographische Salz-Länge

.. es sind für die Erstellung der Schlüssel und für die Verschlüsselung oftmals typische Parameter.

Der Vorteil ist, dass jeder Nutzer dieses individuell für sich definieren kann. Andere Applikationen - selbst quell-offene Anwendungen - erlauben es dem Nutzer kaum, diese entscheidenden Werte für das Verschlüsselungsverfahren selbst zu bestimmen.

## 2.2 Anwendung der Block Cipher Modi & Encrypt-then-MAC

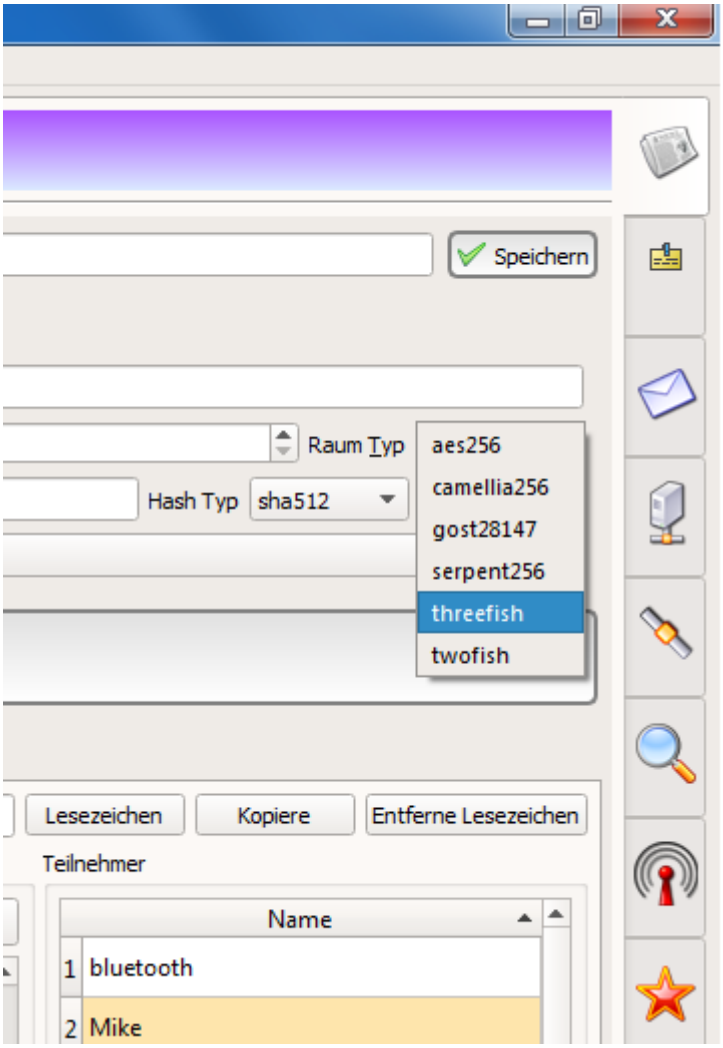
GoldBug hat also nicht nur zukunftsweisende Algorithmen oder zahlreiche Details wie die Umstellung von AES-128 auf AES-256 oder die Nutzung von sehr hohen, weil nötigen, Schlüsselgrössen standardisiert, sondern auch die fachgerechte Integration von etablierten und neuen Verschlüsselungsprozessen umgesetzt.

GoldBug verwendet CBC mit CTS, um die Vertraulichkeit zu schaffen. Der Dateiverschlüsselungsmechanismus unterstützt den Galois/Counter Mode (GCM)-Algorithmus ohne die Eigenschaft zur Authentizität, die durch den Algorithmus zur Verfügung gestellt wird. Um die Authentizität zu bieten, verwendet die Anwendung den methodischen Ansatz von “Erst verschlüsseln-dann-MAC” (Encrypt-then-MAC, ETM). MAC steht für Message Authentication Code - und bedeutet, dass die Reihenfolge determiniert ist: erst verschlüsseln, dann die Nachricht mit einem Code authentifizieren. Die Dokumentation beim source code zum Abschnitt der verschlüsselten und authentifizierten Container enthält dazu weitere technische Details.

Ein weiteres Beispiel für diese Innovation ist die Implementierung des ThreeFish-Hashes, der als Alternative zu SHA-3 zur Auswahl stand, als man erkannte, dass SHA-1 nicht mehr den zukünftigen Anforderungen gewachsen ist.

Threefish ist eine Blockverschlüsselung, die als Teil des Entwurfs der kryptographischen Hashfunktion Skein, welche an dem NIST-Auswahlverfahren zu SHA-3 teilnahm, entwickelt wurde. Threefish verwendet keine S-Boxen oder andere Lookup-Tabellen um zeitliche Seitenkanalattacken (Rechenzeitangriffe) zu erschweren.

Abbildung: Theefish Implementierung




Viele weitere Beispiel - insbesondere im Vergleich zu anderen Messengern - lassen sich finden, die belegen, dass die Verschlüsselungsprozesse in GoldBug sehr dem State-of-the-Art entsprechen.

## 2.3 Hybrides Verschlüsselungs-System

GoldBug implementiert ein Hybridsystem für die Verschlüsselung, inklusive Authentizität und Vertraulichkeit. Hybride heisst zunächst einmal: “beide Varianten sind vorhanden” und können miteinander kombiniert werden. So kann eine Nachricht zuerst mit oben dargestellter PKI a-symmetrisch verschlüsselt werden und sodann auch symmetrisch mit einem AES nocheinmal. Oder umgekehrt. Es ist aber auch noch eine weitere Variante denkbar. Der PKI Übertragungsweg überträgt mit permanenten Schlüsseln nochmals nur temporär eingesetzte Schlüssel, mit denen dann die weitere Kommunikation über diesen temporären Kanal erfolgt. Auch der temporäre Kanal kann dann wiederum eine erneute symmetrische Verschlüsselung mit einem AES vornehmen. Somit besteht nicht nur im Methodenwechsel von PKI zu AES bzw. von asymmetrischer Verschlüsselung zu symmetrischer Verschlüsselung eine Möglichkeit, ein Hybridsystem zu bilden, sondern auch in der Anwendung von permanenten PKI Schlüsseln zur Nutzung von temporären PKI Schlüsseln.

Vielfach zu verschlüsseln und zwischen diesen Methoden bzw. zeitlich limitierten Schlüsseln zu wechseln, ist eine starke Kompetenz von GoldBug in dieser Multi-, Vielfach- und hybriden

Verschlüsselung. Mit diesen Möglichkeiten kann man nun spielen und sie in verschiedensten Weisen anwenden. Wird nun zuerst der permanente oder der temporäre Schlüssel angewandt oder zuerst der symmetrische und dann der asymmetrische nochmals als zweite Ebene der Verschlüsselung? oder alles umgekehrt?

Ein Teil des Systems bei GoldBug erzeugt also pro Nachricht den Schlüssel für die Authentifizierung und die Verschlüsselung. Diese beiden Schlüssel werden für die Authentifizierung und das Einkapseln von Daten (also der Nachricht) verwendet. Die zwei Schlüssel (für die Authentifizierung und die Verschlüsselung) sind dann über den Public-Key-Teil des Systems eingekapselt bzw. angewandt. Die Anwendung bietet darüber hinaus auch einen Mechanismus zur Verteilung von Sitzungs-Schlüsseln für diese Datenkapselung (bzw. Verschlüsselung der Nachricht) - wie oben beschrieben, die temporären Schüssel. Wiederum werden die Schlüssel über das Public-Key-System eingekapselt und übertragen: ein zusätzlicher Mechanismus ermöglicht die Verteilung der Sitzungs-Schlüssel über die vorher festgelegten Schlüssel. Und digitale Signaturen können ebenso wahlweise auf die Daten angewendet werden.

Als ein Beispiel, mag dieses Format der folgenden Nachrichtenverschlüsselung dienen:

```
EPUBLIK Key
(Chiffrierschlüssel || Hash Key)
|| EEncryption Key (Data)
|| HHash Key (EEncryption Key (Data)).
```

Wer sich erstmalig mit Verschlüsselung beschäftigt, für den ist obiges Beispiel der Einkapselung ein erstes Beispiel, um weitergehend zu lernen und um die Methoden zu verstehen; - in jedem Fall aber sieht man, wie der Chiffrierschlüssel noch um den Hash-Key (vgl. MAC) ergänzt ist und auch die Daten in verschiedenen Verschlüsselungs-Ebenen eingebettet werden.

Nicht-NTRU private Schlüssel werden auf Richtigkeit durch die gcry_pk_testkey() Funktion ausgewertet. Der öffentliche Schlüssel muss auch einige grundlegende Kriterien erfüllen, wie beispielsweise den Einschluss der Public-Key-Kennung.

Die Authentifizierung des privaten Schlüssels und der Verschlüsselungsmechanismus ist identisch mit dem Verfahren, wie er in der Dokumentation beim Quelltext in dem Abschnitt zum verschlüsselten und authentifizierten Container weitergehend technisch erörtert ist.

Stellen wir jedoch noch einen einfacheren Fall dar und gehen nochmal etwas ausführlicher auf die symmetrische Verschlüsselung mit einem AES-Passwort ein, das die PKI Verschlüselung wie folgt ergänzen kann:

## 2.4 Symmetrische Verschlüsselung mit AES

Bei der symmetrischen Verschlüsselung wird AES eingesetzt - quasi ein 32 Zeichen langes Passwort, das durch Zufallsprozesse generiert wird. Da alle Zeichen und Sonderzeichen bei der Generierung eingesetzt werden, ist die Möglichkeitsmenge ebenso ausreichend groß, dass selbst schnelle Maschinen nicht innerhalb kurzer Zeit alle Varianten ausprobieren können. Während die asymmetrische Verschlüsselung ein öffentliches und privates Schlüsselpaar nutzt, ist es bei der symmetrischen Verschlüsselung also eine geheime Passphrase, die beide Teilnehmer kennen müssen (daher symmetrisch genannt bzw. wird dieses für GoldBug später auch noch in der Gemini-Funktion (von griechisch “Zwilling” abgeleitet) angesprochen: Beide Seiten müssen die geheime Passphrase austauschen und kennen).

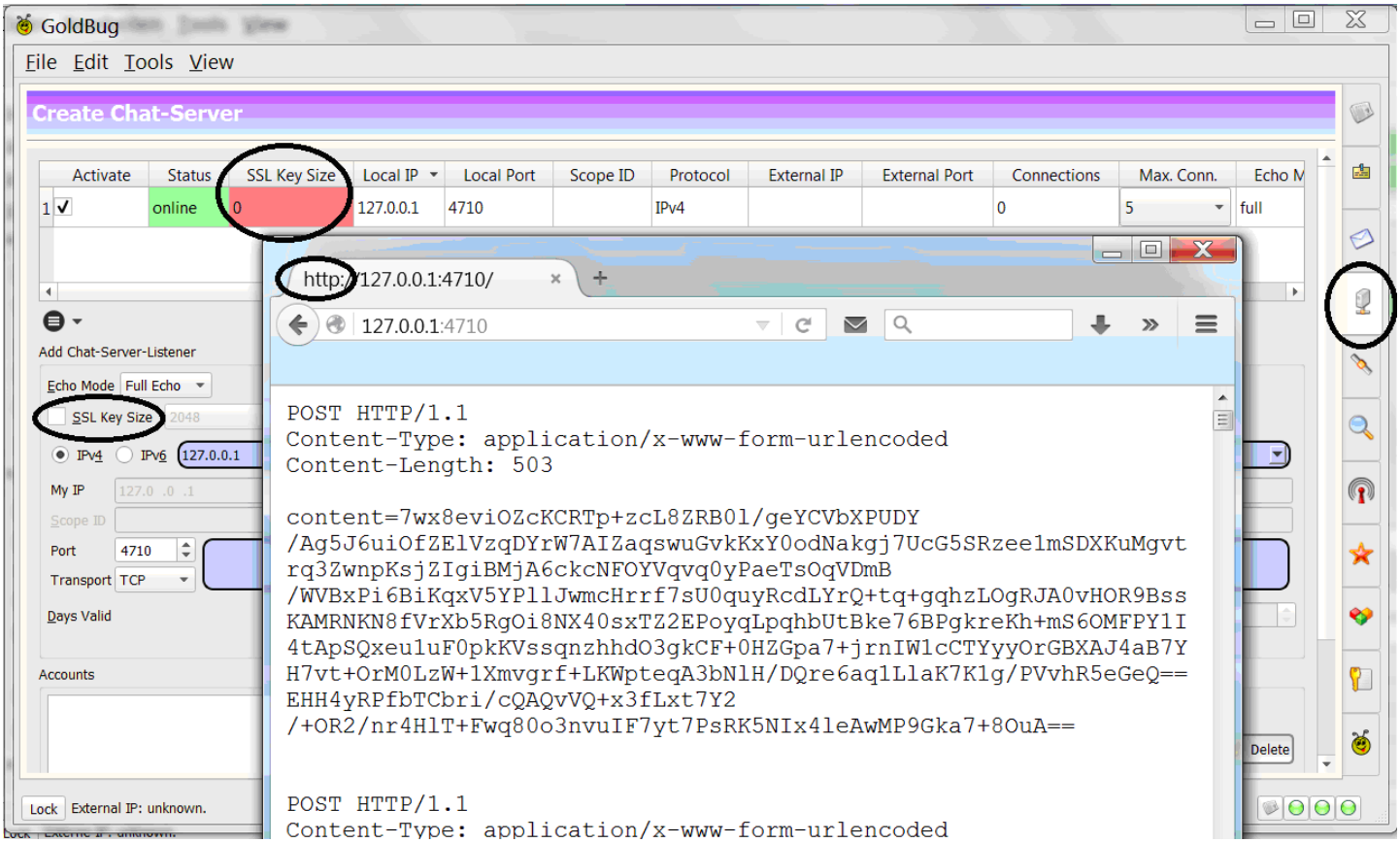
GoldBug nutzt somit wie oben beschrieben beide Standards: asymmetrische Schlüssel und/oder symmetrisch verschlüsselte Nachrichten werden durch SSL/TLS (also asymmetrisch) verschlüsselte Verbindungen gesandt, und auch die asymmetrisch verschlüsselte Nachricht kann ggf. noch zusätzlich mit einer symmetrischen Verschlüsslung (AES) abgesichert werden. Dann nutzt GoldBug sogar drei Ebenen von Verschlüsselung wie dieses Beispiel der Einkapselung nochmals (vereinfacht, da ohne HASH/MAC bzw. Signatur dargestellt,) verdeutlicht:

```
RSA-SSL/TLS (AES (Elgamal (Nachricht)))
```

Übersetzung dieser Formel: Erst wird die Textnachricht mit dem öffentlichen (asymmetrischen) Schlüssel des Freundes über den Elgamal-Algorithmus verschlüsselt, sodann wird der verschlüsselte Text nochmals mit einem AES-Algorithmus (symmetrisch) (ein zweites Mal) (passwort-)verschlüsselt (und abgesichert) und diese Kapsel wird dann durch die bestehende mit SSL/TLS (unter Nutzung von RSA) verschlüsselte (asymmetrische) Verbindung zum Freund gesandt.

Wird ein HTTP-Listener eingerichtet und die verschlüsselte Nachrichtenkapsel also nicht über HTTPS - also der dritten Verschlüsselungsschicht, über eine SSL/TLS-Verbindung gesandt, kann der Ciphertext der Nachrichtenkapsel auch im Browser eingesehen werden. Es zeigt sich, dass selbst bei zwei Verschlüsselungsschichten nur Ciphertest versandt wird (vgl. Abbildung aus der Praxisdemo von Adams/Maier 2016).

Abbildung: Ciphertext




Auch besteht die Möglichkeit, die symmetrische Passphrase (das AES) mit der Gegenstelle auszutauschen über die etablierte asymmetrische (SSL/TLS-)Verschlüsselung. Die Passphrase kann automatisch generiert oder auch manuell definiert werden, wie wir weiter unten bei der Gemini- bzw. Call-Funktion (“Cryptographisches Calling”, was mit GoldBug (bzw. dem Spot-on Kernel Projekt) eingeführt wurde) noch weiter sehen werden. Es gibt kaum andere - zudem quelloffene - Applikationen, die eine (durchgängige) Ende-zu-Ende Verschlüsselung vom einen Teilnehmer zum anderen Teilnehmer inkludieren, in der der Nutzer die Passphrase (z.B. einen AES-String) manuell und individuell definieren kann.

Eine (symmetrische) Ende-zu-Ende Verschlüsselung ist also zu differenzieren von der Punkt-zu-Punkt-Verschlüsselung. Daher wird auch gerne das Wort “durchgängige” Ende-zu-Ende-Verschlüsselung hinzugefügt (besser noch: durchgängige symmetrische Ende-zu-Ende-Verschlüsselung) - denn es geht ja darum, dass nur die Teilnehmerin Alice und der Teilnehmer Bob die geheime Passphrase kennen. Eine Punkt-zu-Punkt-Verschlüsselung läge vor, wenn Alice zum Server und dann der Server zu Bob die Verbindung herstellt. Hierbei kann es sein, dass der Server die Nachricht lesen kann, sie also auspackt und wieder einpackt, insbesondere wenn ein asymmetrischer Schlüssel zwischen den Teilnehmern und dem in der Mitte angesiedelten Server besteht.

GoldBug bietet stattdessen eine durchgängige symmetrische End-zu-Ende Verschlüsselung an, die nicht nur manuell definiert werden kann, sondern mit einer Automation auch instant, jederzeit erneuert werden kann (das wird “Cryptographisches Calling” genannt, siehe unten).

Diese spezielle Art der Mischung von PKI und AES sowie Transfer über eine SSL/TLS-Verbindung - bzw. eine Besonderheit beim Auspacken der verschlüsselten Kapsel - wird als Echo-Protokoll

bezeichnet, das im folgenden Abschnitt nochmals vertieft werden soll, denn es beinhaltet noch eine weitere Charakteristik beim Versand in das Netzwerk. Was ist also das genau spezifische am Echo-Protokoll?

# 3 Was ist das Echo-Protokoll?

Mit dem Echo-Protokoll ist - einfach ausgedrückt - gemeint, dass

- erstens jede Nachrichten-Übertragung verschlüsselt ist...

Beispiel: SSL ( AES ( RSA* (Nachricht)))

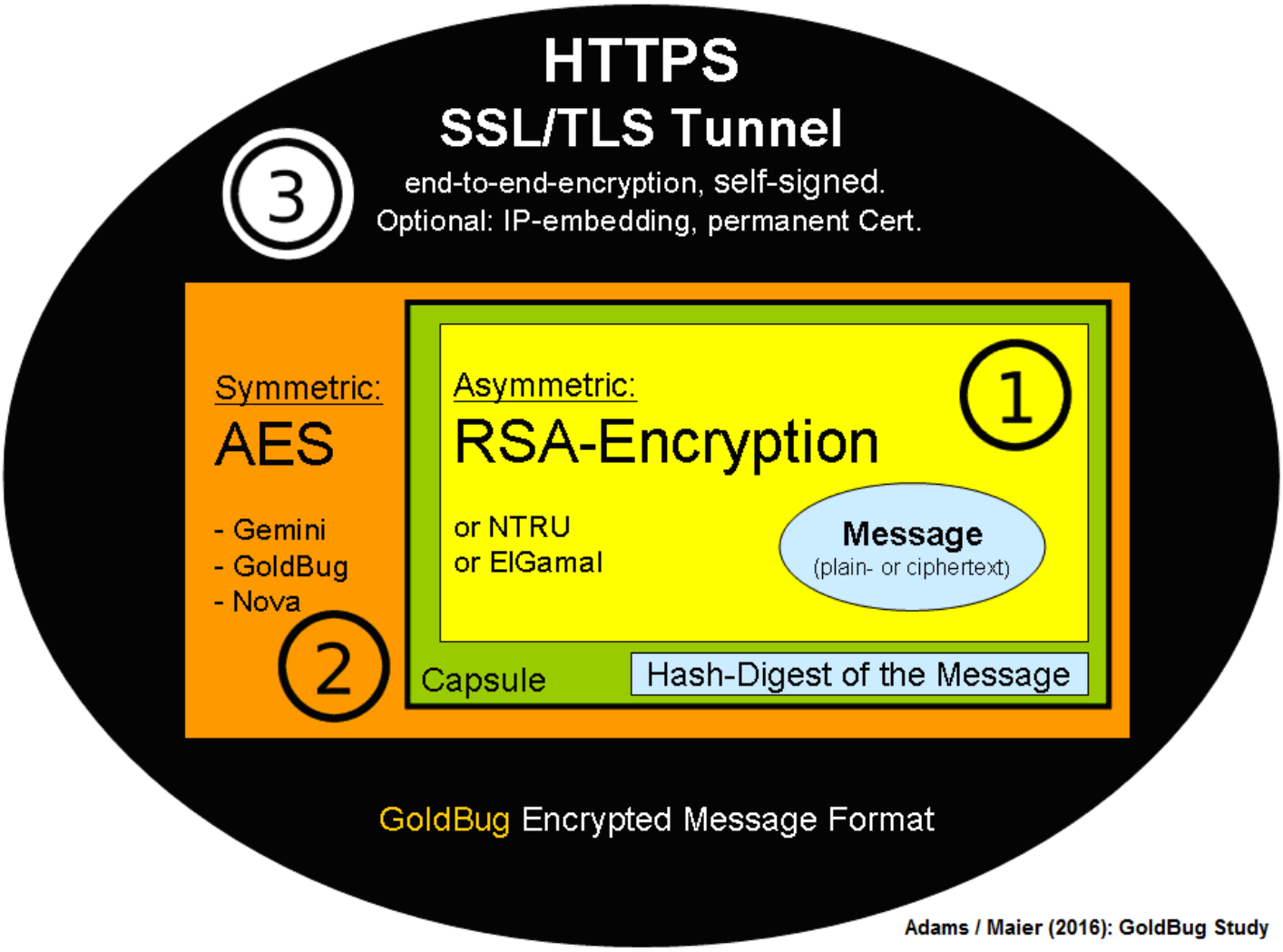
*) anstelle von RSA kann ebenso Elgamal oder NTRU oder McEliece genutzt werden,

- ... und zweitens im Echo-Netzwerk jeder Verbindungsknoten jede Nachricht an jeden verbundenen Nachbarn sendet. Punkt. So einfach ist die Welt.

Zugrunde liegt das sogenannte "Kleine-Welt-Phänomen": Jeder kann jeden über sieben Ecken in einem peer-to-peer oder friend-to-friend Netzwerk irgendwie erreichen - oder aber einfach über einen im Freundeskreis installierten gemeinsamen Echo-Chat-Server die Nachricht verteilen.

- Als drittes Kriterium für das Echo-Protokoll lässt sich anfügen, dass eine Besonderheit beim Auspacken der verschlüsselten Kapseln besteht: Die Kapseln haben weder - und hier unterscheiden sie sich von TCP-Paketen - einen Empänger noch einen Absender. Die Nachricht wird über den Hash der unverschlüsselten Nachricht identifiziert, ob sie für den Empfänger in der Benutzeroberfläche angezeigt und lesbar werden soll oder nicht. Doch zu diesem sog. **Echo-Match** noch weiter unten ausführlicher.

Abbildung: Format des genutzten Echo-Protokolls




Die Abbildung (nach Adams/Maier 2016) zeigt von innen nach außen den Prozess, wie die verschlüsselte Kapsel im Rahmen des Echo-Protokolls gebildet wird:

Erste Ebene der Verschlüsselung: Die Nachricht wird verschlüsselt und der Ciphertext der Nachricht wird gehashed und sodann können mit dem asymmetrischen Schlüssel (z.B. des Algorithmus RSA) auch die symmetrischen Schlüssel verschlüsselt werden. In einem Zwischenschritt werden also der verschlüsselte Text und der Hash-Digest der Nachricht in eine Kapsel gebündelt und zusammen gepackt. Es folgt dem Paradigma: Encrypt-then-MAC. Um dem Empfänger zu beweisen, dass der Ciphertext nicht verfälscht wurde, wird der Hash-Digest zuerst gebildet, bevor der Ciphertext entschlüsselt wird.

Dritte Ebene der Verschlüsselung: Sodann kann diese Kapsel über ein gesicherte SSL/TLS-Verbindung zum Kommunikationspartner übertragen werden.

Zweite Ebene der Verschlüsselung: Optional besteht noch die Möglichkeit, die Kapsel der ersten Ebene zusätzlich mit einem AES-256, also vergleichbar mit einem gemeinsam geteilten, 32-Zeichen langem Passwort symmetrisch zu verschlüsseln. Hybride Verschlüsselung wird sodann zur Mehrfach-Verschlüsselung ergänzt (vgl. Adams/Maier 2016:46).

Der **Modus des “Halben Echos”** sendet eine Nachricht nur einen Hop, d.h. z.B. von Bob zu Alice. Alice sendet die Nachricht dann nicht mehr weiter (wie es beim vollen Echo der Standard

ist).

Neben **Vollem Echo**, Halben Echo gibt es drittens noch das **Adaptive Echo (AE)**. Hier wird die Nachricht nur an Nachbarn oder Freunde versandt, wenn diese einen Verschlüsselungs-Token kennen, diesen also zuvor eingespeichert haben. Wer den Token nicht kennt, an den wird die Nachricht nicht weitergeleitet.

Schließlich kennt das Echo noch **Echo Accounts**. Eine Art Firewall. Hiermit kann sichergestellt werden, dass nur Freunde, die den Account-Zugang kennen, sich verbinden können. So kann ein Web-of-Trust erstellt werden, also ein Netzwerk ausschließlich unter Freunden. Es basiert nicht auf dem Schlüssel für die Verschlüsselung, sondern ist davon unabhängig. D.h. der Nutzer muss nicht seinen öffentlichen Schlüssel auch noch mit seiner IP-Adresse verknüpfen oder gar im Netzwerk bekannt geben.

Grundsätzlich sendet im Echo also jeder Knoten eine Nachricht an jeden verbundenen Knoten: Wenn ein Nutzer dann eine Nachricht ein zweites Mal erhalten sollte, so wird sie in einem temporären Zwischenspeicher verglichen (anhand des Hashwertes für diese Nachricht) und ggf. bei Bekanntsein des Hashes wieder verworfen und somit nicht weitergeleitet. Dieses Vorgehen nennt sich "Congestion Control" und balanciert die Anzahl der Nachrichten im Netzwerk von mehreren Knotenpunkten oder Servern.

> **Eine kleine Analogie:** Die Kryptographie des Echo-Protokolls kann verglichen werden mit dem Geben und Nehmen von Überraschungseiern, einer Kapsel mit zusammenbaubarem Mini-Spielzeug in dem bekannten Schokoladen-Ei. Bob gibt ein Überraschungsei an Alice, Alice öffnet es und verzehrt die Schokolade und stößt auf die Plastik-Kapsel im Inneren des Überraschungsei und versucht, diese zu öffnen und die darin befindlichen Teile zu einem Spielzeug, einem Schlumpf, zusammen zu bauen. Der Zusammenbau gelingt ihr aber nicht, der Schlumpf lässt sich nicht bilden und daher packt sie die Einzelteile wieder in die Plastik-Kapsel, gießt neue Schokolade drum rum und gibt das Ei an den Nachbarn weiter, der ebenso versucht, einen Schlumpf aus den Teilen zu basteln. Alice weiß nicht, wer das Überraschungsei bzw. den Schlumpf erfolgreich zusammenbauen kann, daher kopiert sie es (- welch ein Wunder, Alice hat eine Ü-Ei-Kopiermaschine -) und gibt jeweils eine Kopie an *alle* ihre Freunde weiter. (Auspacken, basteln, anschauen, einpacken, verschenken und wieder auspacken, basteln, anschauen, einpacken, verschenken, und so fort ..
>
> Aus Sicht der im Netzwerk vertretenen Instanzen (Kernels) wäre in diesem Bild das Netz zum Ü-Ei-Paradies geworden, wenn nicht mit Congestion Control die Bastelvorgänge wieder reduziert würden. Einmal bekannte Bastel-Teile werden nicht ein zweites Mal zusammen gebaut. Alice bastelt so lange, bis sie einen Schlumpf mit roter Mütze erkennen kann, sie hat die für sie bestimmte Figur des Papa-Schlumpfs bzw. ihre Nachricht erhalten.

Um im Internet bzw. Echo-Netzwerk Zeit- und Häufigkeitsanalysen auszuschliessen, gibt es in GoldBug noch weitere Funktionen, die die Verschlüsselung erhöhen bzw. eine Crypto-Analyse

erschweren:

So zum Beispiel: kann man mit der GoldBug Applikation ebenso unechte Nachrichten (“Fakes” aus der Simulacra-Funktion) und simulierte Kommunikationsnachrichten (“Impersonated Messages”) aussenden. Einmal ist die Verschlüsselung keine Verschlüsselung, sondern stellt pure Zufallszeichen dar, die von Zeit zu Zeit ausgesandt werden, und das andere Mal wird eine menschliche Unterhaltung simuliert, die ebenso nur auf durcheinander-gewürfelte Zufallszeichen beruht:

**Simulacra:** Diese Funktion sendet bei Aktivierung der Check-Box eine “simulierte” Chat-Nachricht ins Echo Netzwerk. Diese „Fake“-Nachricht besteht aus reinen Zufallsziffern und macht es Analysten schwerer, verschlüsselte Nachrichten mit echten und zufälligen Nachrichten zu unterscheiden. Simulacrum ist ein Begriff, der sowohl aus dem Film “[Matrix]”(https://de.wikipedia.org/wiki/Matrix_(Film)) als auch in der Philosophie Baudrillards nicht unbekannt ist (Neos Aufbewahrungsort für Software in seiner Wohnung ist das Buch “Simulacres et Simulation” des französischen Medienphilosophen Jean Baudrillard, das das Verhältnis von Realität, Symbolen und Gesellschaft untersucht). Einige Jahre nach der Veröffentlichung des Echo-Protokolls haben ähnliche Geldgeber wie für das Tor-Netzwerk eine Software entwickelt namens Matrix Dot Org, die ähnlich dem Echo-Protokoll verschlüsselte Kapseln ins Netzwerk sendet und auch eine Messaging Funktion adressiert; eine Analyse ist ausstehend, wo das Echo gegenüber der Plagiats-Architektur Unterschiede und Vorteile bietet oder quelloffene Anregungen bot.

**Impersonator:** Neben zufälligen Fake-Nachrichten kann mit dem GoldBug-Programm auch ein Chat simuliert werden, als wenn eine echte Person von Zeit zu Zeit chattet und Antworten aussendet. Auch diese Nachrichten sind mit reinen Zufallsdaten gefüllt, variieren jedoch - simuliert an einer echten Chat-Unterhaltung. So kann die Analyse von Nachrichten erschwert werden, wenn dritte Aufzeichner (“Recorder”) sämtliche Kommunikation der Nutzer zwischen-speichern und aufzeichnen sollten, was ggf. anzunehmen ist. Aber mehr noch: Auch das Ausbleiben von Meta-Daten (vgl. Vorratsdatenspeicherung) gibt keinen Anlass zu vermuten, dass eine Nachricht für den Nutzer gewesen sei. Wer eine Nachricht erfolgreich auspacken konnte, der sendet sie normalweise nicht erneut ins Echo-Netz. Ein Aufzeichner von Metadaten könnte gesteigertes Interesse an den nicht weitergeleiteten Nachrichten haben, in der Annahme, dass diese Nachricht sodann vom Nutzer erfolgreich dekodiert worden sein könnte. Für diesen Fall gibt es auch die Option des SuperEchos:

**SuperEcho:** Diese Funktion leitet auch erfolgreich dekodierte und damit lesbare Nachrichten wieder eingepackt weiter an alle Freunde. Das Ausbleiben der Weitersendung der Nachricht kann dann beim SuperEcho nicht mehr darauf schließen lassen, dass die Nachricht ggf. erfolgreich dekodiert worden sein könnte.

SuperEcho, Simulacra und Impersonation sind somit drei Optionen des Programms GoldBug, die es Angreifern schwerer machen sollen, in der Vielzahl der Nachrichten die für den Nuzer (und anscheinend auch für Andere) interessante Nachrichten nachzuvollziehen.

Schauen wir uns nun die einzelnen Echo-Modi einmal genauer an:

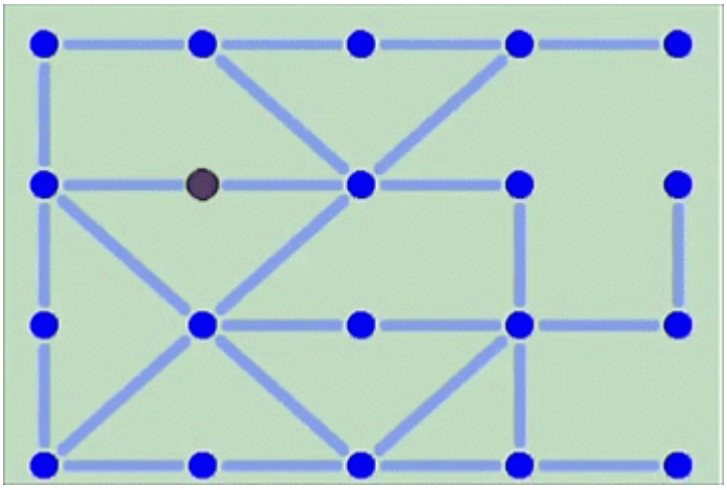
## 3.1 Volles Echo

Das **volle Echo** (“Full Echo”) legt die Annahme zugrunde, wie sie auch beim sogenannten “Kleine-Welt-Phänomen” getroffen wird: über wenige Freunde kann jeder jedem eine Nachricht zukommen lassen. Irgendwie kennt jeder jeden über maximal sieben Ecken. Dieses wird auch in einem peer-to-peer bzw. friend-to-friend Netzwerk angenommen. Daher kann man jeden erreichen, wenn jeder Knotenpunkt jede Nachricht an alle weiteren bekannten Knotenpunkte sendet (siehe Abbildung).

Alternativ kann man diesen dezentralen Anspruch unterstützen bzw. die Nachrichtenwege abkürzen, indem man für seine Freunde einen Chat-Server basierend auf dem Echo-Kernel installiert, so dass sich alle verschlüsselten Nachrichten darüber an die Teilnehmer versenden lassen und der Server auch als E-Mail-Postfach dienen kann.

Die Abbildung simuliert das Versenden der Nachricht von einem Ausgangspunkt an alle Netzknoten über alle verbundenen Netzknoten.

Abbildung: Echo-Simulation: Jeder Knotenpunkt sendet an jeden verbundenen Knotenpunkt



Grundlegend ist also, dass im Echo jeder Knotenpunkt jede Nachricht an jeden Knotenpunkt weitersendet. Dieses klingt einfach dahingesagt, ist es einerseits auch: Das Echo-Protokoll ist ein sehr simples Protokoll, hat aber auch weitergehende Implikationen, sprich: Es gibt beim Echo keine Routing Informationen und auch Metadaten können aus dem Netzwerk kaum aufgezeichnet werden. Die Nodes leiten die Nachricht auch nicht weiter, der Begriff des “Forwarding” ist unzutreffend, denn jeder Knotenpunkt sendet aktiv erneut die Nachricht an die (seine) verbundenen Freunde.

Damit kann es vorkommen, dass man eine Nachricht (von mehreren verbundenen Knotenpunkten) mehrfach erhält - damit dieses jedoch nicht passiert und effizient gehalten wird, wird der Hash der Nachricht in einem Cache zwischengespeichert und die Nachricht möglicherweise für eine Weitersendung zurückgewiesen, wenn sie als Doublette identifiziert wurde. Dieses nennt man wie oben schon angedeutet: “Congestion Control”.

Die Nachricht ist sozusagen in einer Kapsel, ähnlich einer ZIP-Datei. Diese Kapsel wird durch die asymmetrische Verschlüsselung mit dem öffentlichen Schlüssel erstellt. Hinzugepackt wird noch der Hash der Plaintext-Nachricht. Wenn ein Node nun versucht, den Ciphertext zu dekodieren, kommt ein neuer Text heraus - der entweder richtig oder falsch dekodiert sein kann, sprich er ist für den Menschen lesbar oder aus Zufallszeichen wurden bei falschem Dekodierschlüssel wiederum nur Zufallszeichen. Dieser entstehende Text nach dem Dekodierungsversuch wird also wiederum gehashed.

Wenn nun der Hash der dekodierten Nachricht identisch ist mit dem Hash der Original-Nachricht, den der Sender der Kapsel schon beigefügt hatte, ist deutlich, dass der entschlüsselnde Node den richtigen Schlüssel benutzt hat und diese Nachricht im Plaintext für ihn ist: Die Nachricht ist lesbar und wird in der Benutzeroberfläche angezeigt. Dieses kann als **Echo-Match** bezeichnet werden. Erfolglose Decodierungsversuche, bei denen also der Hashwert zwischen Originalnachricht und Nachrichtentext des Dekodierungsversuches nicht übereinstimmen, werden nicht in der Benutzeroberfläche angezeigt, sondern verbleiben im Kernel des Programms für eine weitergehende Aussendung an die verbundenen Nachbarn.

Der Node muss also mit allen Schlüsseln seiner Freunde versuchen, die Nachricht zu entpacken und die Hashwerte vergleichen. Ist der Hashwert nicht identisch, verpackt der Node die Bestandteile wieder in einer Kapsel zusammen und sendet sie jeweils weiter an seine verbundenen Freunde, die dann gleiches versuchen.

Der Hashwert einer Nachricht ist nicht invertierbar, daher kann mit dem (beigepackten) Hash der Original-Nachricht die Verschlüsselung nicht gebrochen werden, es bedarf weiterhin des richtigen Schlüssels.

Eine Nachricht, die erfolgreich entpackt wurde, wird nicht mehr weiter gesendet, es sei denn, man nutzt die Option des Super-Echos, bei dem auch die erfolgreich entpackten Nachrichten weiter gesandt werden. So kann niemand, der die Internetpakete aufzeichnet, nicht weiter gesandte Nachrichten identifizieren.

Schließlich, wie oben beschrieben, kann man auch von Zeit zu Zeit Falsch-Nachrichten aussenden (“Simulacra fake messages”) und auch simulierte Unterhaltungsnachrichten (impersonated messages), so dass es Aufzeichnern von Netzwerkverkehr schwierig gemacht wird, die Nachrichten-Kapsel herauszufinden, die für eine eigene Lesbarkeit interessant gewesen wären. Denn es ist heutzutage zu beachten, dass möglicherweise davon auszugehen ist, dass alle Kommunikationsdaten eines Internetnutzers irgendwo im Internet gespeichert und aufgezeichnet und im Interessensfall auch automatisiert und manuell ausgewertet werden.

Sodann: Diese verschlüsselte Kapsel wird wiederum über einen verschlüsselten SSL/TLS Kanal gesandt, der zwischen den Knotenpunkten aufgebaut ist. Hierbei handelt es sich um eine dezentrale, selbstsignierte p2p Verbindung, ein “two-pass mutual authentication protocol”. Die Implementierung ist präzise definiert nach SSL/TLS, das man aber auch abschalten kann: Die Netzwerkknoten kommunizieren also über HTTPS oder auch nur HTTP.

Wie auch immer, natürlich wird die Übertragung anfälliger, wenn man die mehrfache Verschlüsselung nicht einsetzt. Daher sollte man zu seinen Freunden immer eine HTTPS-Verbindung aufbauen und über diesen verschlüsselten Kanal seine verschlüsselten Kapseln senden, in denen die Nachricht darauf wartet, vom richtigen Schlüssel wachgeküsst und (mittels der Methode des Echo-Matches auf Basis des Hash-Vergleiches) in lesbaren Plaintext konvertiert zu werden.

| **Prozess-Beschreibung des Echo-Matches:** |
| --- |
| Absender A hashed seinen Originaltext zu einem Hash 123456789, verschlüsselt den Text und packt Krypto-Text und Hash der Originalnachricht in die Kapsel (bevor er noch ein AES draufsetzt und es durch eine TLS/SSL Verbindung raussendet). |
| Empfänger 1 konvertiert den erhaltenen verschlüsselten Text der Kapsel zu einem (vermeintlichen) Plaintext, dieser hat aber den Hash 987654321 und ist damit nicht identisch zum mitgelieferten Originaltext-Hash von 123456789. Dieses wird mit allen verfügbaren Schlüsseln aller Freunde des Empfängers 1 wiederholt, da alle Hash-Vergleiche jedoch erfolglos blieben, packt er die Nachricht wieder zusammen und sendet sie weiter. Die Nachricht ist offensichtlich nicht für ihn bzw. von einem seiner Freunde. |
| Empfänger 2 konvertiert nun ebenso den erhaltenen, verschlüsselten Text zu einem (vermeintlichen) Plaintext, dieser hat den Hash 123456789 und ist damit identisch zum mitgelieferten Originaltext Hash von 123456789, die Decodierung war augenscheinlich mit einem der vorhandenen Schlüssel seiner Freunde erfolgreich und daher wird die Nachricht auf dem Bildschirm dieses Empfängers angezeigt (und falls Super-Echo gewählt ist, auch wieder eingepackt und weiter gesandt). |

Niemand im Netz kann sehen, welche Nachricht ein Nutzer erfolgreich auspacken konnte, da alles auf der lokalen Maschine des Nutzers passiert.

## 3.2 Halbes Echo

Der **Modus des Halben Echos** (“Half Echo”) sendet die Nachricht des Nutzers nur einen Hop zum nächsten Knotenpunkt, z.B. von Bob zu Alice. Alice sendet die Nachricht dann nicht weiter auf den Weg ihrer verbundenen Freunde (wie es für das Volle Echo üblich ist). Der Echo Modus wird technisch über die Verbindung zu einem anderen Listener definiert: Bob`s Node teilt mit, wenn er sich zu dem Node von Alice verbindet, dass Alice die Nachricht nicht weiter an ihre Freunde senden soll. So können zwei Freunde bzw. Knotenpunkte über eine direkte Verbindung ausschließen, dass die Nachricht in das weitere Netz über die anderen, weiteren Verbindungen, die jeder Knotenpunkt hat, getragen wird.

Neben dem Vollen und Halben Echo gibt es drittens noch das **Adaptive Echo (AE)**. Hier wird, wie weiter unten noch beschrieben, die Nachricht nur dann an verbundene Nachbarn oder Freunde weiter gesandt, wenn der Knotenpunkt einen bestimmten kryptographischen Token

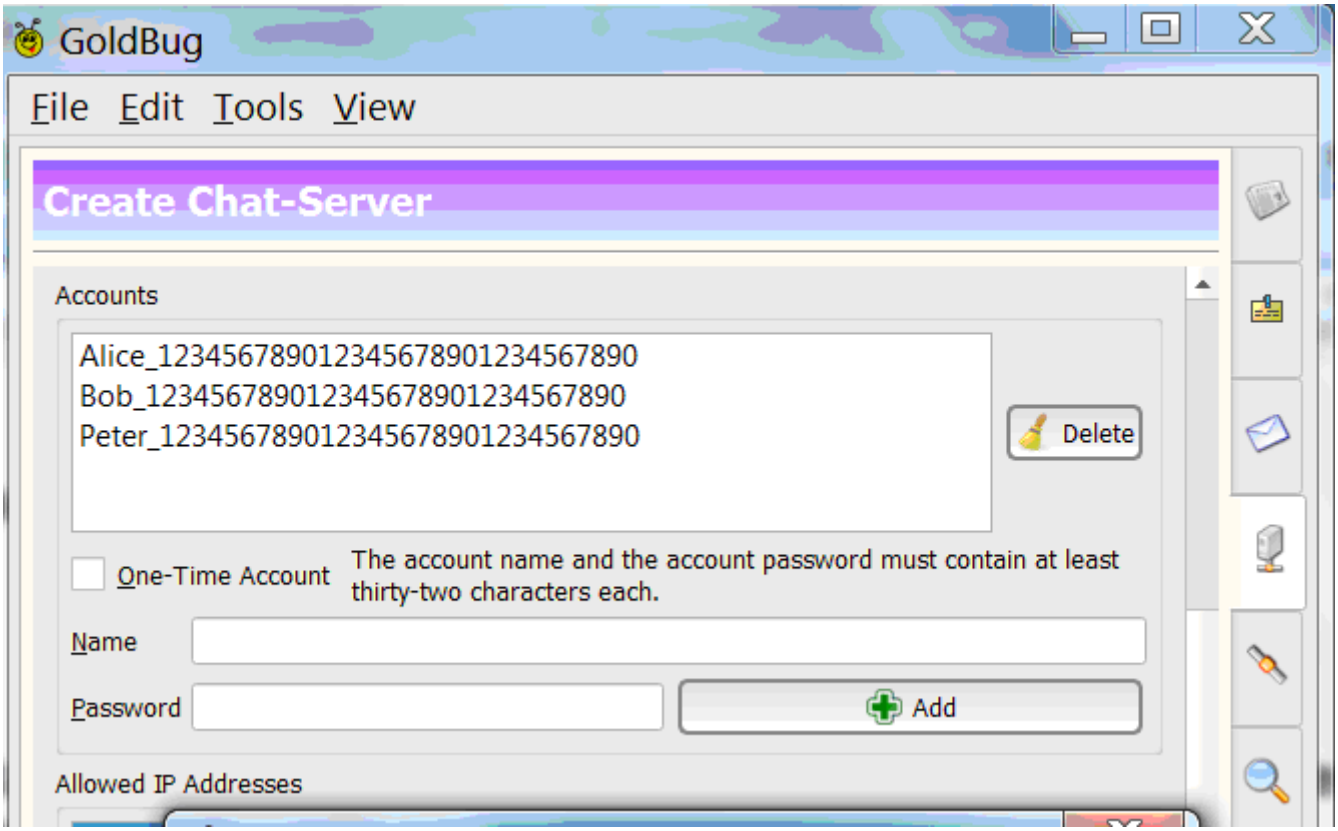
kennt - also ähnlich einer geheimen Passphrase. Diese Passphrase muss natürlich vorher definiert, geteilt und in den jeweiligen Knotenpunkten eingespeichert sein. So können definierte Wege einer Nachricht in einer Netzwerkkonfiguration genutzt werden. Beispiel: Wenn alle Knotenpunkte eines Landes eine gemeinsame Passphrase für das Adaptive Echo nutzen, wird die Nachricht niemals in den Knotenpunkten anderer Nationen erscheinen, wenn diese die Passphrase nicht kennen. So kann ein Routing definiert werden, dass nicht innerhalb der Nachricht verortet ist, sondern in den Knotenpunkten. Wer die Passphrase somit nicht kennt, bekommt die Nachricht auch nicht weitergeleitet! Mit dem Adaptiven Echo werden Nachrichten, die nicht geöffnet werden können, zu Nachrichten, die gar nicht bekannt oder existent sind.

Der Abschnitt weiter unten über das Adaptive Echo (AE) wird daher noch ausführlicher über diese Option berichten.

## 3.3 Echo Accounts

Und ergänzend: das Echo kennt auch **Echo-Accounts**. Ein Konto oder eine Art Firewall. Es kann genutzt werden, um sicherzustellen, dass nur Freunde sich verbinden, die die Zugangsdaten zu dem Konto kennen. Somit wird ein sogenanntes Web of Trust, ein Netzwerk, das auf Vertrauen beruht, gebildet. Es basiert nicht wie in anderen Applikationen auf dem Schlüssel für die Verschlüsselung, es ist davon unabhängig. Das hat den Vorteil, dass der öffentliche Schlüssel für die Verschlüsselung nicht mit der IP-Adresse assoziiert werden muss (wie es z.B. bei RetroShare der Fall ist); oder der Nutzer seine IP-Adresse im Netzwerk der Freunde bekannt geben muß, beispielsweise in einem DHT, in dem Nutzer danach suchen können. Die Echo-Accounts stellen eine Peer-to-Peer-(P2P)-Verbindung auf ein Friend-to-Friend (F2F) Netzwerk um bzw. erlauben beide Verbindungsarten. Damit ist GoldBug für beide Paradigma ausgelegt.

Abbildung: Abbildung: Account Firewall




Die Echo-Accounts funktionieren wie folgt:

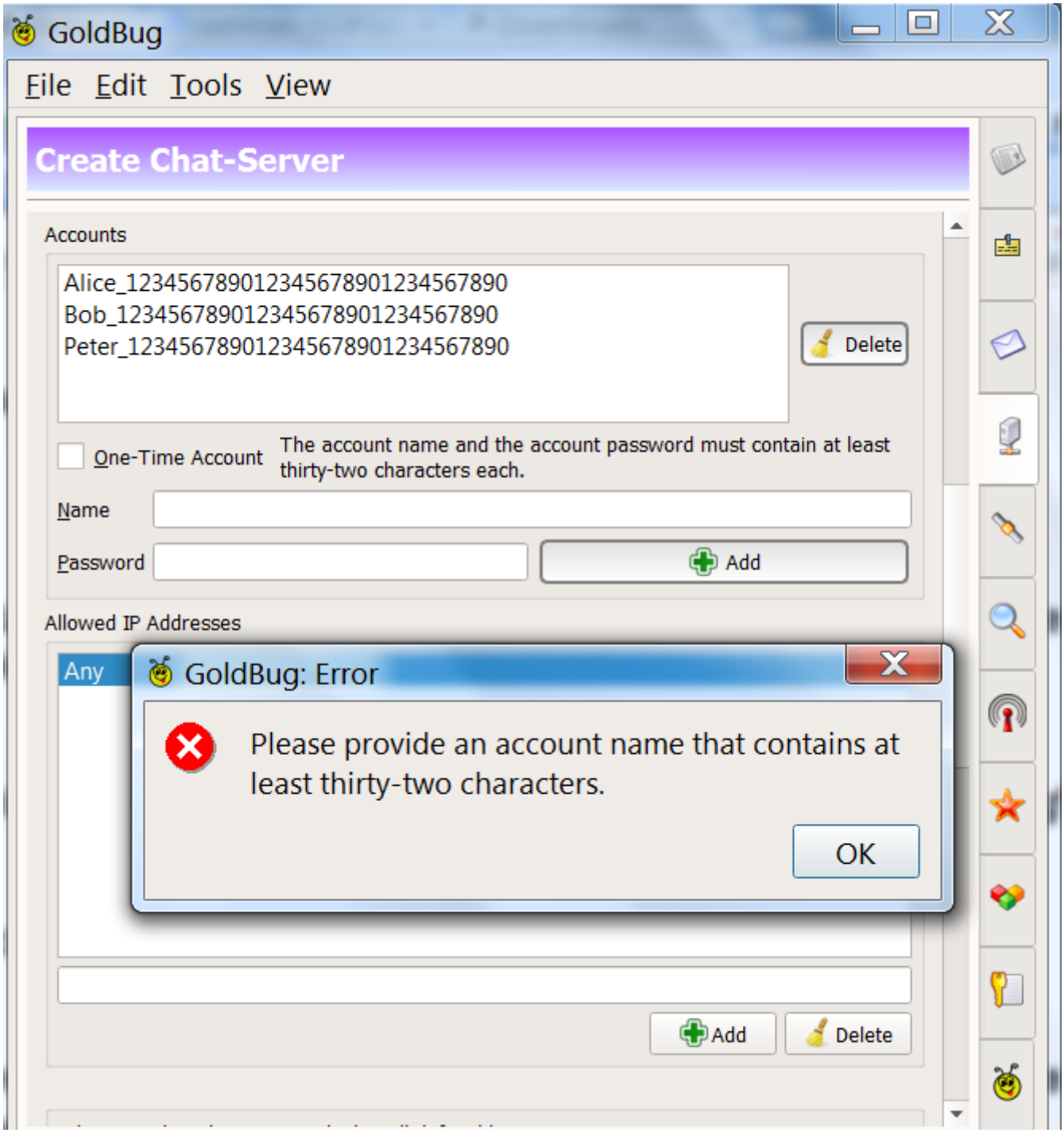
- Bindende Endpunkte sind verantwortlich für die Definition der Konto-Informationen. Während des Erstellungsprozesses für ein Konto kann dieses für eine einmalige Nutzung definiert werden (One-Time-Account oder One-Time-Use). Konto-Name und auch die Passphrase für das Konto erfordern wenigstens 32 Bytes an Zeichen. Ein langes Passwort ist also erforderlich.
- Nachdem eine Netzwerkverbindung erstellt worden ist, informiert der bindende Endpunkt den anfragenden Knotenpunkt mit einer Anfrage zur Authentifizierung. Der bindende Endpunkt wird die Verbindung abwerfen, wenn der Peer sich nicht innerhalb eines Zeitfensters von fünfzehn Sekunden identifiziert hat.
- Nachdem die Anfrage zur Authentifizierung erhalten worden ist, antwortet der Peer zu dem bindenden Endpunkt. Der Peer überträgt dann die folgenden Informationen: HHash Key (Salt / Time) // Salt, wobei der Hash Key eine konzertierte Zusammenfassung aus dem Konto-Namen und auch dem Konto-Passwort ist. Derzeit wird der SHA-512 Hash Algorithmus genutzt, um dieses Hash-Ergebnis zu generieren. Die Zeit-Variable hat eine Auflösung von wenigen Minuten. Der Peer behält den Wert für das kryptographische Salz.
- Der bindende Endpunkt erhält die Informationen des Peers. Folgerichtig prozessiert dieser dann HHash Key (Salt // Time) für alle Konten, die er eingerichtet hat. Wenn der Endpunkt kein Konto identifizieren kann, wartet er eine Minute und führt eine weitere Suche durch. Wenn ein auf diesen Hash-Key passendes Konto gefunden wurde, erstellt der bindende Endpunkt eine Nachricht ähnlich zu der, die der Peer im vorherigen Schritt erstellt hat und sendet die Informationen an den Peer. Die authentifizierte Information wird gespeichert. Nach einer Dauer von etwa 120 Sekunden wird die Information wieder gelöscht.

- Der Peer erhält die Information des bindenden Endpunktes und führt einen ähnlichen Validierungsprozess durch - dieses mal jedoch unter Einschluss der Analyse des kryptographischen Salz-Wertes des bindenden Endpunktes. Die beiden Salz-Werte müssen dann eindeutig übereinstimmend sein. Der Peer wird die Verbindung abwerfen, wenn der Endpunkt sich nicht selbst innerhalb eines fünfzehn-Sekunden-Zeitfensters identifiziert hat. Zu beachten ist, nebenbei bemerkt, dass das Account-System noch weiter ausgestaltet werden kann, indem ein Schlüssel zur Verschlüsselung einbezogen wird. Der zusätzliche Schlüssel erlaubt dann, noch genauere Zeitfenster zu definieren.

Wenn SSL/TLS während dieser Aushandlung nicht verfügbar ist, mag das Protokoll wie folgt angreifbar werden: Eine Zwischenstation mag die Werte aus dem dritten Schritt aufzeichnen und folgerichtig zum bindenden Endpunkt senden. Sodann könnte der bindende Endpunkt auch einer unbekannten Verbindung Zugang zum Account gewähren. Das aufzeichnende Gerät könnte dann die Antwort des bindenden Endpunktes, also die Werte des vierten Schritts, an sich reißen und die Informationen an den Peer weiterleiten. Wenn die Account-Informationen bzw. das erforderliche Passwort dann akkurat vorgehalten werden, würde der Peer die Antwort dieses neuen bindenden Endpunktes dann akzeptieren. Darum heißt es wie immer: Passworte schützen.

In GoldBug ist daher für einen Server Account - sofern er dediziert festgelegt wird - daher auch ein Passwort erforderlich, dass der Länge eines AES-256 entspricht: das ist eine Pass-Phrase von 32 Zeichen.

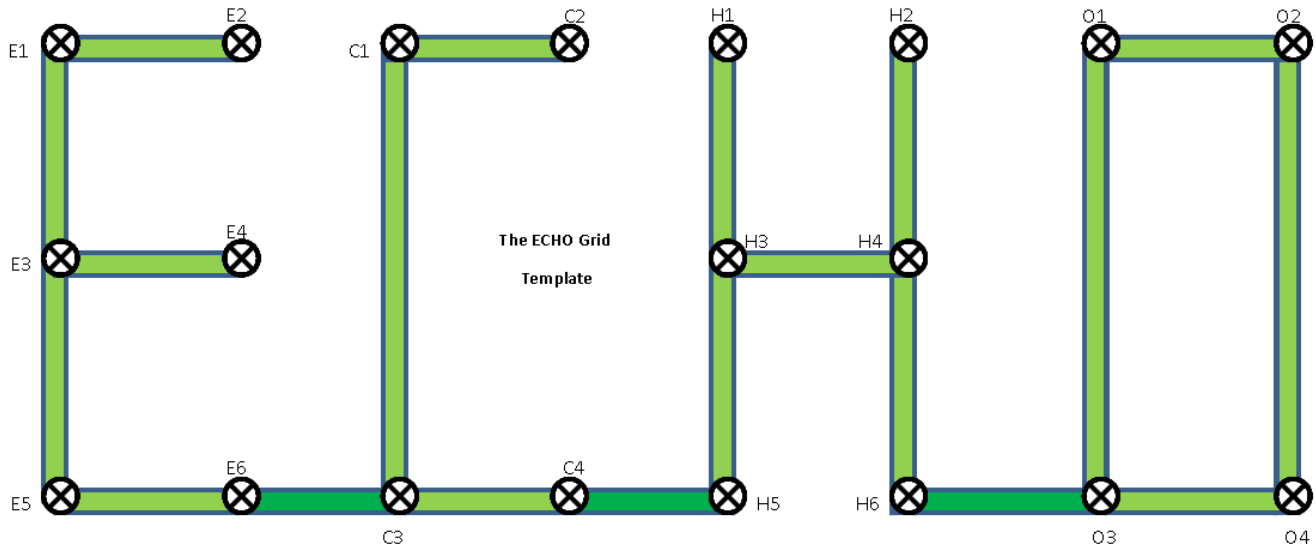
Abbildung: Account Passwords




## 3.4 Das Echo-Grid

Wenn Studierende oder Schüler über das Echo-Protokoll reden und unterrichten oder unterrichtet werden, dann lässt sich einfach ein Echo-Grid mit den Buchstaben E_C_H_O zeichnen. Die Knotenpunkte von E1 bis O4 werden nummeriert und verbinden die Buchstaben mit einer Verbindungslinie am Boden (vgl. Abbildung).

Abbildung: Das Echo Grid Template




Beispielsweise bezeichnet dann die Verbindung E1-E2 eine IP-Verbindung zu einem Nachbarn.

Wenn die einzelnen Knotenpunkte nun Schlüssel tauschen, so entstehen Verbindungen, die als neue Ebene auf der Ebene der IP-Verbindungen des P2P/F2F-Netzwerkes entstehen.

Mit der GoldBug zugrunde liegenden Architektur wurde nicht nur das kryptographische Routing in einem Kernel-Programm elaboriert etabliert, sondern - wie oben dargelegt - dem Begriff des “kryptographischen Routings” wurde mit dem Echo-Protokoll auch das Routing paradoxerweise entzogen. Es muss daher genauerweise von dem „kryptographischen Echo“ statt einem „kryptographischen Routing“ gesprochen werden. Ein davon zu differenzierendes Protokoll ist das des „Cryptographic Discovery“ auf das weiter unten noch in einem Extra-Abschnitt eingegangen wird.

Echo ist somit “beyond” Routing: Erstens, die Nachrichten-Pakete enthalten keine Routing Informationen (Adressaten) und die Knotenpunkte nutzen auch kein “Forwarding” im eigentlichen Sinne, denn sie senden einfach alles an alle Verbindungen. Und zweitens: Auch der kryptographische Schlüssel, der die Nachricht zu dekodieren versucht, ist keine Adresse (die gar dem Nachrichten-Packet beigefügt wäre), sondern nur eine polarisierende Bille: sie lässt uns Texte anders sehen und ggf. verstehen. Im Echo-Protokoll wird daher auch mehr der Begriff des “Traveling” anstelle des Begriffes “Routing” benutzt. Oder halt eben: „kryptographisches Echo”.

Auch rechtlich gesehen ist sodann hier eine andere Bewertung vorzunehmen, da ein Knotenpunkt nicht im Namen für einen Adressaten als Mittelsmann weiterleitet, sondern jeder die Nachbarn eigenständig informiert (vgl. ergänzend z.B. die Weiterleitungen in anderen Routing-Modellen wie AntsP2P mit seinem Ameisenalgorithmus, Mute, AllianceP2P, RetroShare, Onion-Routing oder I2P.

So wie sich ein guter Ruf in der Nachbarschaft verbreitet, so verbreitet sich auch die Nachricht im Echo - ansonsten lässt das Echo-Protokoll jegliches kryptographische “Zeugs“ vorbei „schwimmen” (indem es nicht dekodiert wird oder werden kann).

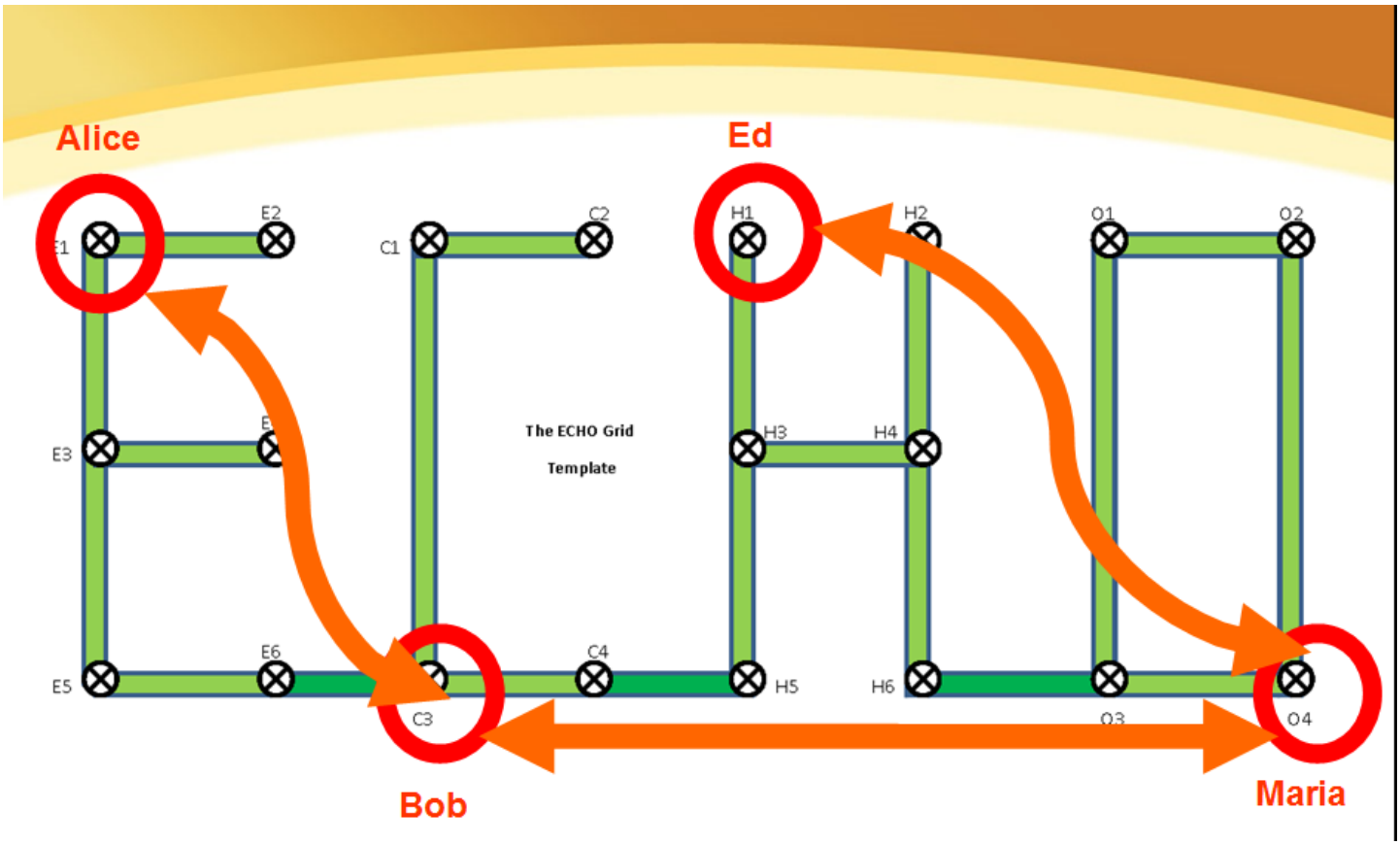
Es erinnert an das aus Star-Trek bekannte Borg-Kollektiv-Paradigma: jeder hat Zugang zu allen von den Nachbarn aufnehmbaren Nachrichten (wenn nicht halbes oder adaptives Echo benutzt wird und wenn der Nachrichtentext überhaupt verstanden (dekodiert) werden kann).

Im Echo ist der Knotenpunkt mehr ein “Souverän” oder “Gebender und Nehmender von (ungerichteten) Informationen”, in anderen Netzwerken könnte ein Knotenpunkt mehr als “Postbote”, “Dealer”, “Forwarder” oder “Vermittler” bezeichnet werden.

Das Echo-Grid als einfache Netzwerkdarstellung dient nicht nur zur Analyse von “Routing” (bzw.: “Travel”)-Wegen, zur Darstellung von Echo-Modi und Verschlüsselungs-Stationen, sondern kann letztlich insbesondere auch in der Graphentheorie Anwendung finden: Welchen Weg eine Nachricht nimmt, ist abhängig von der Struktur des Netzwerkes, aber auch von der Nutzung von Echo-Accounts, halbem bzw. vollem Echo sowie dem Adaptiven Echo, wie folgende Beispiele der Graphen zwischen Alice, Bob, Ed und Maria verdeutlichen.

### 3.4.1 Beispiele des Schlüssel-Austausches von Alice, Bob, Ed & Maria

Abbildung: Alice, Bob, Ed und Mary im Echo-Grid - Ein Beispiel für das Echo




Folgende Beispiele der Abbildung können weitergehend diskutiert werden - sie nutzen ein paar Vokabeln und Prozesse der Funktionen des GoldBug-Klienten, so dass mit dem Programm unerfahrene Leser diesen Abschnitt auch überspringen können und sich erst einmal mit den Grundfunktionen (Installation, Chat, E-Mail, Dateitransfer oder URL-Suche) vertraut machen und diese technischen Beispiele dann zu einem späteren Zeitpunkt wieder aufgreifen und nachvollziehen können):

- Alice (IP=E1) und Bob (IP=C3) haben ihren öffentlichen Schlüssel getauscht und sind über die folgenden IP-Nachbarn verbunden: E1-E3-E5-E6-C3.

- Bob (C3) und Maria (O4) sind ebenso Freunde, sie haben ihre öffentlichen Schlüssel für die Verschlüsselung ebenso getauscht: und nutzen die IP-Verbindungen der Nachbarn: C3-C4-H5-H3-H4-H6-O3-O4.

- Schließlich: Maria (O4) ist eine Freundin von Ed (H1). Sie kommunizieren entweder über den Weg: O4-O3-H6-H4-H3-H1 oder sie nutzen den Pfad von: O4-O2-O1-O3-H6-H4-H3-H1. Da im Echo-Protokoll ja jeder IP-Nachbar jede Nachricht an jeden verbundenen IP-Nachbarn sendet, wird derjenige Pfad erfolgreich sein, der die Nachricht am schnellsten übermittelt. (Die zweite eingehende Nachricht wird sodann von Congestion Control herausgefiltert).

- Direkte IP-Verbindungen von Nachbarn wie z.B. E1-E3 können durch die Erstellung eines sog. “Echo-Accounts” weiter abgesichert werden: Keine andere IP-Adresse als E1 kann dann zu dem sogenannten “Listener” des Nachbarn E3 verbinden. Über diese Methode kann ein Web-of-Trust erstellt werden - ohne von Schlüsseln zur Verschlüsselung abhängig zu sein - noch braucht der Nutzer einen Freund als Nachbarn, mit dem er seinen Chat oder E-Mail-Schlüssel tauscht.

- Sogenanntes „Turtle Hopping“ wird im Echo-Netzwerk wesentlich effizienter: Wenn Ed und Alice eine Dateiübertragung starten (über die StarBeam-Funktion mittels eines Magnet-URI-Links), dann transportiert das Echo-Protokoll die Pakete über den Weg H1-H3-H5-C4-C3-E6-E5-E3-E1. Maria ist nicht in der Route, aber sie wird die Pakete ebenso über das volle Echo erhalten, wenn sie den StarBeam-Magneten kennt. Vorteil ist, dass das Hopping nicht über die Schlüssel geht, sondern über die IP-Verbindungen (z.B. das Web-of-Trust). Grundsätzlich ist alles immer verschlüsselt, also warum nicht den kürzesten Weg im Netz nehmen?

- Ein sogenannter “Buzz” bzw. “ge-Echo-ter IRC Channel” (daher auch kurz: e’IRC) Raum kann z.B. durch den Knotenpunkte O2 erstellt oder “gehostet” werden. Da nur der Nutzer Ed den Buzz-Raum-Namen kennt, bleiben alle anderen Nachbarn und Freunde außen vor. Vorteil: Der Nutzer kann mit unbekannten Freunden in einem Raum sprechen, ohne mit diesen einen öffentlichen z.B. RSA-Schlüssel jemals getauscht zu haben. Stattdessen wird einfach ein Einmal-Magnet (“one-time-magnet”) für diesen “buzz”/”e’IRC” Raum genutzt.

- Maria ist eine gemeinsame Freundin von Ed und Bob und sie aktiviert die C/O (care of)-Funktion für E-Mails: Das erlaubt Ed, E-Mails an Bob zu senden, obwohl er offline ist, denn: Maria speichert die E-Mails zwischen, bis Bob dann online kommt.

- Weiterhin: Alice erstellte eine sogenannte virtuelle “E-Mail Institution”. Das ist nicht vergleichbar mit einem POP3 oder IMAP Server, da die E-Mails nur zwischengespeichert werden: Ed sendet dazu seinen öffentlichen E-Mail-Schlüssel an Alice - und Ed fügt den Magneten der “E-Mail Institution” von Alice bei sich in seinem Programm ein. Nun werden

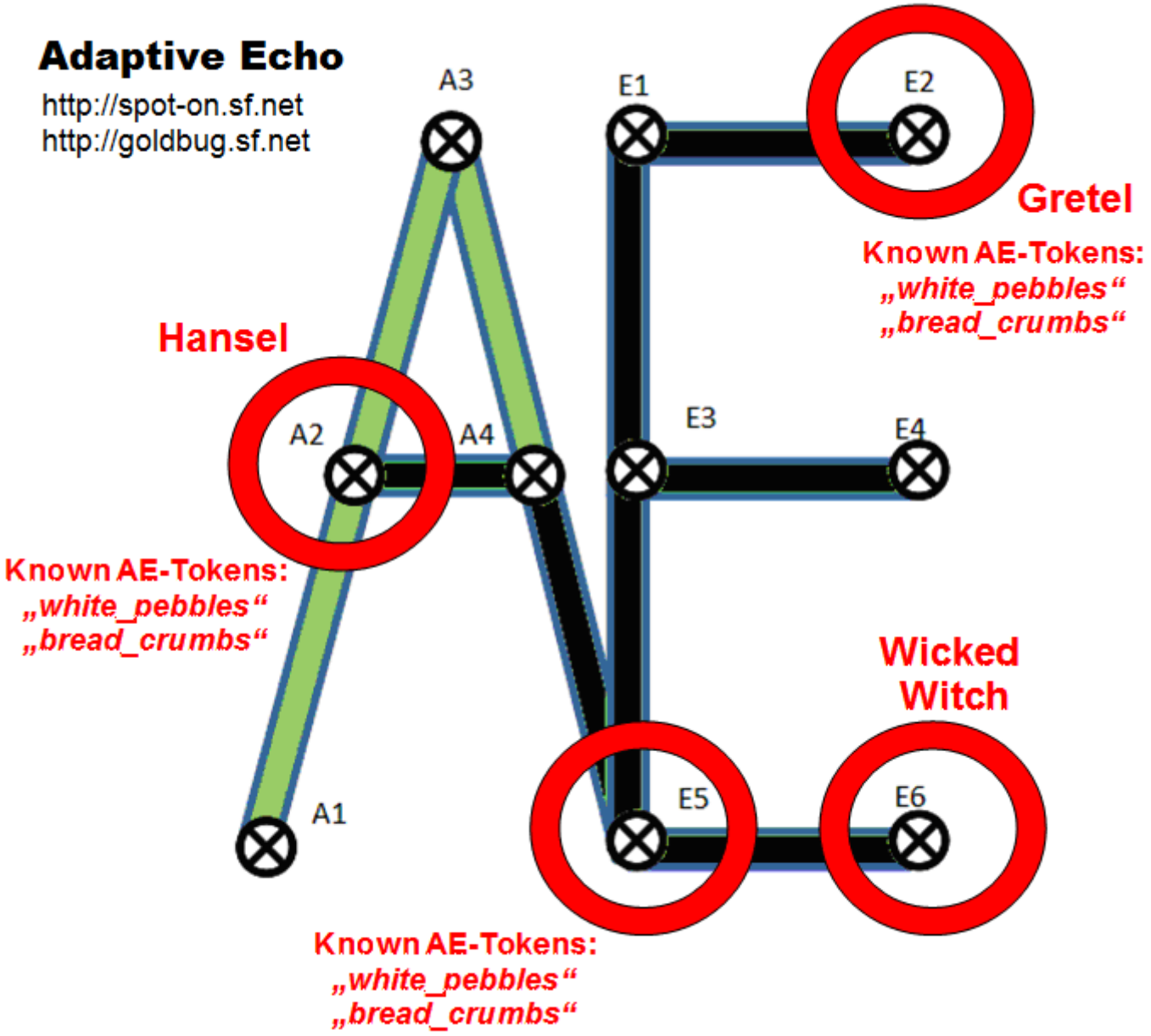
auch die E-Mails von Bob und Ed bei Alice zwischengespeichert (in der E-Mail-Institution), selbst wenn Maria offline sein sollte.

Es ist hilfreich, die Beispiele auf obiger Grafik nachzuvollziehen bzw. erst am Ende des Manuals nach weiteren Erläuterungen der Funktionen darauf zurück zu kommen.

## 3.5 Adaptives Echo (AE) und seine AE-Tokens

Für die Erklärung des “Adaptiven Echos” kann ein weiteres Echo-Grid mit den verbundenen Buchstaben A und E gezeichnet werden (siehe folgende Abbildung).

Abbildung: Adaptives Echo (AE): Das „Hänsel und Gretel“-Beispiel des Adaptiven Echos]]




Wenn ein Nutzer, sein Chat-Freund und ein eingerichteter dritter Kontenpunkt als Chat-Server denselben AE-Token (“Adaptive-Echo Token”) in das Programm einfügen, dann wird der Chat-Server die Nachricht des Nutzers nur zu seinem Freund senden - und nicht zu allen anderen

verbundenen Nachbarn oder Nutzern wie es normalerweise bei dem Vollen Echo Modus der Fall wäre.

Der AE-Token besteht wie eine Passphrase aus mindestens 96 Zeichen. Beim Adaptiven Echo wird die Information vom aussendenden Knotenpunkt der verschlüsselten Kapsel beigefügt - und alle weiteren Knotenpunkte lernen, dass Sie die Nachricht nur an Knotenpunkte bzw. Verbindungspartner weitersenden, die diesen AE-Token ebenso kennen.

Mit einem AE-Token wird kein anderer Knotenpunkt, der die Passphrase nicht kennt, die Nachricht des Nutzers erhalten oder einsehen können. Damit können potentielle “Recorder” ausgenommen werden, also mögliche Nachbarn, die möglicher- und anzunehmender Weise den gesamten Nachrichtenverkehr aufzeichnen und sodann versuchen wollen, die mehrfache Verschlüsselung aufzubrechen, um an den Nachrichten-Kern der Kapsel zu kommen.

Um den Graphen, die Reiseroute, für das Adaptive Echo bestimmen zu können, müssen sich mehrere Knotenpunkte untereinander absprechen und die Passphrase auf dem Wegespfad lückenlos vermerken. Im Falle des Adaptiven Echos kann von einem Routing gesprochen werden.

### 3.5.1 Hänsel und Gretel – Ein Beispiel für den Adaptiven Echo Modus

Zur Erläuterung des Adaptiven Echos dient als klassisches Beispiel das Märchen von Hänsel und Gretel.

In das oben erläuterte AE-Grid werden die Personen Hänsel, Gretel und die böse Hexe als Knotenpunkte eingezeichnet. Nun überlegen Hänsel und Gretel, wie sie miteinander kommunizieren können, ohne dass die böse Hexe dieses mitbekommt. Dem Märchen nach sind sie im Wald bei der Hexe und wollen aus diesem Wald wieder herausfinden und markieren den Weg mit “Brotkrumen” (“bread crumbs”) und “Weissen Kieselsteinen” (“white pebbels”).

Diese Märcheninhalte können nun auch in obigem Grid-Muster das Adaptiven Echo verdeutlichen und aufzeigen, an welchen Stellen des Grids bzw. des Kommunikationsgraphens ein kryptographischer Token namens “Weisse Kieselsteine” eingesetzt werden kann:

Wenn Knotenpunkt A2, E5 und E2 denselben AE-Token einsetzen, dann wird Knotenpunkt E6 keine Nachricht erhalten, die der Knotenpunkt A2 (Hänsel) und der Knotenpunkt E2 (Gretel) austauschen werden. Denn, der Knotenpunkt E5 lernt über den bekannten Token “Weisse Kieselsteine” (“white_pebbles”), die Nachrichten nicht an den Kontenpunkt E6, die “Böse Hexe” (“Wicked Witch”), zu senden. Ein lernendes, sich anpassendes (“adaptives”) Netzwerk.

Ein **“Adaptives Echo”-Netzwerk** enthüllt dabei keine Ziel-Informationen (vergleiche dagegen auch nochmals oben: “Ants Routing”). Denn - zur Erinnerung: Der Modus des “Halben Echos” sendet nur einen Hop zum verbundenen Nachbarn und das “Volle Echo” sendet die verschlüsselte Nachricht zu allen verbundenen Knotenpunkte über eine nicht weiter spezifizierte Hop-Anzahl. Während “Echo Accounts” andere Nutzer quasi als Firewall oder

Berechtigungskonzept beim Verbinden fördern oder behindern, halten hingegen “AE-Tokens” Graphen- oder Pfad-Exklusivität vor - und zwar für Nachrichten, die über Verbindungsknoten gesandt werden, die den AE-Token kennen.

Chat-Server Administratoren können ihre Token mit anderen Server-Administratoren tauschen, wenn Sie sich untereinander vertrauen (sogenanntes “Ultra-Peering for Trust”) und ein Web-of-Trust definieren. In Netzwerk-Laboren oder zuhause mit drei, vier Rechnern kann man das Adaptive Echo einfach austesten und seine Ergebnisse dokumentieren:

Für einen Test des Adaptiven Echos nutzt man einfach ein Netzwerk mit drei oder mehreren Computern (oder “SPOTON_HOME” als (endungslose) Datei im Binärverzeichnis, um mehrere Programminstanzen auf einer einzigen Maschine zu launchen und zu verbinden) und setzt sodann diesen beispielhaften Ablauf um:

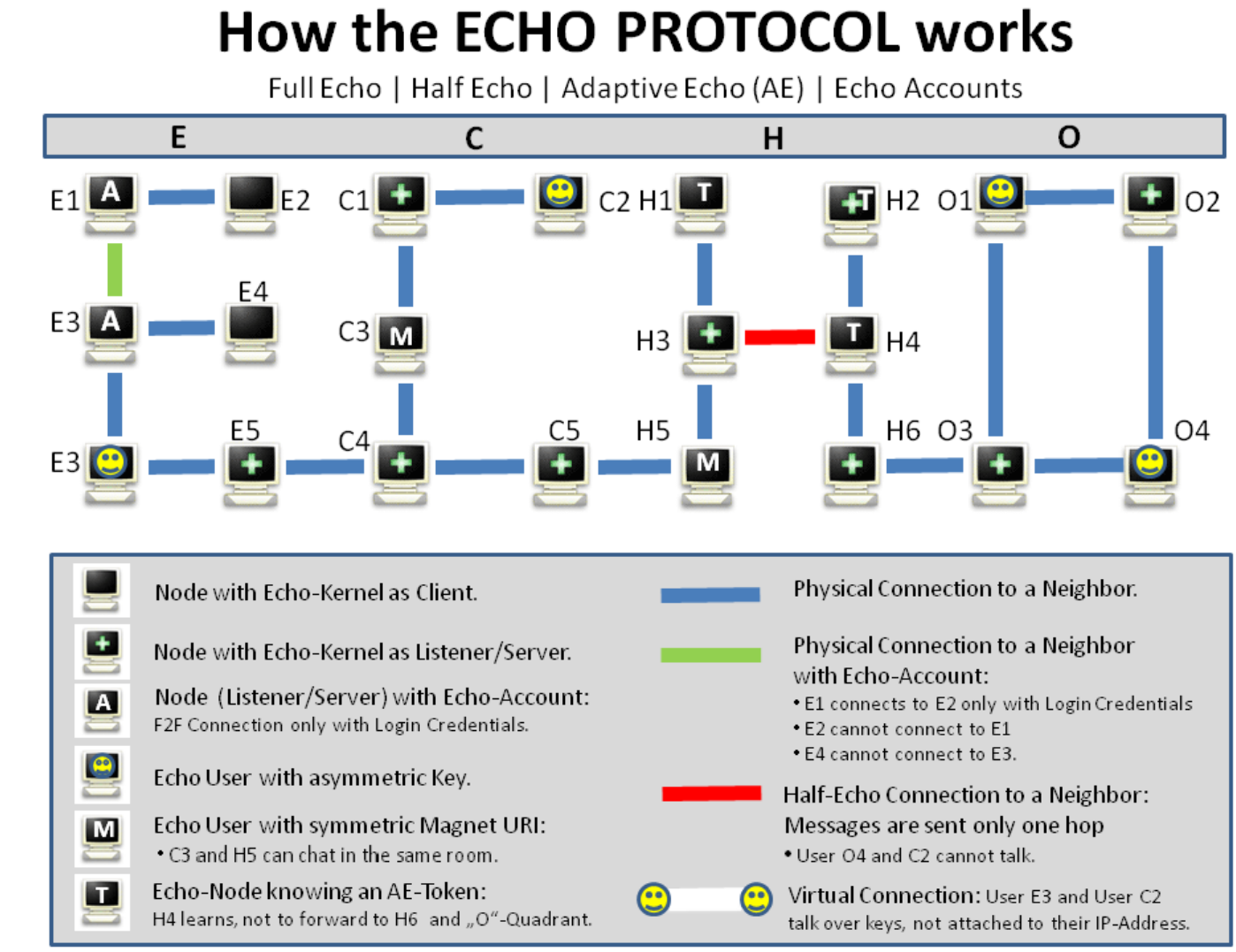
- Erstelle einen Knotenpunkt als Chat Server.
- Erstelle zwei Knotenpunkte als Klient.
- Verbinde die beiden Klienten zum Chat Server.
- Tausche Schlüssel zwischen den Klienten.
- Teste die normale Kommunikationsfähigkeit beider Klienten.
- Setze einen AE-Token auf dem Server.
- Teste die normale Kommunikationsfähigkeit beider Klienten.
- Setze denselben AE-Token nun auch in einem Klienten.
- Notiere das Ergebnis: Der Server-Knotenpunkt sendet die Nachricht nicht mehr an andere Knotenpunkte aus, die den AE-Token nicht haben bzw. kennen.

Dieses Beispiel sollte einfach replizierbar sein.

## 3.6 Einige Beispiele, wie das Echo-Protokoll funktioniert

Nimmt man nun die verschiedenen Methoden und Optionen zusammen, wird die Abbildung “Wie das Echo-Protkoll funktioniert” einen komplexeren Überblick bieten.

Abbildung: Wie das Echo PROTOCOL funktioniert




- In der Graphik abgebildet sind die unterschiedlichen Nutzungsbeispiele von “Full Echo”, “Half Echo”, Adaptive Echo” sowie “Echo Accounts”.
- Unterschieden wird zwischen physischen IP-Verbindungen und virtuellen Verbindungen zu Schlüsseln. Schlüssel sind daher nicht zwingend einer IP-Verbindung zugeordnet.
- Nutzer können darin asymmetrische öffentliche Schlüssel, aber auch Magnet-URIs mit symmetrischen Verschlüsselungsdetails sowie Tokens und Account-Credentials tauschen.
- Verbindungsknoten können Verbindungen erlauben und verbieten - ebenso wie Nachrichten dediziert adressiert (oder auslassend adressiert) senden.
- Dementsprechend entstehen unterschiedliche Kommunikations-Szenarien.

**Beispiele:**

- a. Nutzer H4 hat einen AE-Token. Er sendet keine Nachrichten (über den Verbindungsknoten H6) in den O-Quadranten, wenn HG den Token nicht kennt.
- b. Wenn H3 eine Nachricht an H4 sendet, dann sendet H4 diese Nachricht ebenso nicht weiter, da es sich um eine Verbindung des „Halben Echos“ handelt.

- c. Der Nutzer E1 lässt den Nutzer E2 nicht verbinden, da er das Login für den Echo-Account nicht kennt.
- d. Nutzer O1 und O4 chatten miteinander und kennen nur ihren öffentlichen Schlüssel für die Verschlüsselung.
- e. Nutzer H3 und C5 chatten über einen URI-Magneten im gleichen Gruppen-Chat-Raum (auch Buzz oder e'IRC genannt).

Es zeigt sich, dass das Echo-Protokoll ein durchaus komplexes Netzwerk abbilden kann, obschon es eine simple Struktur hat. Mit den einzelnen Optionen und Begriffen können Nutzer in der Praxis sich viel Netzwerk- und Krypto-Theorie erschließen und diese praktisch ausprobieren. Ein ideales Instrument für die Lehre, die Heranführung an kryptographische Prozesse, Graphen-Theorie und eine praktische Nutzererfahrung in der Gruppe. Zum Beispiel für ein Dreier-Team, wie in der GoldBug-Geschichte von Edgar Alan Poe.

# 4 Cryptographisches Discovery

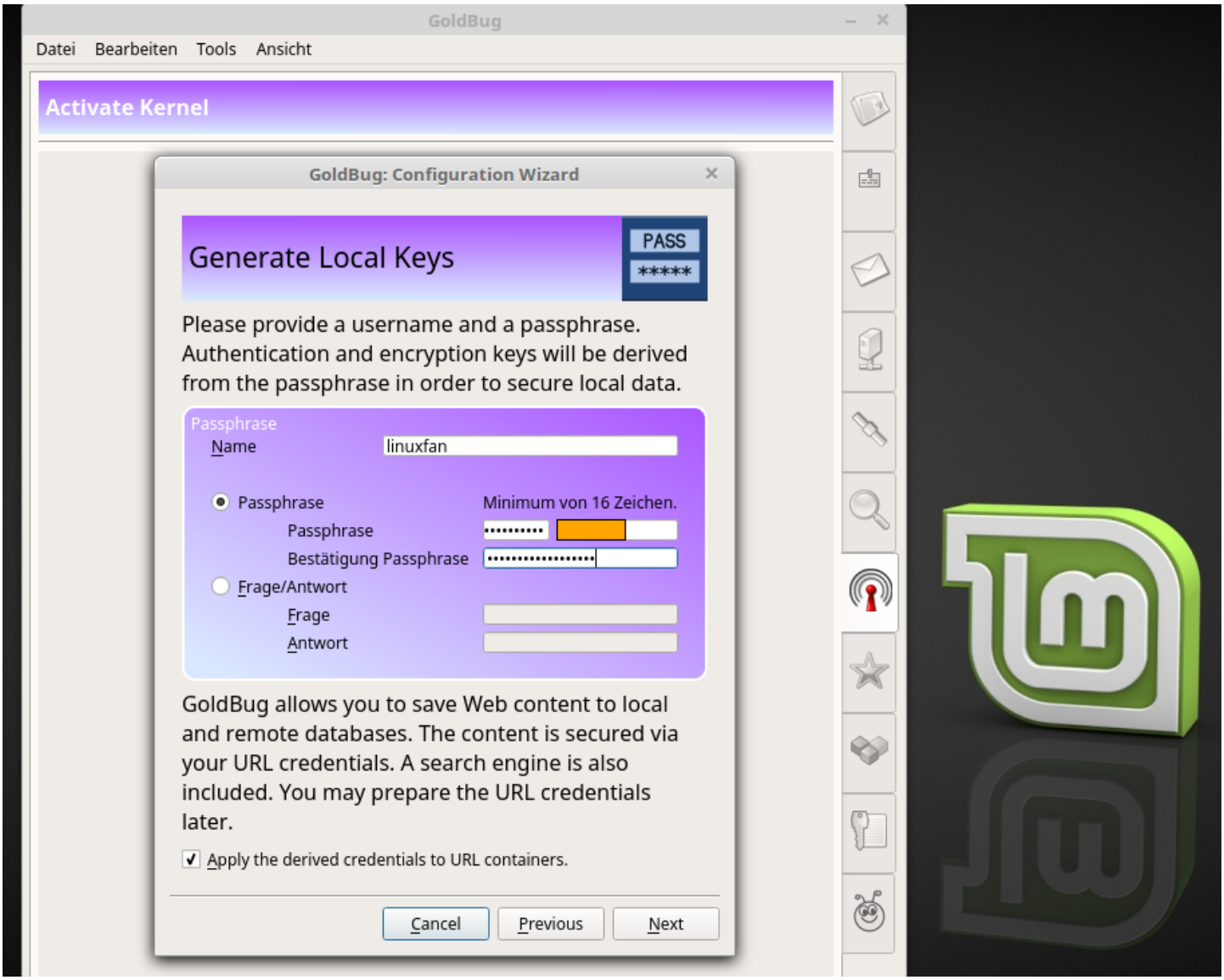
Cryptographic Discovery beschreibt die Methode eines das Echo ergänzenden Protokolls, Nodes in einem Echo-Netzwerk zu finden. Das Echo-Protokoll wird damit um eine weitere sinnvolle Methode ergänzt, wenn sie nicht sogar bedeutender ist, als das Echo selbst. Cryptographic Discovery ist in den bestehenden Clienten wie GoldBug, Spot-on oder auch dem Chat-Server für das Betriebssystem Android, SmokeStack, auf der Code Basis implementiert. Der Source Code und seine Dokumentation legt die Methode entsprechend dar. Cryptographic Discovery kann z.B. einen Distributed Hash Table (DHT) ersetzen, um einen Freund im Netzwerk zu finden. Eine weitergehende Publikation speziell zu diesem Thema als Anlage zum Quellcode der Entwicklung ist in Vorbereitung.

# 5 Einen ersten Setup einrichten – z.B. mit dem Wizard

Der erste initiale Setup der Software ist mit wenigen Schritten ganz einfach,

- der Nutzer entpackt das Programm aus dem Zip und startet (unter Windows) die GoldBug.exe aus dem Pfad, in den das Programm entpackt wurde, z.B. C:/GoldBug/GoldBug.exe oder C:/Programme/GoldBug/GoldBug.exe.
- Es erscheint die Benutzeroberfläche und ein Wizard, mit dem Einstellungen Schritt für Schritt umgesetzt werden können. Alternativ kann man den Wizard auch schließen und die Einstellungen manuell im Tab für die Einstellungen bzw. für die Kernel-Aktivierung vornehmen. Es empfiehlt sich, den Wizard zu nutzen.
- Im Wizard werden sodann mit dem Nutzernamen und einer zweifach einzugebenden Passphrase die notwendigen kryptographischen Schlüssel generiert.

Abbildung: Initialer Wizard



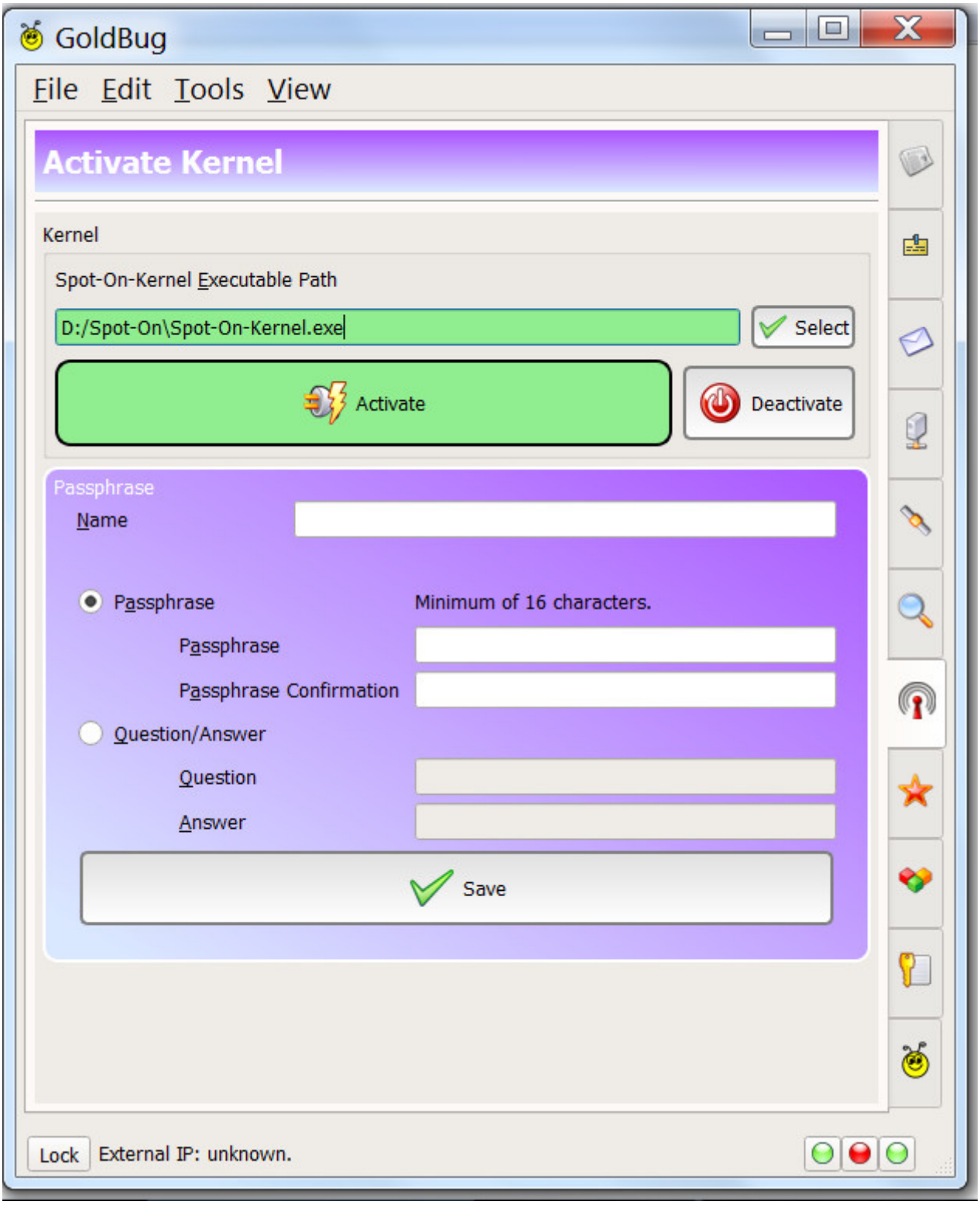

- Nach Abschluss des Wizards ist noch der Kernel zu aktivieren. Der GoldBug Messenger hat also eine Benutzeroberfläche (auch Interface oder Graphical User Interface (GUI) genannt) und einen Kernel. Beide sind als Binärdatei gegeben (also unter Windows als GoldBug.exe und Spot-On-Kernel.exe bezeichnet). Spot-On ist das Original-Projekt für die Echo-Applikation und GoldBug stellt lediglich eine vereinfachte Benutzeroberfläche zur Verfügung.
- Wenn der Kernel läuft, ist eine Verbindung zu einem Nachbarn oder Server mit der entsprechenden IP im Tab Nachbarn vorzunehmen.
- Sodann tauscht der Nutzer mit einem Freund den Schlüssel und die verschlüsselte Kommunikation per Chat oder E-Mail kann beginnen...

Ansonsten muss in diesem Tabulator für Einstellungen bzw. für die Kernel-Aktivierung nach jedem Start von GoldBug.exe der Kernel über den Knopf “Aktiviere” aktiviert werden, der dann die Verbindungen zu Freunden oder zu einem Chat-Server koordiniert. Die Kernel-Datei Spot-on-Kernel.exe wird also aus dem Programm von GoldBug an oder abgeschaltet.

## 5.1 Zwei Login-Methoden

Wenn der Nutzer GoldBug das erste Mal startet, ist in dem entsprechenden Kasten ein Nicknamen einzugeben und eine Passphrase für den Login in die Applikation zu definieren (vgl. Abbildung, blauer Widget-Kasten).

Abbildung: Setze Passwort




Dazu gibt es zwei Methoden: die Passphrase-Methode oder die Frage-Antwort (Question/Answer) Methode.

Das Passwort muss mindestens 16 Zeichen lang sein. Wem das zu lang ist, der kann ein kürzeres Passwort auch dreimal wiederholen wie z.B. “passwort_passwort_passwort”, jedoch ist das Passwort dann nicht so sicher wie eines mit zufälliger Zeichenkette.

Die zwei Methoden lassen sich wie folgt unterscheiden:

**Passphrase-Methode:** Bei der Erstellung des Passwortes wird dieses nicht lokal gespeichert, sondern nur der Hash der Eingabe. Der Hash wird um eine ergänzende Zeichenfolge ergänzt, das sogenannte kryptologische Salz. Dieses ergänzt den Hash und macht ihn daher sicherer. Der „Salted Hash“ definiert sich also wie folgt: Hash (Passphrase + Salt). Damit das Passwort auch beim Nutzer eingeübt wird und Tippfehler ausgeschlossen werden, muss es ein zweites Mail eingegeben werden.

**Frage/Antwort-Methode:** Bei dieser Methode wird nicht ein Passwort zweimal eingegeben, sondern es wird eine Zeichenkette als Frage definiert und eine Zeichenkette als Antwort. Beide Strings werden nicht ein zweites Mal überprüft. Technisch wird diese Login-Methode über eine HMAC umgesetzt: Hash (Question, Answer), weißt darauf hin, dass ein “HMAC” (Keyed-Hash Message Authentication Code) genutzt wird. Und: Weder die Frage, noch die Antwort, werden auf der Maschine des Nutzers gespeichert und kein kryptographisches Salz wird durch die Maschine per Zufall generiert. Anstelle der Frage kann der Nutzer natürlich auch zwei Passworte ohne ein Fragezeichen eingeben. Es ist zu beachten, dass hier die Frage und die Antwort bei späteren Logins exakt so eingegeben werden müssen, wie sie definiert wurden und hier bei der erstmaligen Definition kein zweiter Eingabecheck (“Confirmation”) hinsichtlich Tippfehler wie bei der Passwort-Methode erfolgt.

Abbildung: Login in die Applikation mit einem Passwort




Da der Hash, der aus der Login-Passphrase generiert wird, die verschlüsselten Container freischaltet, in denen auch der private Schlüssel für die Verschlüsselung gespeichert wird, ist es besonders wichtig, den Login-Prozess und das Login-Passwort zu schützen. Daher wurden oben genannte zwei Methoden berücksichtigt, um es Angreifern schwerer zu machen: Diese wissen somit nicht, a) welche Methode ein Nutzer gewählt hat und b) die Frage-Antwort-Methode ist insofern wie oben schon beschrieben zusätzlich sicherer, weil hier weder die Frage, noch die Antwort irgendwo gespeichert werden und ein HMAC ggf. aufwendiger ist als ein Passwort als „nur“ gesalzener Hash. Nur der Nutzer kennt Frage und zusätzlich die Antwort und nur der Match aus beidem kann den Container öffnen.

Um auch Tastatur-Loggern die Eingaben nicht Preis zu geben, besteht bei der Login-Eingabe die Möglichkeit, eine virtuelle Tastatur zu nutzen (siehe Abbildung). Diese startet der Nutzer mit einem Doppelklick auf die Eingabezeile für das Passwort. Allenfalls können hier nur Mausklicks aufgezeichnet werden, aber keine Tastatureingaben.

Abbildung: Virtuelles Keyboard




Grundsätzlich ist es wichtig, dass der private Schlüssel in einem ausreichend gesicherten Container verschlüsselt aufbewahrt wird. Es liegt die Vermutung nahe, dass insbesondere der Zugriff von Anbieterfirmen auf mobile Betriebssysteme es andernfalls leicht machen würde, den privaten Schlüssel abzugreifen.

Dieses ist insbesondere auch bei Web-Mail-Angeboten kritisch zu hinterfragen, die Verschlüsselung im Browser oder mit Schlüsseln anbieten, die beim Mail-Anbieter online hinterlegt sind. Verschlüsselung sollte immer auf der Maschine des Nutzers stattfinden und dazu ist ein quell-offener Klient und keine Online-Web Anwendung im Browser zu nutzen, in der der Nutzer ggf. - auch selbst generierte - Schlüssel online hinterlegen soll.

Die Gefahr des Abgreifens des ggf. nicht ausreichend verschlüsselten privaten Schlüssels ist viel zu groß. Auch Programm-Audits sollten auf das Auf-greifen der Passworte für den verschlüsselten Container, in denen sich der private Schlüssel befindet, sowie auf das Ab-Greifen durch remote-initiiertes Hochladen des privaten Schlüssels besonderes Augenmerk legen.

Selbst die wenigen quell-offenen Messenger mit Verschlüsselung, die man für den Desktop wie auch für mobile Geräte an einer Hand abzählen kann, die ein Security-Audit durchlaufen haben, sind kaum ausreichend hinsichtlich der Sicherheit der verschlüsselten Speicherung vom - und gesicherter Zugangsprozesse zum - privaten Schlüssel analysiert.

## 5.2 Generierung von 10 Schlüsseln für die Verschlüsselung

Wenn der Nutzer das erste Mal den GoldBug Messenger startet, fragt der Wizard, ob der Nutzer die Schlüssel für die Verschlüsselung generieren möchte. Für die Schlüssel-Erstellung sollte man einen Schlüssel von mindestens 3072 Bit (Voreinstellung) oder größer wählen. Der Nutzer kann auch weitere Optionen wie Algorithmus, Hashtype, Cipher, Salz-Länge oder Iteration Count selbst wählen, wenn er den Schlüssel z.B. neu (re-)generiert. Der erste Setup hält eine Voreinstellung auf Basis von RSA bereit: Wer also NTUR oder McEliece als Algorithmus austesten will, sollte sodann nach dem ersten Setup nochmals neue Schlüssel mit einem der dann wählbaren Algorithmen generieren.

Die generierten Schlüssel sind im Unterpfad „/.spot-on“ gespeichert. Wenn der Nutzer einen neuen Login mit neuen Schlüsseln aufsetzen will, und alle Nutzerdaten gelöscht werden sollen, dann wird am besten dieser Pfad einfach gelöscht und die GoldBug.exe neu gestartet. Für Linux und die weiteren Betriebssysteme gelten deren adäquaten Pfadangaben. Gleiches kann im Hauptmenü mit “!!!*Total_Database Erase*!!!” erreicht werden.

Für folgende Funktionen werden asymmetrische Schlüssel generiert (jeweils ein Schlüssel für die Verschlüsselung und ebenso ein Schlüssel für die (optionale) Signatur):

- Chat: Hierbei geht es um den 1:1 Chat
- E-Mail: hierbei geht es um E-Mail zu anderen Nutzern von GoldBug oder Spot-on
- POPTASTIC: Hierbei geht es um den Chat über E-Mail-Server
- URLs: Hierbei geht es um die Suche von URLs in der URL-Datenbank (Websuche)
- Rosetta: Mit dem Rosetta Verschlüsselungs-Pad können Texte mit asymmetrischen Schlüsseln von Plaintext zu Ciphertext und umgekehrt hin und her konvertiert werden, bevor sie versandt werden. Dieses empfiehlt sich, wenn andere unsichere Messenger oder E-Mail genutzt werden oder der Ciphertext irgendwo im Web gepostet werden soll - oder der Plaintext, bevor er in GoldBug versandt wird, nochmals eine zusätzliche Verschlüsselungsebene erhalten soll.

Dass jede Funktion ein eigenes Schlüsselpaar nutzt, ist wiederum ein Sicherheitsmerkmal. Wenn der Chat-Schlüssel kompromittiert wäre, ist somit die E-Mail-Verschlüsselung davon nicht betroffen. Ferner kann der Nutzer auch Freunden nur seinen Chat-Schlüssel übergeben und nicht den E-Mail-Schlüssel. Somit kann der Nutzer entscheiden, wem er es erlaubt, mit ihm zu chatten oder nur zu e-mailen oder ggf. auch URLs auszutauschen für die Funktion der p2p Websuche in der integrierten URL-Datenbank.

Beschrieben ist bislang die minimale Sicht auf die Benutzeroberfläche: Über das Hauptmenü kann man zwischen „voller Ansicht“ oder „minimaler Ansicht“ wählen. Wer sich mit Computern nicht so gut auskennt, sollte die minimale Ansicht wählen, da es die ggf. nicht benötigte Optionsvielfalt ausblendet. Keep it simple! Qt-Entwickler, und solche, die ein Übungs-Projekt für ein eigenes Qt-Entwicklungsprojekt suchen, können die Benutzeroberfläche auch ggf. noch minimaler ausgestalten (und das GoldBug Projekt gerne „forken“).

Beim ersten Setup ist die Option der maximalen Ansicht nicht verfügbar, sie wird erst bei den weiteren Logins eingeblendet und einstellbar. Die Möglichkeit, noch mehr Details in der Benutzeroberfläche anzusehen, soll daher nur deshalb an dieser Stelle kurz angesprochen werden, da sich viele Details auch auf den unten zuletzt genannten Punkt der kryptographischen Werte beziehen, die auch in dem Tabulator der Kernel-Aktivierung und Schlüssel-Generierung zu finden sind (eben in der Einstellung der maximalen Ansicht). Die Werte können bei einer erneuten Schlüssel-Generation (ohne Wizard) aus der erweiterten Benutzeransicht individuell eingestellt werden. Wer jedoch den Klienten zum ersten Mal benutzt, hat die typischen Einstellungswerte automatisch parat, d.h. z.B. der Schlüssel hat eine (vordefinierte) Größe von 3072 Bit im RSA-Algorithmus.

Bei nicht-minimaler Ansicht zeigen sich im Tabulator “Activate Kernel” somit z.B. noch folgende Elemente in der Benutzeroberfläche:

- Pfad zum Kernel: Hier kann der Nutzer den Pfad zum Kernel eingeben. Ist der Kernel mit der “spot-on-kernel.exe” im Pfad richtig angegeben, dann ist der Pfad grün hinterlegt. Andernfalls ist zu schauen, wo die ausführbare Datei des Kernels liegt oder man kopiere sie ebenso zur ausführbaren Datei der GUI (GoldBug.exe) bzw. passt den Pfad entsprechend an.

- PID: Die PID Nummer kennzeichnet die Prozess-ID, mit der in Windows die ausführbare Datei gekennzeichnet ist. Der Nutzer findet auch im Windows Task-Manager die Prozess-IDs.

- “Key-Regeneration”-Funktion: Mit der “Regeneration”-Funktion kann der Nutzer einzelne Schlüssel auch neu generieren - mit neuen Werten und Optionen. Dazu ist die Check-Box zu aktivieren, sind die Werte zu setzen und er jeweilige Schlüssel zu re-generieren. Dann muss der Nutzer seinen neuen Schlüssel jedoch wieder seinen Freunden zur Verfügung stellen, denn der Schlüssel ist ja die Kommunikations-ID.

Eine weitere Vielzahl an Optionen findet sich auch unter dem Hauptmenü/Optionen in einem Pop-Up-Fenster, die später noch erläutert werden. Dieses ist das eigentliche Options-Fenster, das für einen ersten Start jedoch zunächst vernachlässigt werden kann.

Abbildung: Optionen z.B. für digitale Signaturen

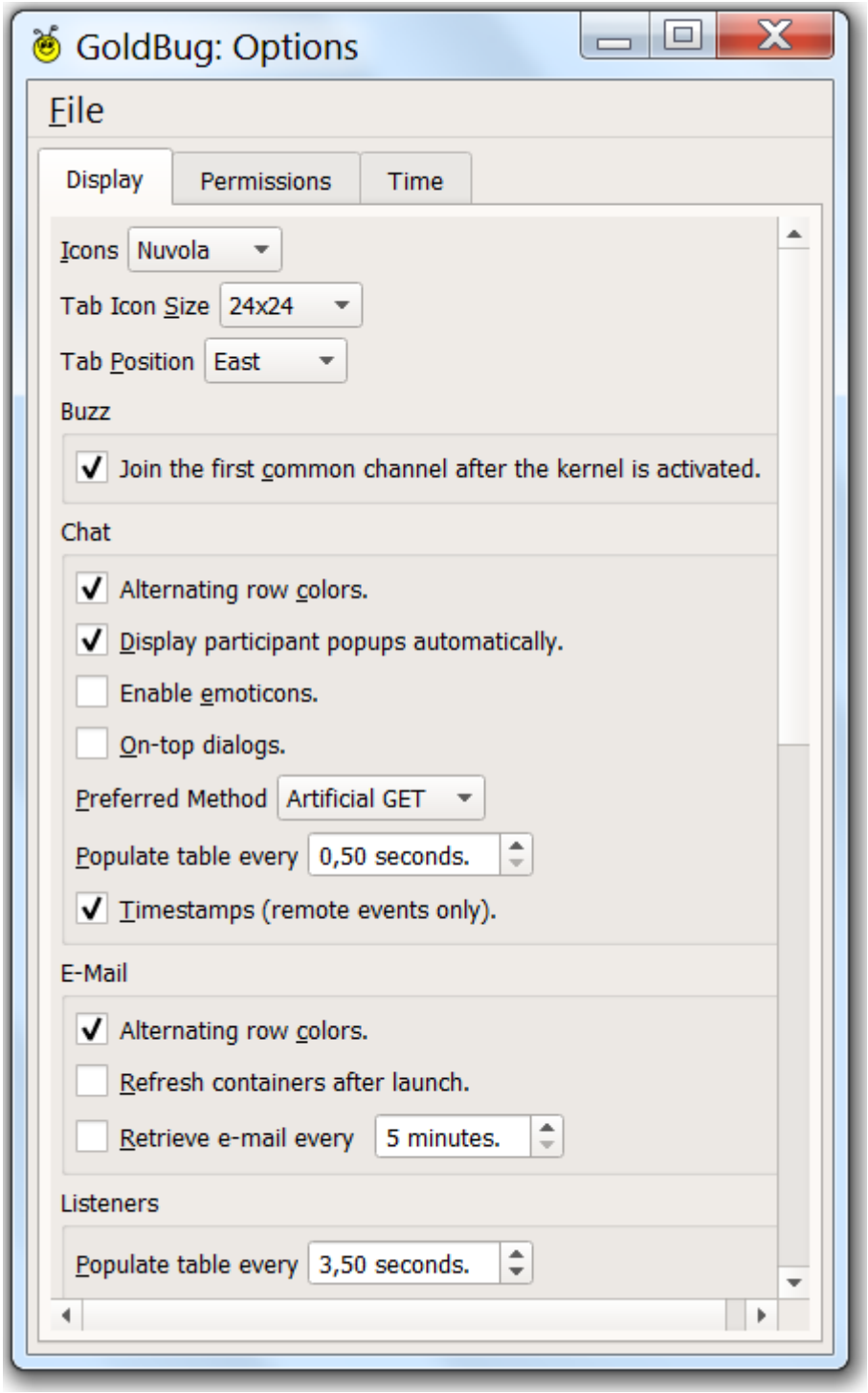
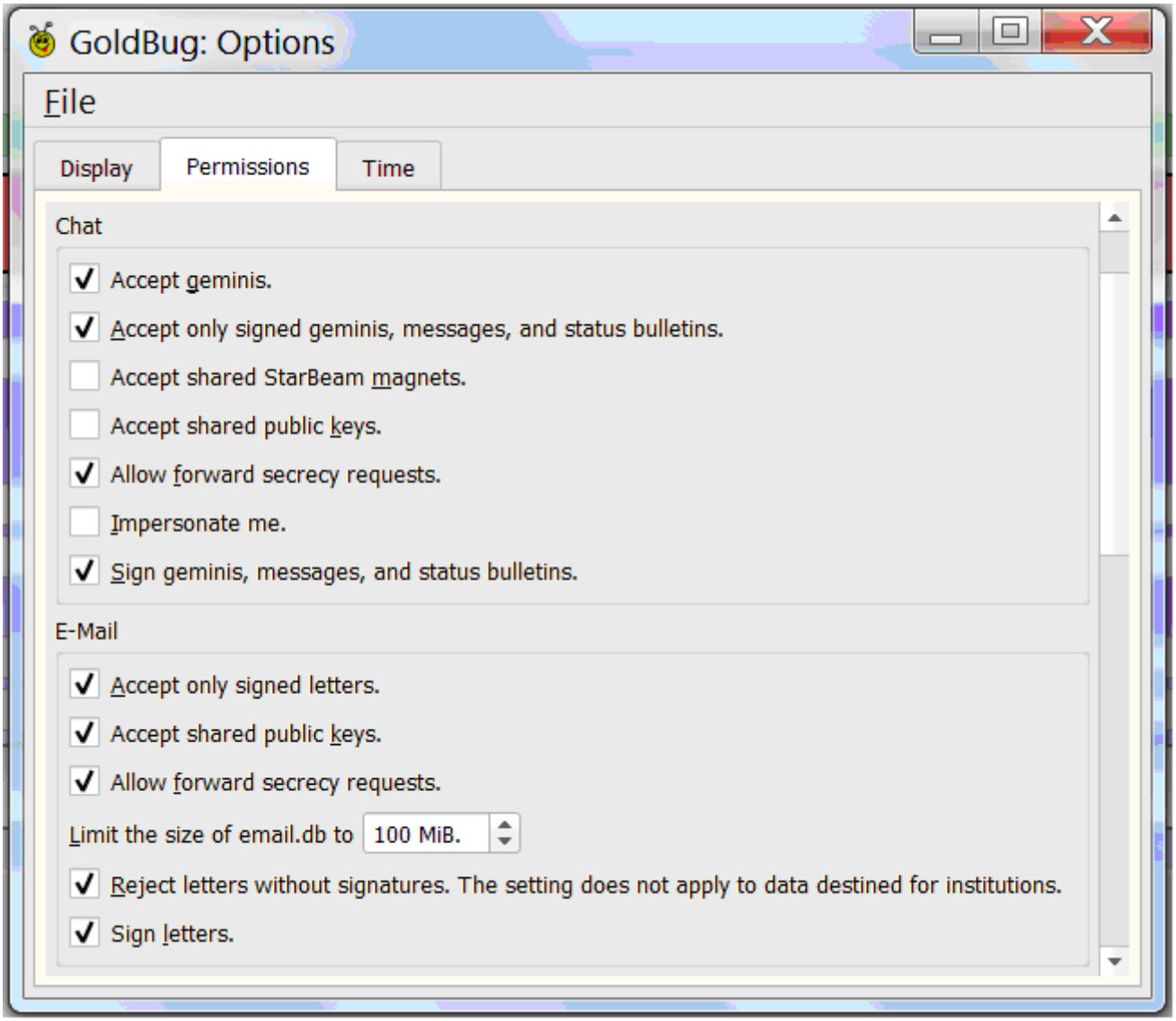



Abbildung: Optionen z.B. für die Ansicht im Klienten




Wichtiger ist, nach der ersten Schlüssel-Generierung über den Wizard den Kernel zu starten.

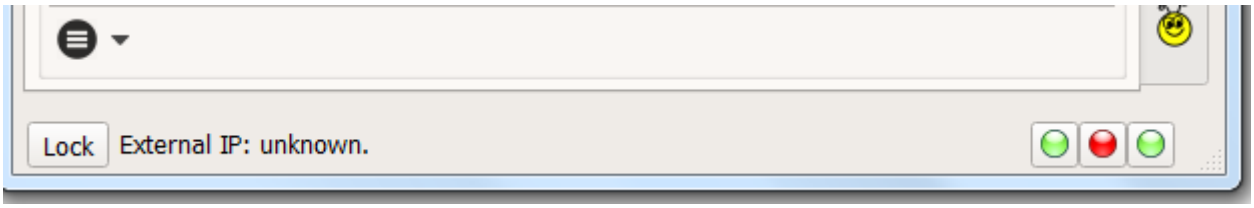
## 5.3 Aktivierung des Kernels

Wenn der Nutzer das erste Mal den GoldBug Messenger startet, fragt ein Pop-Up Fenster am Ende des Wizards, ob der Kernel aktiviert werden soll. Ansonsten ist bei allen weiteren Starts nach dem Login im Einstellungs-Tab der rote “Aktiviere Kernel”-Knopf zu drücken, sodann kann es losgehen: Wenn der Knopf grün ist, läuft der Kernel.

Wenn der Nutzer die Programmoberfläche schließt, wird ebenso in einem Pop-up Fenster gefragt, ob der Kernel noch weiterlaufen soll. Es empfiehlt sich also, erst den Kernel zu deaktivieren und dann die GUI von GoldBug zu schließen, wenn man das Programm komplett schließen will.

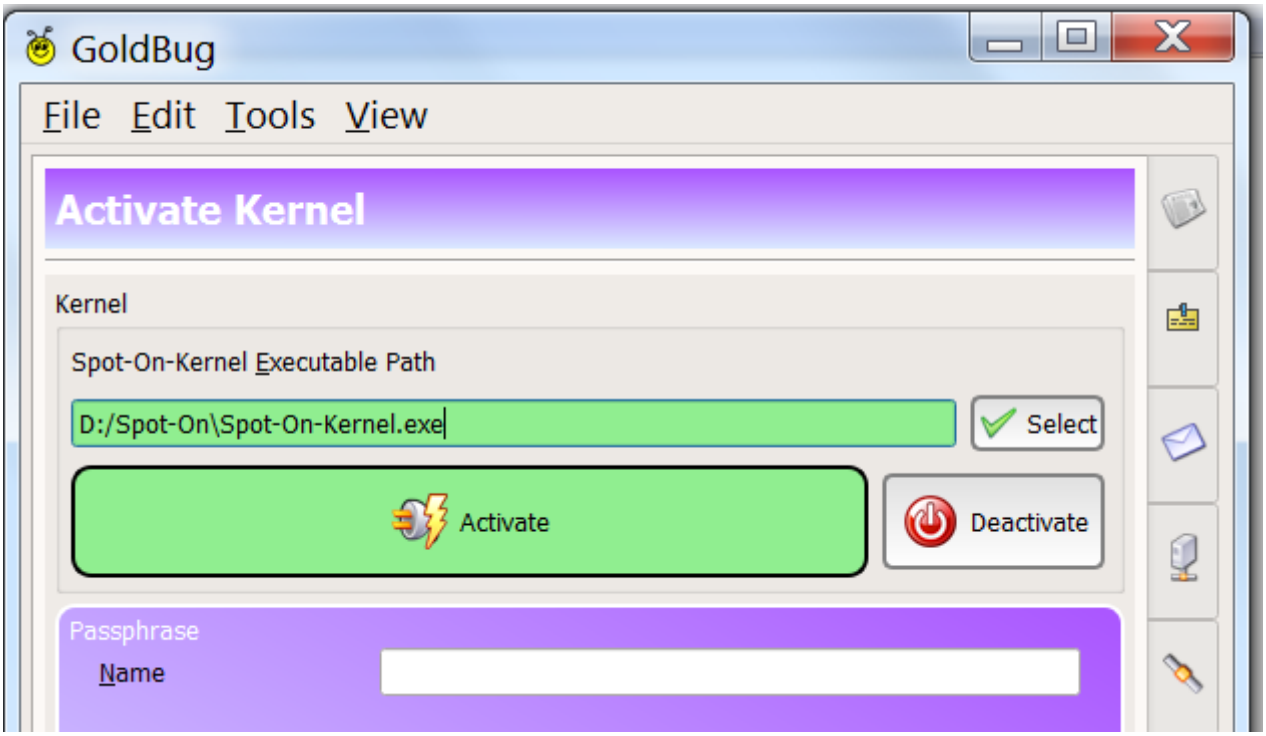
Ansonsten betreibt der Nutzer den Kernel ohne GUI, was ja manchmal auf einem Webserver gewünscht ist, damit niemand sich in die offene Benutzeroberfläche einschalten kann. (Neben dem Spot-on-Kernel gibt es für diesen Dämon-Webserver-Zweck aber auch noch den Spot-on-Lite-Kernel, der beim Repositorium des Source Codes als eigenständiges Repositorium zu finden ist.)

Abbildung: Lock der Benutzeroberfläche in der Statuszeile




Wenn der Nutzer die GUI bestehen lassen will, aber niemand während der Abwesenheit Eingaben oder Veränderungen vornehmen können soll, besteht auch die Möglichkeit, links in der unteren Statuszeile den Knopf “Lock” zu klicken, die Benutzeroberfläche schließt dann und kehrt zum Login-Tab für die Eingabe des Passwortes zurück, so dass die laufenden Prozesse und Eingaben der anderen Tabulatoren nicht sichtbar sind. Um die Oberfläche wieder zu ent-lock-en, ist der Lock-Knopf in der Statuszeile erneut zu drücken und der Nutzer gibt die Passphrase(n) dann in einem Pop-Up-Fenster wieder ein.

Abbildung: Aktivierung des Kernels




Der Nutzer kann den Kernel auch aktivieren/deaktivieren, indem in der Status-Zeile unten links die erste LED gedrückt wird. Wenn sie grün leuchtet, ist der Kernel aktiv, wenn sie rot leuchtet, ist der Kernel ausgeschaltet. Die mittlere LED zeigt an, ob der Nutzer einen Listener/Chat-Server eingerichtet hat und die dritte LED zeigt an, ob der Nutzer eine aktive und erfolgreiche Verbindung zu einem Nachbarn/Server bestehen hat.

Abbildung: Verschlüsselung zwischen Kernel und Gui/Benutzeroberfläche




Die Verbindung der Benutzeroberfläche (goldbug.exe) zum Kernel (spot-on-kernel.exe) ist ebenso verschlüsselt, obwohl beide auf der selben Maschine laufen. Ein Tooltip bei Maus über die erste LED zeigt die Verschlüsselung an.

## 5.4 Einen Nachbarn mit der IP-Adresse verbinden

Bei der erstmaligen Aktivierung wird die IP-Adresse des Projekt-Chat Server automatisch als Nachbar hinzugefügt. Dieser dient als temporärer Chat-Server, über den der Nutzer mit seinen Freunden test-weise chatten kann, bis ein eigener Verbindungsknoten auf einem Webserver oder Zuhause erstellt wurde (oder zwei Nutzer direkt miteinander verbinden). Der Test-Server wird nicht dauerhaft bestehen, insofern werden Nutzer ab einem gewissen Zeitpunkt zunächst erst selbst einen Server aufsetzen müssen, bevor sie zwei Klienten daran verbinden können.

Durch den bisherigen Test-Server ist der Nutzer bislang direkt nach der Aktivierung des Kernels mit einem Chat-Server verbunden, wenn der Nutzer einen weiteren hinzufügen möchte, ist der Tabulator “Nachbar verbinden” zu nutzen. Hier zeigt sich ein Eingabefeld für die IP-Adresse des Nachbarn bzw. des Webservers, auf dem ein Spot-On-Kernel läuft bzw. ein Freund ebenso einen GoldBug Messenger nutzt.

Abbildung: Füge einen Nachbarn/Server hinzu

Abbildung: Füge einen Nachbarn/Server hinzu

Abbildung: Verbindung mit einem Nachbar-Server

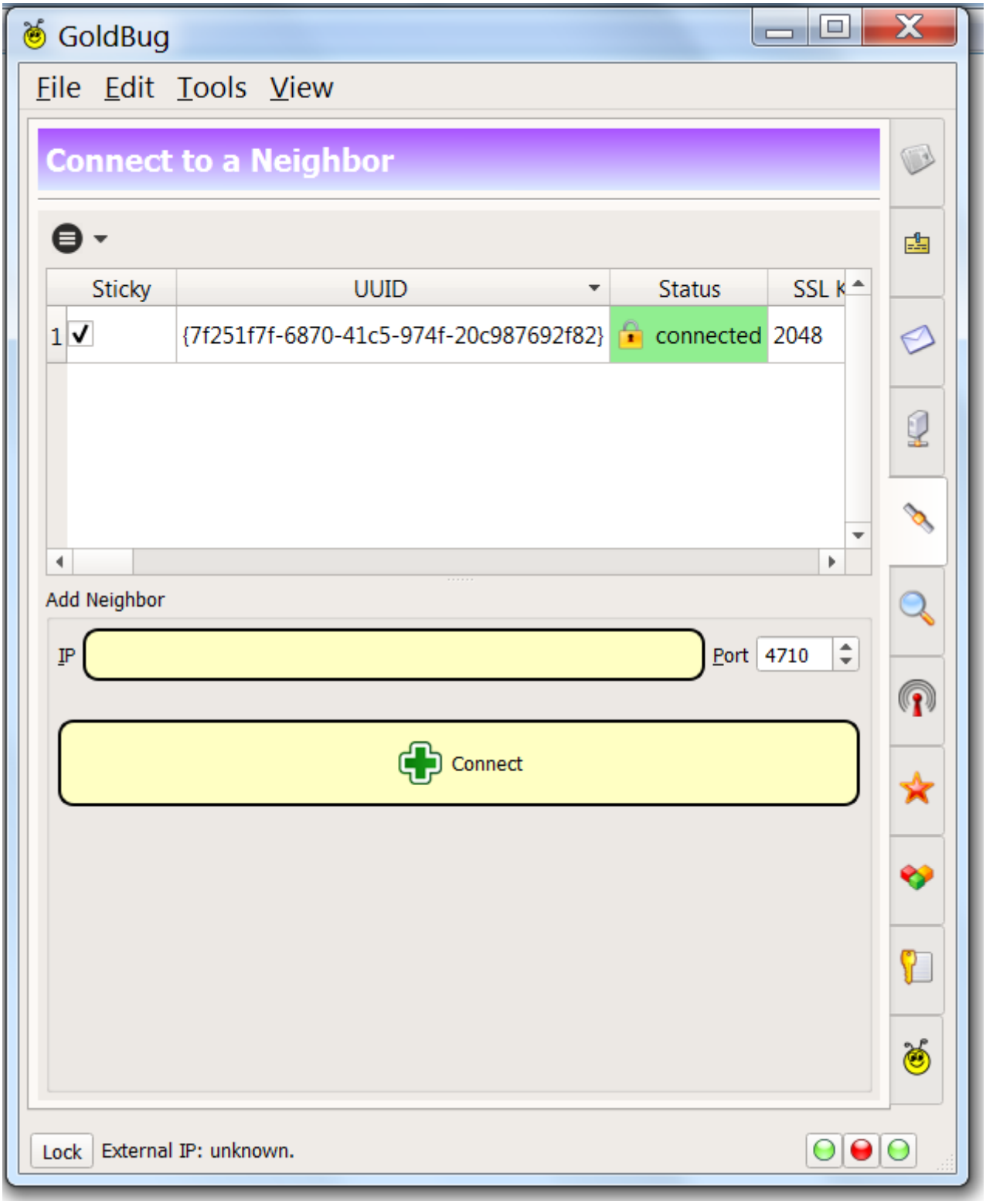
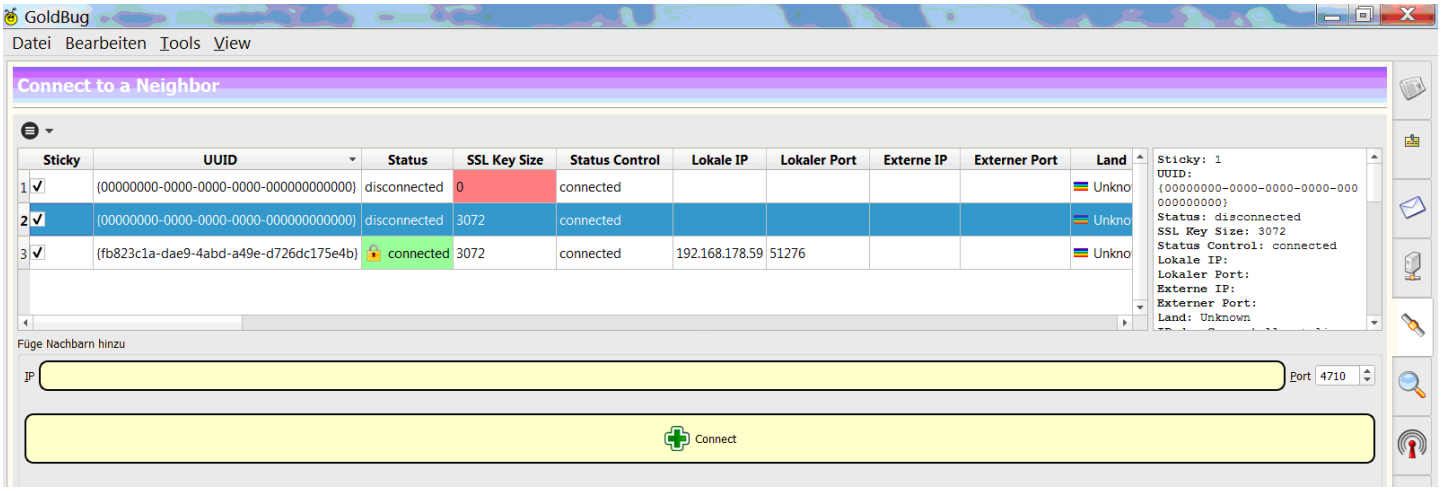



Abbildung: Nachbar-Server verbinden




In das Feld ist die IP-Adresse des Nachbarknoten einzugeben. Mit den Punkten sind jeweils drei Stellen der IP Adresse (nach IP-V4) getrennt. Umfasst ein Block nur zwei Stellen, z.B. in 37.100.100.100, dann kann die 37 in dem ersten Block beliebig platziert werden oder als 37 auf den ersten beiden Positionen eingegeben werden. Sodann ist der Knopf „Verbinden“ bzw. „Hinzufügen“ zu drücken. Die IP-Adresse ist sodann auf dem voreingestellten Port 4710 hinterlegt und erscheint als Verbindung in der Tabelle der Nachbarn.

Wenn eine Fehlermeldung erscheint, dann ist diese IP-Adresse schon eingegeben. Um alle Nachbarn zu löschen, kann dann der Knopf „Alle Nachbarn löschen“ gedrückt werden (über den Kontext-Menü-Knopf oder über rechte Maustaste in der Tabelle, in der der Nachbar erscheint) und die IP-Adresse kann erneut eingeben werden. Wahlweise kann im Installations-Pfad „./spot-on“ auf der Festplatte auch die Datei „neighbors.db“ gelöscht werden. Sie bildet sich sofort neu und ist dann leer.

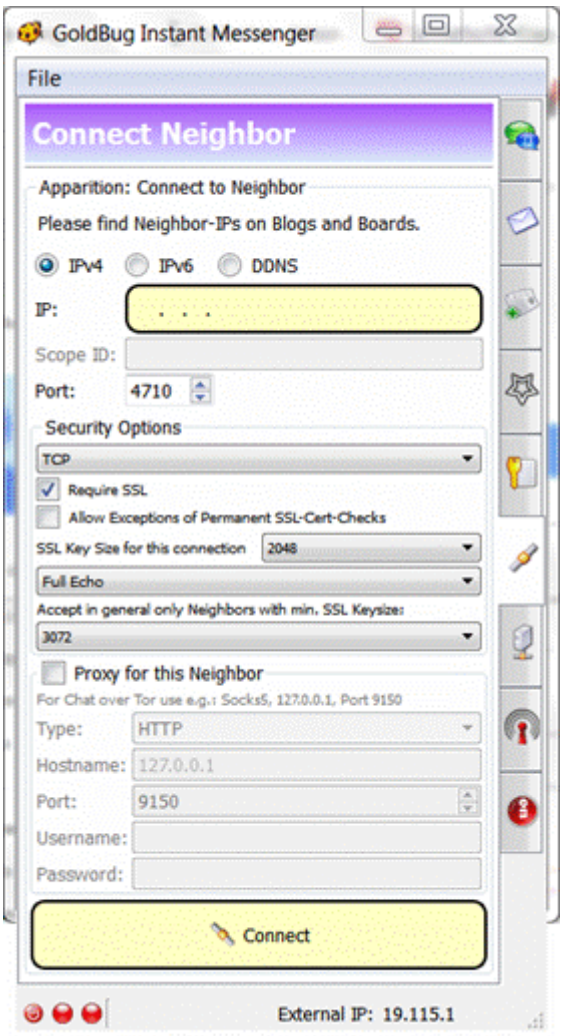
Wenn der Kernel aktiviert ist (linke, erste LED in der Statuszeile leuchtet grün) und der Nachbar bzw. Server verbunden ist (mittlere LED leuchtet grün), ist alles erfolgreich installiert und online. Eine IP-Adresse einzugeben und den Verbindungsknopf zu drücke, sollte insofern ganz einfach gelingen.

Wenn der Nutzer ohne Server direkt mit einem anderen Nutzer verbinden will, muss einer von beiden einen sogenannten Listener im Tabulatur Chat-Server erstellen (und für den Port die Firewall freigeben und ggf. den Port im Router zusätzlich an seine Maschine weiterleiten, siehe auch unten ausführlicher). Oder falls sich beide Nutzer in demselben Windows-Netzwerk befinden, kann der bereits vorhandene Nachbar „239..“ aktiviert werden, dann wird der GoldBug Messenger zum Lan-Messenger umgewandelt und findet alle anderen GoldBug-Teilnehmer im lokalen LAN automatisch und verbindet sie als Nachbar. Tauschen nun die Nutzer noch die Schlüssel, kann die Kommunikation starten.

Wer mehr Details sehen will, kann die minimale Ansicht auch in die volle Ansicht wechseln: In dieser Ansicht wird deutlich, dass neben der IP-Adresse auch der Port der IP-Adresse individuell konfiguriert werden kann. Standardmäßig nutzt GoldBug den Port 4710. Ferner kann das Programm auch über IPv6 betrieben werden sowie einen Listener/Server ansteuern, der über das Dynamische DNS verlinkt ist. Bei DNS gibt man dann keine Nummernfolge bei der IP ein,

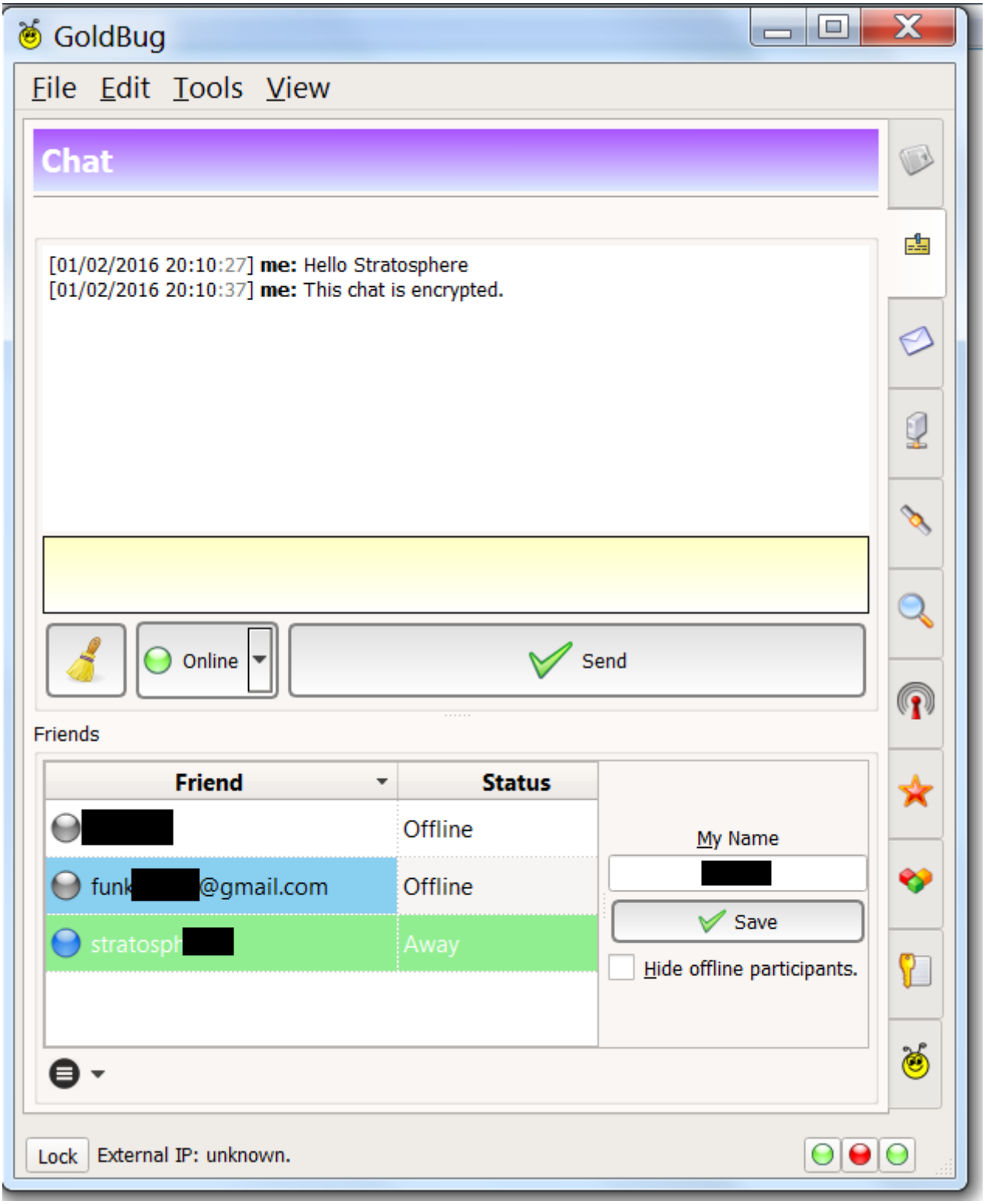
sondern einen Domain-Namen. In der darunter liegenden Box können weitere Sicherheitsoptionen definiert werden oder auch der Verbindungsserver über einen Proxy angesprochen werden (z.B. wenn man GoldBug über das TOR-Netzwerk nutzen möchte).

Abbildung: Einen Nachbarn mit IP-Adresse hinzufügen (ausführliche Ansicht




## 6 Die Chat-Funktion

Abbildung: Chat Tab




Wenn nun Login-Passwort definiert, Schlüssel generiert, Kernel aktiviert und ein Nachbar/Server verbunden ist, also in der Statuszeile zwei LED grün leuchten, dann kann der Nutzer mit einem Freund seinen Schlüssel tauschen und die Kommunikation kann im Chat-Tab (siehe Abbildung) oder Pop-Up-Fenster für einen definierten Teilnehmer beginnen.

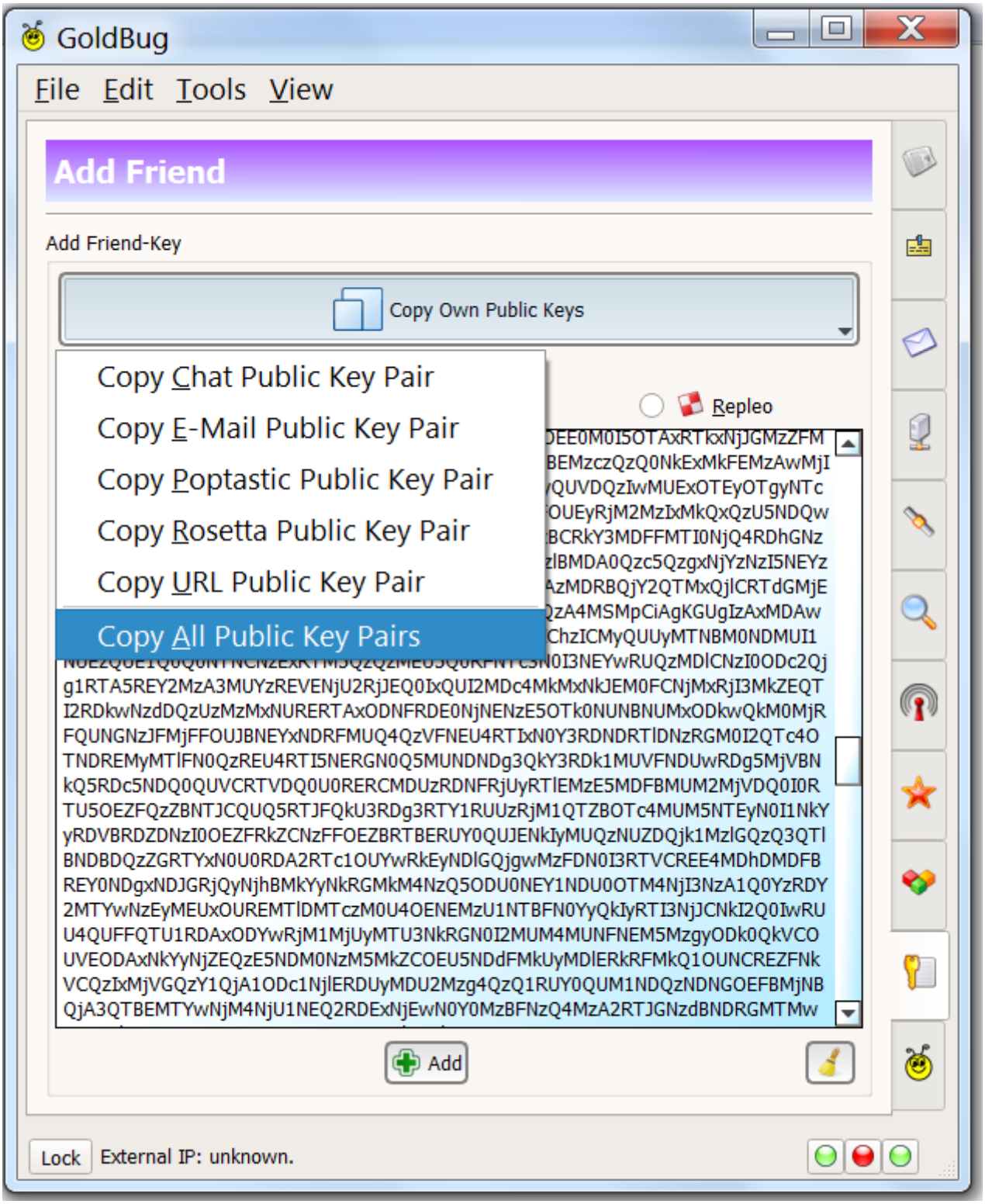
Der Schlüsseltausch lässt sich wie folgt beschreiben:

## 6.1 Freund hinzufügen durch Tausch und Einfügen der Schlüssel

GoldBug nutzt eine öffentliche/private Schlüssel-Infrastruktur, wie es bei der Verschlüsselung bekannter Standard ist: Der öffentliche Schlüssel kann mit Freunden getauscht werden und der private Schlüssel bleibt auf Festplatte des Nutzers in einem wiederrum verschlüsselten Container, der für die Applikation-Laufzeit mit dem Login-Passwort geöffnet (gemountet) wurde.

Der Nutzer und sein Partner, beide Freunde, müssen zuvor einmalig jeweils ihren öffentlichen Schlüssel tauschen, d.h. auskopieren und sodann den Schlüssel des Freundes im eigenen Tabulator: „Freund hinzufügen“ („Add Friend/Key“) einfügen und bestätigen (vgl. Abbildung). Der Freund kann seinen Schlüssel per E-Mail oder einem anderen Messenger senden. Der Nutzer kopiert den Schlüssel dann in diesen Tabulator ein und drücke den „Hinzufügen“-Knopf am unteren Rand.

Abbildung: Add Friend/Key




Der Nutzer findet seinen eigenen öffentlichen Schlüssel ebenso im Tabulator „Freunde hinzufügen“ („Add Friend/Key“). Über den großen Knopf (“Kopiere Schlüssel”) oben kann der Nutzer alle seine (oder ausgewählte Funktions-)Schlüssel in die Zwischenablage auskopieren. Der Nutzer kopiert hier also den vollen Text und sendet diesen zu seinem Freund. Ebenso macht es der Freund und der Nutzer fügt den Schlüssel des Freundes in das Textfeld ein.

'''Optional nur zur Kenntnis:''' Ggf. kann es notwendig sein, mit der rechten Maustauste im Kontext-Menü einen neuen Freund als Freund zu bestätigen (Make-Friend-Funktion). Dieses kommt dann zum Einsatz, wenn ein Freund seinen Schlüssel online in einer direkten IP-Verbindung an den Nutzer sendet. Diese Funktion ist in der Benutzeroberfläche von Spot-on

gegeben, in der GoldBug Benutzeroberfläche steht dieses nicht zur Verfügung, so dass beide idealerweise immer den Schlüssel füreinander einfach auskopieren, e-mailen und einfügen. Sollte aber ein Freund z.B. den Spot-on Klienten mit der dortigen Benutzeroberfläche nutzen und eine direkte IP-Verbindung zu einem Nutzer des GoldBug-Klienten aufbauen, dann ist es ebenso möglich, den Schlüssel auch per direkter IP-Verbindung zu transferieren anstelle vom Copy/Paste. Sodann erscheint der Freund mit seinem Nick-Namen im Tabulator Chat oder E-Mail (mit differierenden Icon) und ist mit rechter Maustaste oder aus dem Kontext Menü heraus als Freund zu bestätigen. Dieses ist eine Weiterentwicklung des Repleo, also der Funktion, den eigenen Schlüssel mit dem öffentlichen Schlüssel des Freundes (bei Erhalt) ebenso vor der Rück-Übertragung zu verschlüsseln. Hierbei wird der Schlüsselaustausch also automatisiert: es folgt ein Synchronisierungsprozess. Der Nutzer muss zustimmen, dass der Schlüssel nach Synchronisierung über die Nachbar-Verbindung im eigenen Klienten bzw. der eigenen Freundesliste angezeigt wird.

'"Weitere Option nur zur Kenntnis:"' Neben dem Online-Versand des Schlüssels über die direkte Verbindung zu einem Freund kann auch das weiter unten beschriebene Werkzeug des Echo Public Key Sharing (EPKS) genutzt werden. Dieses wird angewandt, wenn der Freund nicht mit einer direkten Verbindung verbunden ist (z.B. beide Partner einen gemeinsamen Chat-Server bzw. Knotenpunkt in der Mitte nutzen). Beide Partner geben dann in EPKS ein gemeinsames Passwort-Geheimnis ein und senden ihre öffentlichen Schlüssel über dieses Passwort ins Echo-Netz. Siehe dazu weiter die ausführlicheren Details im Abschnitt diesem Tool, das ggf. eine gute Alternative zu den oftmals unkomfortablen und unsicheren üblichen Key-Servern darstellen kann. Diese Innovation von Repleo und der Synchronisierung der Schlüssel über die sogenannte Echo-Public-Key-Share-Funktion (EPKS) bzw. die bestehende IP-Verbindung ist später von anderen Projekten ebenso aufgegriffen (kopiert) worden unter dem Namen AutoCrypt bzw. KeySync. Diese Funktionen gehen also auf die Repleo, EPKS und den Schlüsseltausch per IP-Verbindung von Echo-Knotenpunkten zurück.

### 6.1.1 Besonderheit: Repleo

Wenn der Nutzer schon einen Schlüssel seines Freundes erhalten hat (z.B. den Chat-Schlüssel) und diesen in seinen Klienten eingefügt hat, nunmehr aber seinen eigenen öffentlichen (Chat-)Schlüssel nicht der Öffentlichkeit preisgeben will, ihn nicht bei einem E-Mail-Programm gespeichert und transferiert wissen will (obschon der öffentliche Schlüssel ja eigentlich öffentlich sein darf), dann kann der Nutzer auch mit dem erhaltenen Schlüssel seines Freundes seinen eigenen öffentlichen Schlüssel verschlüsseln. Das nennt man dann REPLEO. Der Schlüssel wird verschlüsselt übertagen, alsbald ein Nutzer bereits einen öffentlichen Schlüssel des Gegenübers erhalten hat.

Beim Repleo wird also der öffentliche Schlüssel des Nutzers mit dem öffentlichen Schlüssel des Freundes bereits verschlüsselt. Dieses ist sodann für jede Funktion bzw. jeden Schlüssel durchzuführen, d.h. der Nutzer kann jeweils das Chat-Repleo, E-Mail-Repleo und URL-Repleo etc. zurücksenden. Auch ein Repleo kann der Freund dann in den Kasten des Tabulators “Add Friend/Key” einfügen. Über dem Einfüge-Kasten ist lediglich der Radio-Auswahl-Knopf zu

definieren, ob es sich um einen Key, ein Repleo, oder eine E-Mail-Adresse handelt, die man hinzufügen möchte. Mittlerweile sind die K- und R- Auswahlknöpfe in GoldBug verschwunden, weil der Klient automatisch erkennt, ob es ein Key oder ein Repleo ist.

Der Text eines Schlüssels startet also immer mit einem Buchstaben “K” oder “k” und ein Repleo startet mit einem “R” oder “r”. Daran kann man es noch erkennen.

## 6.2 Einen ersten sicheren Chat beginnen

Der Nutzer findet nach erfolgreichem Key-Tausch seinen Chat-Freund im Tabulator “Chat”. Damit der Chat funktioniert, sollten beide Teilnehmer idealerweise die gleiche und aktuellste Version des Programmes nutzen, ihre Schlüssel generiert und ausgetauscht haben und zum einem Netzwerk-Knoten bzw. Chat-Server (Nachbarn) im Web verbunden sein. Wenn die ersten beiden LEDs in der Status-Zeile unten grün leuchten und der Name des Freundes im Chat-Tab auftaucht, sieht es schon gut aus.

Wenn der Online-Status des Freundes blau (abwesend), rot (beschäftigt) oder grün (gesprächsbereit) aufleuchtet, kann der Chat beginnen. Entweder markiert der Nutzer den Freund in der Tabelle und chattet aus dem Tab heraus, oder doppelklicket mit der Maus auf den Freund und ein Pop-Up Chat Fenster für diesen Freund öffnet sich.

Der Vorteil im Chat-Tab zu chatten ist, dass man gleich mehrere Freunde markieren kann, so dass die Nachricht alle Freunde erreicht. Wenn der Nutzer den Pop-up Chat nutzt (siehe Abbildung), dann muss der Nutzer nicht mehr auf die Markierung zur Selektion seines Freundes aus der Freundesliste im Chat-Tab achten.

Abbildung: 1:1-Chat im Pop-up Fenster

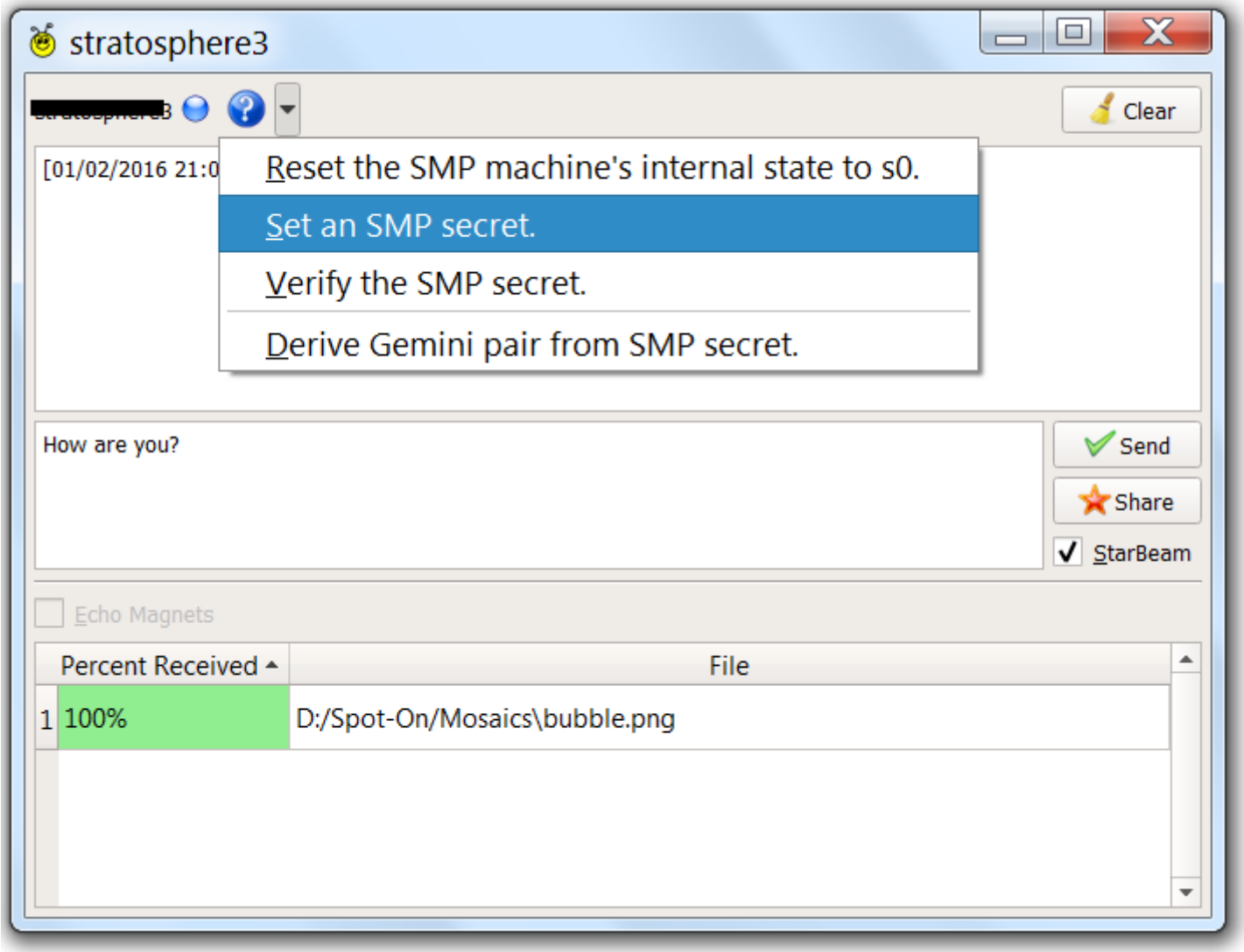



Und: Im Pop-Up-Chat hat der Nutzer den Options-Knopf “Share StarBeam”, mit dem er eine Datei von der Festplatte auswählen kann und diese dann verschlüsselt und sicher an den Freund transferiert (vgl. auch den Abschnitt weiter unten über StarBeam-FileSharing). Diese Funktion, dass man einen Chat-Freund per Mausklick einfach eine Datei senden kann und diese zudem vollautomatisch für den Transport von Ende-zu-Ende verschlüsselt ist, ist nicht in vielen

Applikationen enthalten. Verschlüsselte Übertragung z.B. eines ZIPs mit Urlaubsbildern zu seinen Geschwistern wird somit ganz einfach und ist ohne die Nutzung einer Hosting-Plattform im Web durchführbar.

In der Status-Zeile oben am Pop-Up-Fenster kann der Nutzer den Nicknamen und den Online-Status einsehen und z.B. auch das Socialist- Millionaire-Protokoll (SMP) starten, um einen Freund zu authentifizieren und zu testen, ob er (ein weiteres) gemeinsames Geheimnis kennt und richtig eingibt, wie es weiter unten beschrieben wird. Beide Nutzer sind dann authentisch, wenn sie das gleiche Passwort bei SMP eingeben.

Abbildung: Chat im Pop-up Fenster

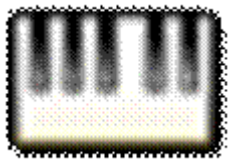



## 6.3 Zusätzliches Sicherheitsmerkmal: MELODICA: Calling mit einem Gemini

MELODICA steht für “Multi Encryted LOng DIstance Calling” – ins Deutsche übersetzt in etwa: „Vielfach-verschlüsseltes Anrufen über eine lange Distanz“

Das MELODICA-Symbol ist daher auch ein Keyboard eines Musikinstrumentes.

Abbildung: MELODICA-Symbol



Der MELODICA-Knopf führt die Calling-Funktion aus. Das „Cryptographische Calling“ ist vom Kernel-Projekt Spot-On entwickelt worden und sichert die Verbindung über eine sofortig erneuerte Ende-zu-Ende Verschlüsselung ab, indem das Passwort über die symmetrische Verbindung des Echo-Protokoll übertagen wird. Cryptographisches Calling mit dem MELODICA-Knopf bezeichnet, einen Freund wie mit einem Telefon anzurufen – nur, dass damit eine sichere Ende-zu-Ende Verschlüsselung aufgebaut wird.

Die Ende-zu-Ende Passphrase – auch Gemini genannt – wird über einen AES-String abgebildet und sollte zwischen beiden Teilnehmern geheim bleiben. Daher gilt es, die elektronische Übertragung immer gut über weitere Verschlüsselungs-Ebenen abzusichern (wie hier im Echo-Protokoll mit dem asymmetrischen Chat-Key und der TLS/SSL-Verbindung), wenn die Übertragung potentiell abgehört werden kann.

### 6.3.1 Asymmetrisches Calling

GoldBug hat diese Übertragungsfrage des Passworts für die Ende-zu-Ende Verschlüsselung gelöst, indem das Gemini (Zeichenfolge für die dann zu bildende symmetrische Verschlüsselung) asymmetrisch verschlüsselt wird (mit dem Schlüssel für Chat) und dann durch einen nochmals (asymmetrischen) verschlüsselten SSL/TLS-Kanal übertragen wird.

Gemini ist der griechische Begriff für Zwilling, d.h. es bezieht sich auf beide Teilnehmer, die die Passphrase sodann kennen sollten.

Diese Funktion erzeugt somit einen „Call“, einen Anruf, bei dem das Passwort übertragen wird, das dann später die Ende-zu-Ende Verschlüsselung gestaltet. Genau genommen besteht das Gemini aus zwei Schlüsseln bzw. Komponenten, denn das Gemini wird durch einen weiteren Prozess authentifiziert: Diese weitere Komponente wird auch MAC-Hash genannt, wie es oben schon erläutert wurde.

Das „Cryptographische Calling“ als ausführbares Protokoll mit dem MELODICA Knopf erweitert daher das bisherige Paradigma von Forward Secrecy wie folgt:

### 6.3.2 Instant Perfect Forward Secrecy (IPFS)

Der Nutzer kann somit die (symmetrische) Verschlüsselung bzw. das Gemini jederzeit erneuern. D.h. das Paradigma des „Perfect Forward Secrecy“ ist um zwei Komponenten erweitert worden: Einerseits kann man die Ende-zu-Ende Passphrase (das Gemini) manuell oder automatisiert definieren und sie andererseits auch sofortig, also „instant“ jederzeit erneuern. Daher wird von „Instant Perfect Forward Secrecy“ (IPFS) gesprochen.

Im Vergleich bieten viele andere Applikationen nur einen Key pro Online-Sitzung an und man kann die symmetrische Ende-zu-Ende-Verschlüsselungs-Phrase auch nicht manuell edieren.

Das hier angesprochene Instant Perfect Forward Secrecy (IPFS) nutzt die asymmetrische Verschlüsselung (des Chat-Keys), wobei der temporäre Schlüssel ein symmetrischer Schlüssel (eben das Gemini, ein AES-String) ist.

### 6.3.3 Symmetrisches Calling

Als weitere Option besteht bei GoldBug auch die innovative Möglichkeit, ein neues Gemini durch den Kanal eines bestehenden Gemini zu senden. Hier wird der Ende-zu-Ende Schlüssel (also das symmetrisch verschlüsselnde Gemini) durch eine andere Ende-zu-Ende Gemini-Verbindung gesandt (d.h. der neue symmetrische Schlüssel wird durch einen Kanal eines bestehenden symmetrischen Schlüssels mitgeteilt). Die symmetrische Verschlüsselungsphrase (das Gemini bzw. das AES-Passwort) wird also nicht mit einer asymmetrischen Verschlüsselung (dem Chat-Key) verschlüsselt (z.B. mit RSA, Elgamal. McEliece oder NTRU) und dann durch einen sicheren Kanal (SSL/TLS) von Punkt-zu-Punkt gesandt, sondern wird selbst mit dem bestehenden Gemini verschlüsselt und sodann erst durch die beschriebene Methode (wieder über SSL/TLS) gesandt.

Das Double-Rachet-Verfahren, bei dem der Schlüssel der folgenden Nachricht in dem verschlüsselten Inhalt des vorangegangenen Paketes ist, mag aus dem symmetrischen Calling angelehnt oder abgeleitet gewesen sein.

Somit können asymmetrische Calls und symmetrische Calls grundlegend unterschieden werden. Symmetrische Calls nutzen ein bestehendes Gemini. Asymmetrische Calls senden das Gemini über die asymmetrisch verschlüsselte Verbindung (nämlich den permanenten Chat-Key) zum Freund. Auch bei einem Call über ein bestehendes Gemini kann das gesandte Gemini jederzeit instant erneuert werden.

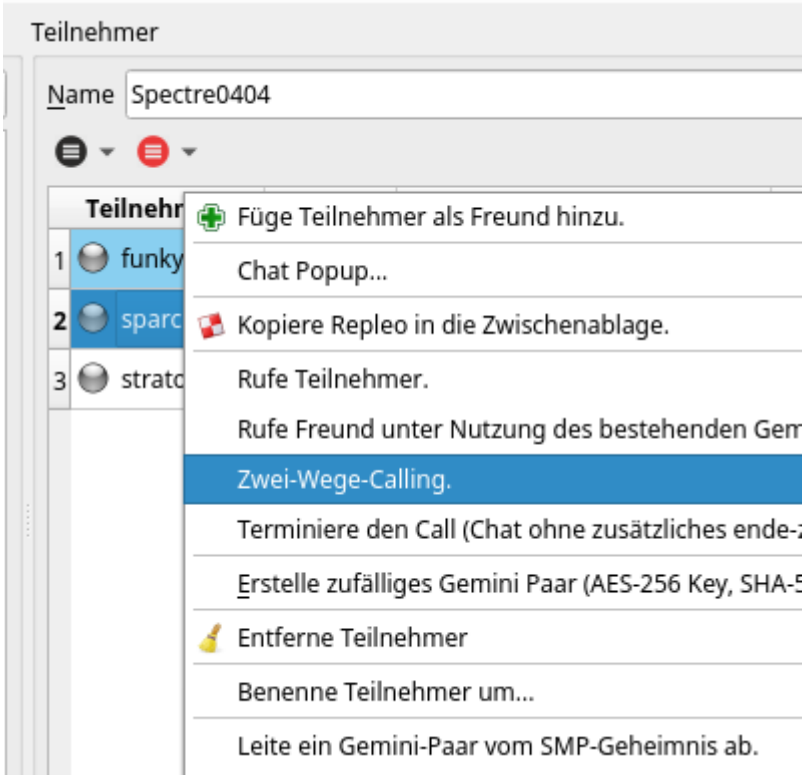
Sichere Ende-zu-Ende-Multi-Verschlüsselung entsteht, wenn ein Messenger einen manuell definierten symmetrischen Schlüssel mit einem bestehenden symmetrischen Schlüssel encodiert und dann mit einem asymmetrischen Schlüssel zusätzlich verschlüsselt. Und dieses Paket sodann durch eine sichere Verbindung gesandt wird.

### 6.3.4 Zwei-Wege-Calling

Schließlich ist im Kontext-Menü (gehe mit rechter Maustaste auf einen Freund in der Freundesliste) noch eine dritte Methode für einen sogenannten “Call” hinzugefügt: Das Zwei-Wege-Calling. Hierbei wird von dem Nutzer aus ein AES-256 als Passphrase für die zukünftige Ende-zu-Ende-Verschlüsselung an den Freund gesandt und der Freund sendet als Antwort ebenso ein AES-256 an den ersten Nutzer. Nun wird jeweils von dem AES des ersten Nutzers die erste Hälfte und von dem zweiten Nutzer die zweite Hälfte des AES genommen, und zu einem gemeinsamen AES-256 zusammengesetzt. Das benennt die Methode der 2-Wege-Sicherheit. Damit ist gewährleistet, dass kein Dritter - wenn es ihm gelänge, die Maschine des Freundes zu

kompromittieren - ein Gemini (oder ein altes Gemini) in seinem Namen von einer dritten, fremden Maschine sendet (was eigentlich nicht möglich ist, da es die unbemerkte Übernahme einer Maschine oder das Brechen der bestehenden TLS und RSA (bzw. NTRU- oder Elgamal-) Verschlüsselung bedeuten würde). Durch das Ping-Pong-Spiel beider Parteien im Zwei-Wege-Calling wird sichergestellt, dass beide Teilnehmer aktuell ihren Teil jeweils dazu beitragen, sich gegenseitig auf ein sicheres Ende-zu-Ende-Passwort zu einigen - und zwar Fifty-Fifty.

Abbildung: 2-Way-Calling im Kontext-Menü aus der Freundesliste




Die Möglichkeit, das Passwort

- erstens manuell zu edieren,
- zweitens sekündlich oder für jeden - oder innerhalb eines jeden - Anruf(es) erneuern zu können,
- drittens das Passwort durch eine bestehende Ende-zu-Ende Verschlüsselung zu versenden,
- und schließlich viertens, das Ende-zu-Ende-Passwort in einem zwei-Wege-Verfahren generieren zu können, macht es Angreifern somit sehr schwer, die Ende-zu-Ende-Verschlüsselung der GoldBug Calling-Funktion aufbrechen zu können.

Aus “Perfect Forward Secrecy” (PFS) ist nicht nur “Instant Perfect Forward Secrey” (IPFS) geworden, sondern (in dieser Funktion) sogar ein “2-Way Instant Perfect Forward Secrecy”: 2WIPFS. Diese Funktion hat damit FS und PFS und das wichtige Element der Ende-zu-Ende Verschlüsselung mit dieser Prozessimplementierung entscheidend weiterentwickelt. Die Verschlüsselung selbst ist dabei nicht neu, sondern lediglich der Verfahrensprozess ist ausgeklügelt implementiert, um mehr Sicherheit zu bieten.

Ende-zu-Ende Verschlüsselung wird im GoldBug durch einfaches Knopf-Drücken so einfach wie telefonieren: Einfach den Hörer aufnehmen oder wieder auflegen. Zu jeder Zeit bleibt die

Kommunikation asymmetrisch verschlüsselt und die symmetrische End-zu-Ende Verschlüsselung kann einfach hinzu geschaltet werden - und auch durch asymmetrische oder symmetrische Verschlüsselung (innerhalb eines SSL-Kanals) erneuert werden. Das ist ein neuer architektonischer Implementierungs-Standard, den diese Methode des Crypto-Callings etabliert.

## 6.4 Zusätzliches Sicherheitsmerkmal: Socialist Millionaire Protocol

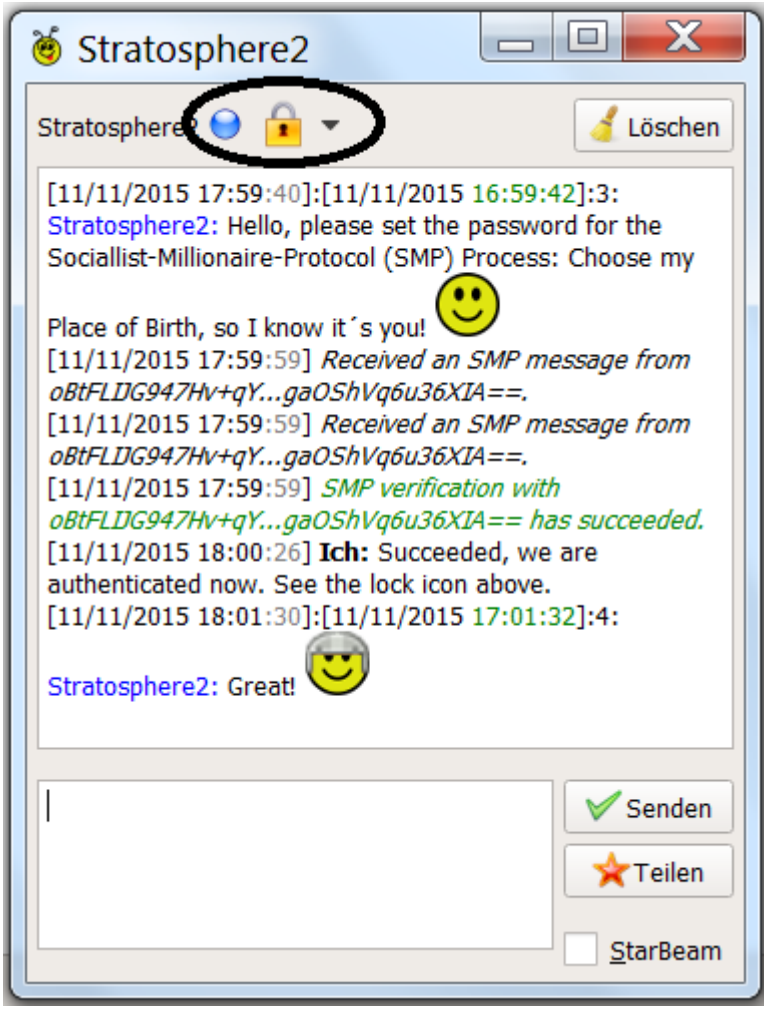
Während GoldBug die Nachrichten dreimal verschlüsselt -

- zum einen wird die Nachricht ja in einem sicheren TLS/SSL-Kanal gesendet,
- zweitens wird jede Nachricht asymmetrisch verschlüsselt (z.B. mit RSA, NTRU, McEliece oder Elgamal, also mit dem Chat-Key),
- und drittens besteht ja die die Möglichkeit, mit der “Call” bzw. “Anruf”-Funktion ein Gemini zu senden, um eine symmetrische Ende-zu-Ende Verschlüsselungs-Passphrase zu setzen (wie gesehen mit verschiedenen Methoden zur Durchführung des “Calls” innerhalb einer bereits bestehenden symmetrischen Verschlüsselung oder über die Zwei-Wege-Aufruf-Funktion, bei der jeder die Hälfte des Ende-zu-Ende Passwortes definiert) -

gibt es viertens zusätzlich ein weiteres Verfahren zur Erhöhung der Sicherheit: es ist das “SMP”-Protokoll: Socialist Millionaire Protocoll (eine Methode, die auch für Off-the-Record-Messaging (OTR) hier beschrieben wird: https://otr.cypherpunks.ca/Protocol-v3-4.0.0.html).

Die Idee dahinter ist, im Chat an den Freund eine Frage zu stellen wie: “Was ist der Name der Stadt, die wir gemeinsam im letzten Jahr besucht haben?”, oder eine Frage zu stellen wie: “Was ist der Name des Restaurants, in dem wir uns das erste Mal getroffen haben?” usw. (vgl. Abbildung).

Abbildung: SMP-Protokoll




Beide Teilnehmer signieren normalerweise die Nachrichten mit einem RSA (oder anderen) Algorithmus, um zu überprüfen, dass der verwendete Schlüssel vom ursprünglichen Absender ist. Aber für den (ggf. unwahrscheinlichen) Fall, dass eine Maschine gehackt würde oder falls der Verschlüsselungsalgorithmus gebrochen werden würde, kann mit dem Socialist Millionaire Protocol (SMP)-Prozess ein Freund einfach durch Eingabe des gleichen Passwortes auf beiden Seiten identifiziert werden. Es ist darauf zu achten, dass das Passwort nicht über den Chat zu senden ist, stattdessen sollte man eine Situation beschreiben, die zu dem gleichen Kennwort führt. Wird der SMP-Prozess das erste Mal getestet, kann man beiderseitig auch das Passwort “test” eingeben (kleingeschrieben).

Praktisch angewandt wird es wie folgt: Der Nutzer eröffnet für die Nutzung von SMP ein persönliches Pop-Up-Chat-Fenster und klickt das Fragezeichen-Symbol oben neben dem Benutzernamen des Chat-Freundes. Sodann wird ein Kennwort mit dem Menü definiert. Sodann ist der Chat-Freund zu fragen, das gleiche Passwort einzugeben. Drittens, klickt der erste Nutzer sodann schließlich auf die Schaltfläche “Überprüfen/Verify”.

Wenn beide Teilnehmer dasselbe Passwort eingestellt haben - bzw. der gleiche Hash-Wert vom gleichen Passwort generiert wurde - dann ändert sich das Fragezeichen-Symbol zu einem “Schloss”/”Lock”-Symbol. Der Chat-Freund wurde nun authentifiziert und der Chat bleibt weiterhin sicher.

SMP ist somit eine weitere ideale Möglichkeit, den Chat-Freund mit einem gemeinsamen Geheimnis sozusagen im Live-Prozess zu authentifizieren, es ist also keine zusätzliche Verschlüsselung!

Ein Beispiel verdeutlicht den Rechenprozess dieses Protokolls wie folgt. Nehmen wir in einem Beispiel an, Alice beginnt den Austausch:

'''Alice:'''

- 1. Nimmt die zufälligen Exponenten a2 und a3 auf
- 1. Sendet Bob g2a = g1a2 and g3a = g1a3

'''Bob:'''

- 1. Nimmt die die zufälligen Exponenten b2 und b3
- 1. Berechnet g2b = g1b2 und g3b = g1b3
- 1. Berechnet g2 = g2ab2 und g3 = g3ab3
- 1. Nimmt den zufälligen Exponenten r auf
- 1. Berechnet Pb = g3r und Qb = g1r g2y
- 1. Sendet Alice g2b, g3b, Pb und Qb

'''Alice:'''

- 1. Berechnet g2 = g2ba2 und g3 = g3ba3
- 1. Nimmt den zufälligen Exponenten s auf
- 1. Berechnet Pa = g3s und Qa = g1s g2x
- 1. Berechnet Ra = (Qa / Qb) a3
- 1. Sendet Bob Pa, Qa und Ra

'''Bob:'''

- 1. Berechnet Rb = (Qa / Qb) b3
- 1. Berechnet Rab = Rab3
- 1. Überprüft, ob Rab == (Pa / Pb)
- 1. Sendet Alice Rb

'''Alice:'''

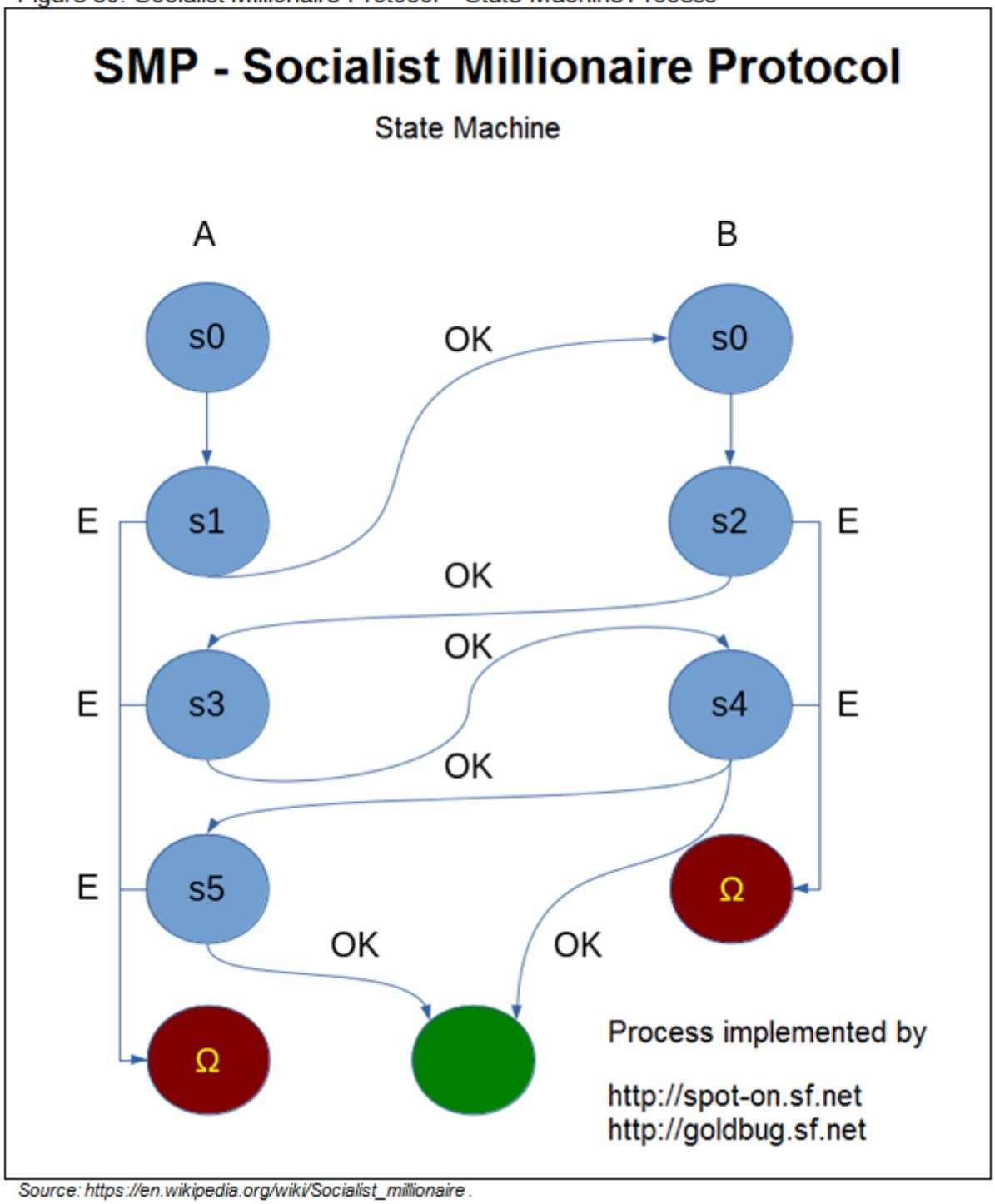
- 1. Berechnet Rab = Rba3
- 1. Überprüft, ob Rab == (Pa / Pb)

Wenn alles korrekt abgelaufen ist, dann sollte Rab den Wert erhalten von (Pa / Pb) mal (g2a3b3) (x - y), was bedeutet, dass der Test am Ende des Protokolls nur dann erfolgreich ist, wenn x == y ist. Weiterhin wird keine weitergehende Information enthüllt, als dass g2a3b3 eine zufällige Nummer ist, die zu keiner Seite bekannt ist, wenn x nicht gleich zu y ist! (vgl. auch die Formeln in der Dokumentation des Quellcodes).

GoldBug beschreibt während der verschiedenen Daten-Austausch-Vorgänge für SMP sogenannte “Zero-Knowledge-Beweise”. Ferner nutzt GoldBug den SHA-512 der jeweils eingegebenen geheimen Passphrase als die x- und y-Komponenten.

Abbildung: Socialist-Millionaire-Protocol (SMP) im Chat Fenster zur Authentifizierung des Chat-Partners

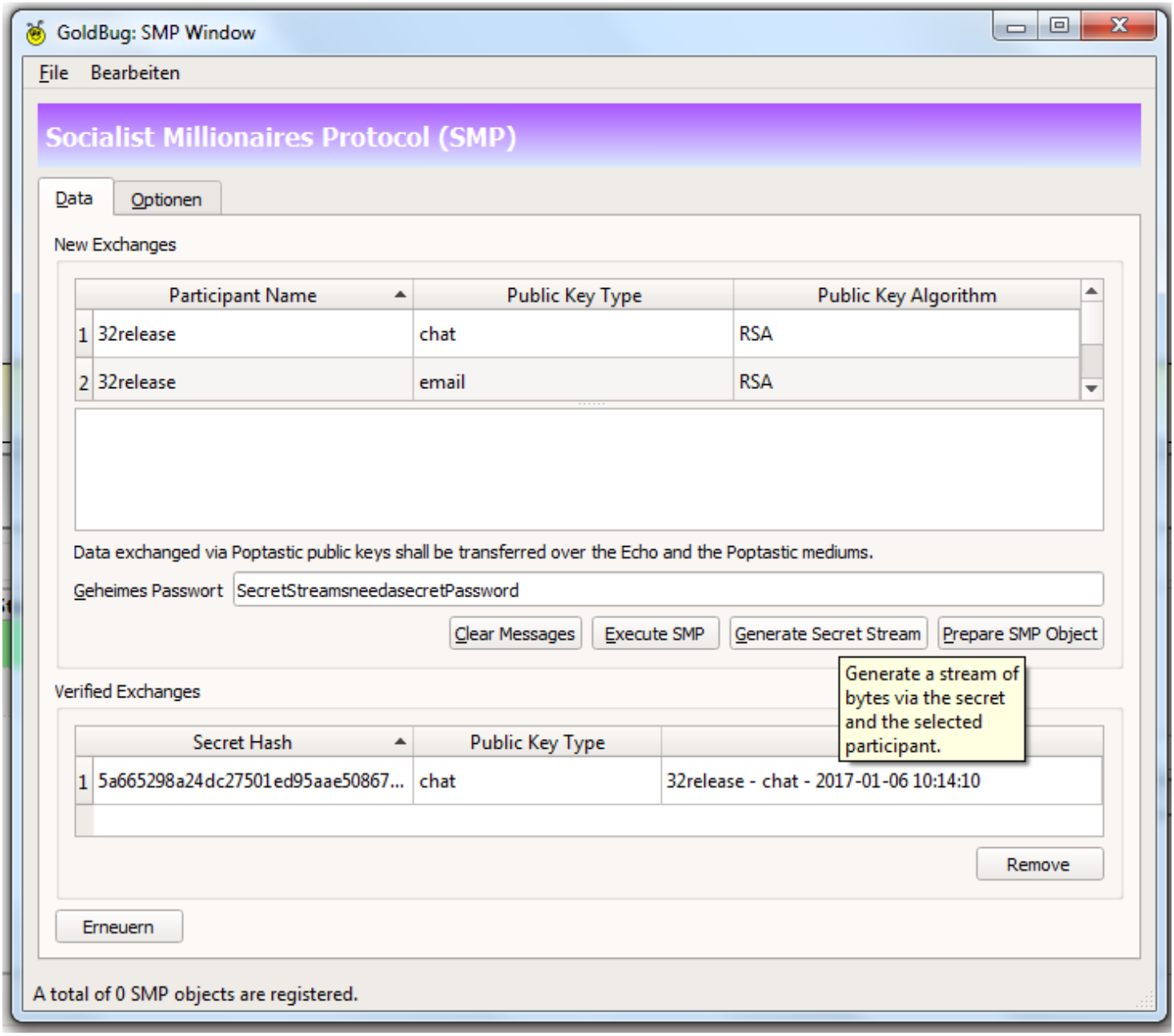
Figure 30: Socialist Millionaire Protocol – State Machine Process




Source: https://en.wikipedia.org/wiki/Socialist_millionaire .

SMP erfordert also, mit seinem Kommunikationspartner ein gemeinsames Geheimnis zu teilen. Ist das SMP-Passwort zugegen, kann es auch noch für weitere Funktionen als Grundlage dienen. Die für Forward Secrecy relevanten Secret Streams (vgl. auch bei der E-Mail-Funktion weiter unten) können daraus abgeleitet werden.

Abbildung: Implementierung von Secret Streams im extendierten SMP Protokoll




Die Secret Streams werden im Abschnitt unter E-Mail weiter erläutert, bleiben wir hier im Chat-Abschnitt zunächst bei der Funktion des Crypto-Callings, die sich ebenso auf dem erfolgreich verifizierten SMP-Passwort aufbauen kann. Dieses wird durch das SMP-Crypto-Calling beschrieben.

### 6.4.1 SMP-Calling

Oben haben wir die Call-Funktion erläutert, wie ein Gemini generiert und übertragen wird. Nun kann man das Gemini nicht nur manuell oder durch die AES-Funktion definieren, sondern es kann auch aus dem Passwort abgeleitet werden, das im SMP-Prozess wie oben dargelegt hinterlegt wurde. Somit wird die Passwort-Eingabe aus dem SMP-Prozesses genutzt (nicht der SMP-Prozess selbst). Es ist eine weitere Art des „kryptologischen Anrufens“ und damit seinem Gegenüber ein Ende-zu-Ende-Passwort sicher zu übertragen, das dieses Mal nicht aus einem AES-Generator entspringt! – falls jemand die Zufälligkeit eines maschinellen Zahlengenerators

anzweifelt. Sind die Grundfunktionen der Verschlüsselung in GoldBug erst einmal deutlich, erkennt man beispielsweise hier nun die Verwobenheit der einzelnen Prozesse in der Architektur, die damit nochmals mehr Sicherheit bietet.

## 6.5 Zusätzliches Sicherheitsmerkmal: Forward Secrecy (asymmetrisch)

Seit Version 2.7 unterstützt GoldBug Messenger Perfect Forward Secrecy auch für seine Funktion als E-Mail-Klient, und war damit der erste E-Mail-Klient, der Forward Secrecy für E-Mail sowohl mit symmetrischer als auch asymmetrischer Weise anbot (vgl. auch weiter unten).

Während oben für die Chat-Funktion das Crypto Calling mit einem Gemini das “Instant Perfect Forward Secrecy” prägte und sich auf einen symmetrischen Schlüssel (eben das Gemini bzw. den AES-String) bezog, ist das Perfect Forward Secrecy beim E-Mail mit temporären, a-symmetrischen Schlüsseln definiert.

Diese Variante der Nutzung von temporären asymmetrischen Schlüsseln kann natürlich auch wiederum auf die Chat-Funktion übertragen werden. Und dieses ist eben seit dem Release 2.7 erfolgt.

Während der Chat mit dem permanenten Chat-Key ja immer (asymmetrisch) verschlüsselt ist, wird nun noch ein temporärer asymmetrischer Schlüssel mit dieser neuen Schicht von Ende-zu Ende-Verschlüsselung eingesetzt. Dieser temporäre asymmetrische Key wird ephemeraler Schlüssel genannt. Dieser Schlüssel entsteht durch die Forward-Secrecy Funktion im Chat, die über das Kontext-Menü (rechter Mausklick) oder über den Menü-Knopf dargestellt wird.

Ein Tooltip auf dem Bildschirm (im Systray) zeigt an, wenn der Chat-Partner im Chat ein Forward-Secrecy mit temporären (ephemeralen) asymmetrischen Schlüsseln erzeugt hat, so dass der Nutzer dieses in seinem Klienten in einem Pop-Up-Fenster bestätigen kann. Der Nutzer schaut dazu unten in der Status-Zeile nach dem neu auftauchenden Symbol, klicke es an und kann dann in dem erscheinenden Pop-Up-Fenster den Forward-Secrecy-Prozess bestätigen. Sodann wird nicht mehr der (permanente) Chat-Schlüssel benutzt, sondern die neuen-temporären a-symmetrischen Schlüssel. Der permanente Chat-Schlüssel wird also praktisch ergänzt durch den temporären Chat-Schlüssel.

Nur wenige Software-Anwendungen verstehen Ende zu Ende Verschlüsselung auch asymmetrisch und bauen Forward Secrecy über eine asymmetrische Verschlüsselung darauf auf.

### 6.5.1 Forward Secrecy Calling

Somit lässt sich also auch wieder das Calling erweitern: Das symmetrische Gemini wird beim Forward Secrecy Calling (FSC) nicht wie oben beschrieben durch den permanenten (asymmetrischen) Chat-Key oder durch ein bestehendes (symmetrisches) Gemini gesandt, sondern durch den ephemeralen, temporären - und asymmetrischen - Chat-Key.

Während der Versand eines Geminis über ein bestehendes Gemini ein “symmetrisches” “Instant Perfect Forward Secrecy” definiert, kann der Versand eines Geminis über die ephemeralen Schlüssel des initiierten “Forward Secrecy” in der Chat-Funktion als ein “asymmetrisches” “Instant Perfect Forward Secrecy” bezeichnet werden.

(Aber auch der Versand eines Geminis über die permanenten Chat-Schlüssel ist als asymmetrisches “Instant Perfect Forward Secrecy” zu bezeichnen).

Während beim “Forward Secrecy Calling” und “dem Call durch ein Gemini” ein “Forward Secrecy” schon besteht und sodann die jederzeitige Erneuerbarkeit des Ende-zu-Ende Schlüssels definiert (Instant Perfect Forward Secrecy), ist bei den anderen Calling-Arten kein Forward Secrecy vorab gegeben, sondern Instant Perfect Forward Secrecy wird hier erst durch einen Call als Ergebnis des Calls erzeugt.

Die Fortsetzung von Forward Secrecy heisst also Forward Secrecy Calling.

## 6.6 Übersicht der verschiedenen Calling-Arten

Aus den beschriebenen Methoden (vgl. auch Abbildung), einen Ende-zu-Ende Schlüssel an den Freund zu übertragen, lässt sich folgende Übersicht gestalten, die die verschiedenen Methoden mit ihren jeweils spezifischen Merkmalen herausstellt.

**Übersicht der verschiedenen Calling-Arten mit jeweiligen Kriterien**

Abbildung: Übersicht der verschiedenen Calling-Arten mit jeweiligen Kriterien

| Kriterium | Asymmetrisches Calling | Forward Secrecy Calling | Symmetrisches Calling | SMP-Calling | 2-Wege-Calling |
|---|---|---|---|---|---|
| TLS/SSL-Verbindung | JA | JA | JA | JA | JA |
| Permanenter asymmetrischer Chat-Schlüssel | JA | JA | JA | JA | JA |
| Symmetrisches AES-Gemini als Kanal | NEIN | NEIN | JA | NEIN | NEIN |
| Halbes symmetrisches AES als Gemini (50 % AES + 50 % AES) | NEIN | NEIN | NEIN | NEIN | JA |
| Geheimes SMP-Passwort als Gemini | NEIN | NEIN | NEIN | JA | NEIN |
| Ephemerale asymmetrische Chat-Schlüssel | NEIN | JA | NEIN | NEIN | NEIN |
| Forward Secrecy als Voraussetzung | NEIN | JA | JA | NEIN | NEIN |
| Instant Perfect Forward Secrecy als Ergebnis | JA | JA | JA | JA | JA |

Die Call-Informationen - sprich die ende-zu-ende verschlüsselnde Passphrase - kann natürlich auch manuell, z.B. mündlich oder fernmündlich, übertragen werden. Nimmt man noch die oben genannten bestehenden fünf Call-Arten hinzu, kommt man dann insgesamt auf sechs verschiedene Arten, einen Call umsetzen zu können. Die Spot-on Architektur hat erstmals im Bereich von Crypto von “Calling” zur Übermittlung von Ende-Zu-Ende Passworten gesprochen und spätere Konzepte haben diesen Begriff in Anlehnung übernommen.

Bitte beachte folgende Erläuterungen:

- Jeder der dargestellten Methoden hat Instant Perfect Forward Secrecy (IPFS) zum Ergebnis.

- Nur das symmetrische und asymmetrische Calling erfordert keine Aktion auf Seiten des Gegenübers.
- Das “Forward Secrecy Calling” und das “Symmetrische Calling” setzten einen vorhandenen Status von Forward Secrecy voraus.
- "Symmetrisches Calling” und “Forward Secrecy Calling” haben dreifache Verschlüsselungs-Schichten (TLS/SSL, Permanenter Chat Key, sowie temporärer symmetrischer bzw. asymmetrischer Schlüssel, durch den das neue Gemini sodann gesandt wird).
- "SMP-Calling” und “2-Wege-Calling” durchbrechen die AES-Generierung, indem sie Teile der AES-Phrase ersetzen bzw. einen neuen Passwortstring bilden.

Die Nachrichtenformate mit den Verschlüsselungsebenen sehen dann vereinfacht - da Signaturen, HMACs und Hashes nicht einbezogen sind - wie folgt aus:

- Asymmetrisches Calling: (TLS/SSL (Permanenter Chat Key z.B. RSA (Nachricht ist ein AES-String)))
- Symmetrisches Calling: (TLS/SSL (AES (Permanenter Chat Key z.B. RSA (Nachricht ist ein AES-String)))
- Forward-Secrecy-Calling: (TLS/SSL (Permanenter Chat Key z.B. RSA (ephemerale Keys RSA (Nachricht ist ein AES-String))))
- SMP-Calling: (TLS/SSL (Permanenter Chat Key z.B. RSA (Nachricht ist ein String, der aus dem SMP gebildet wird)))
- 2-Wege-Calling: (TLS/SSL (Permanenter Chat Key z.B. RSA (Nachricht ist ein AES-String der zu 50% mit dem AES des Freundes modifiziert wird)))

Aus dieser Optionsvielfalt heraus sich in der Ende-zu-Ende Verschlüsselung abzusichern oder gar die Ende-zu-Ende Verschlüsselungs-Passphrase selbst zu definieren und manuell einzugeben, ist der Slogan, Claim oder die Headline für GoldBug entstanden: „Your Instant Definition in Decentralized Crypto“. Die Verschlüsselung ist somit nicht nur eine nutzerspezifische, die jederzeit (instant) erneuert werden kann, sondern auch eine die dezentral am Orte des Nutzers durch diesen Selbst definiert und gestaltet wird.

Abbildung: GoldBug Claim: Your Instant Definition in Decentralized Crypto

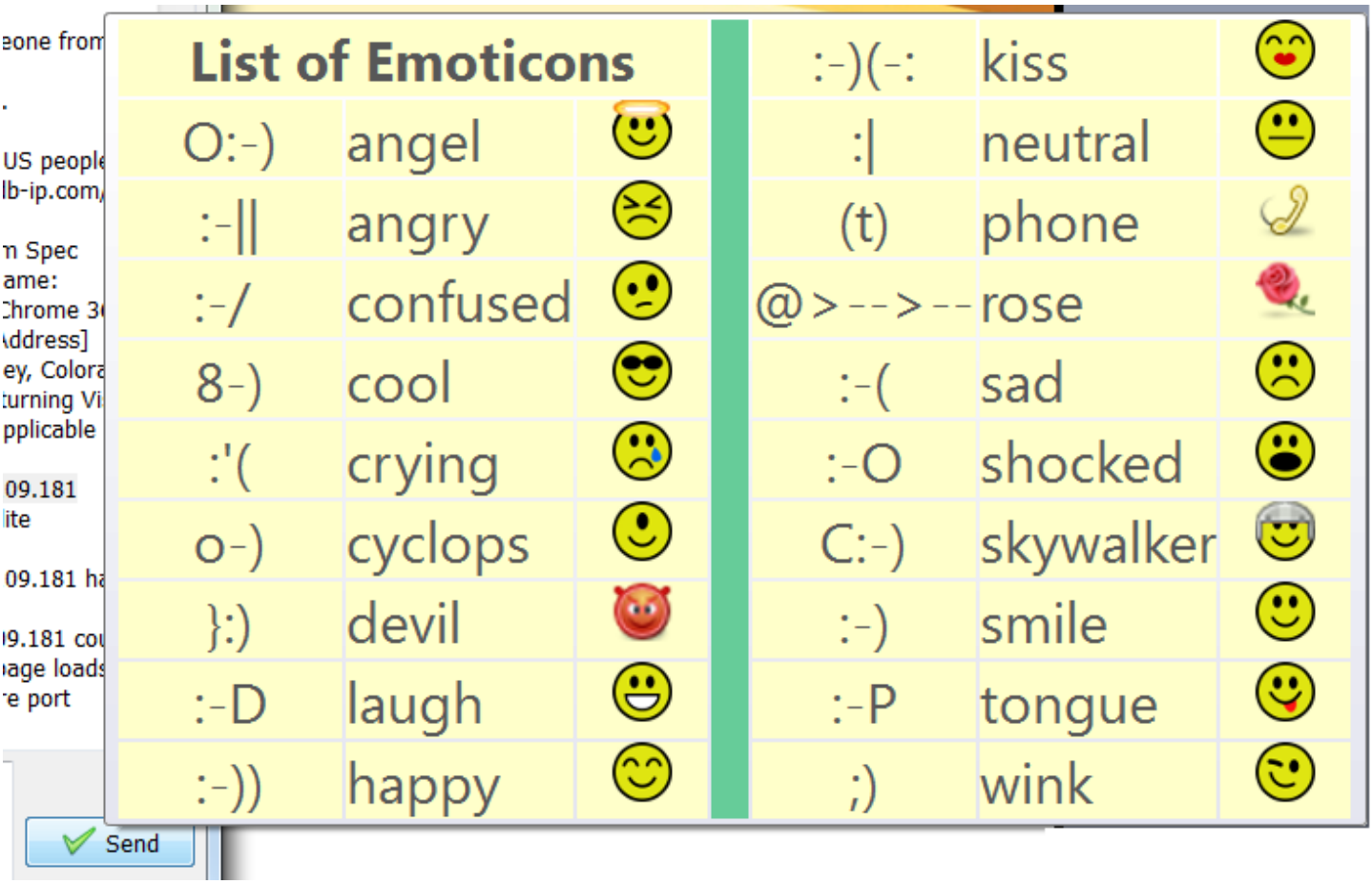



Ein einfacher Lackmus-Test im Vergleich mit anderen Software-Applikationen besteht in der einfachen Frage, ob der Nutzer das Ende-zu-Ende verschlüsselnde Passwort selbst eingeben kann. Mit GoldBug kann er es (wie auch mit der mobilen Version von GoldBug: Smoke Chat für Android).

## 6.7 Emotikons

GoldBug bietet eine Vielzahl an verschiedenen Emotikons - auch Smileys genannt - für den Chat an (siehe Abbildung).

Abbildung: Liste der Emotikons im GoldBug Messenger




Um diese zu nutzen, klickt der Nutzer zweimal auf einen Freund, so dass sich ein Pop-Up Chat-Fenster für den privaten Chat öffnet. Geht der Nutzer nun mit der Maus über den Senden-Knopf, werden in einem dann erscheinenden Tooltip die Smileys angezeigt. Mit der Eingabe des ASCI-Codes werden die Emoticons dann im Chat dargestellt.

Im Chat-Tab besteht in den Optionen des rechten Seiten-Splitters auch die Möglichkeit, die graphische Darstellung von Smileys generell auszuschalten.

## 6.8 Datei-Transfer im Chat-Pop-up-Fenster

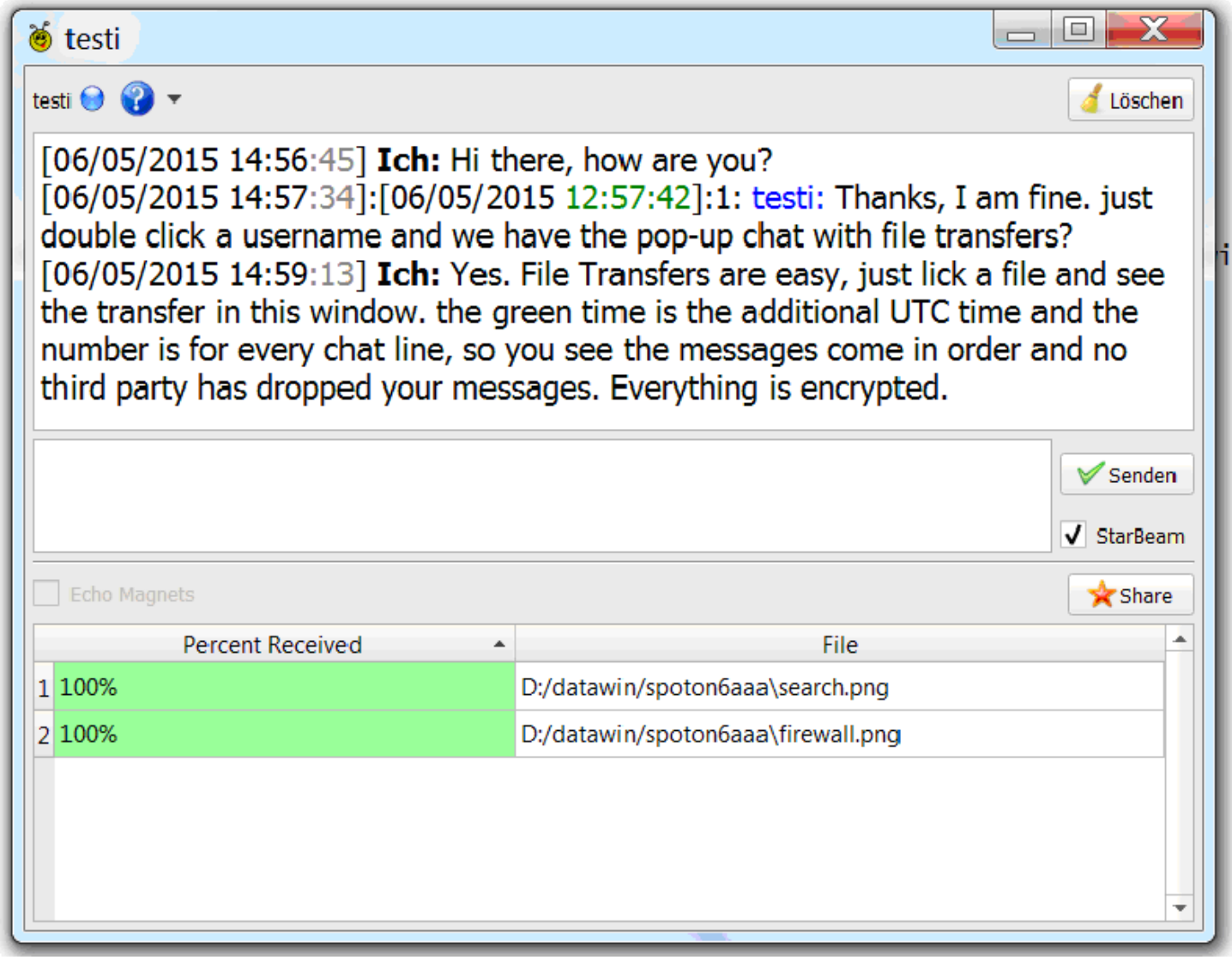
Das Qt Menü erlaubt es, aus der regulären Benutzeroberfläche einzelne Menü-Teile herauszunehmen und als Pop-Up Fenster zu gestalten.

Abbildung: Tear-Off/Hook-Up von Bedienungselementen




Ebenso ist insbesondere die File-Sharing Funktion in einem Pop-up Menü integriert: In dem 1:1 Chat Fenster. Wenn ein Nutzer also einem spezifischen Freund eine Datei senden möchte, kann er dieses einfach über den Knopf im Pop-Up Chat Fenster mit diesem Freund.

Abbildung: File Transfer im Chat Fenster

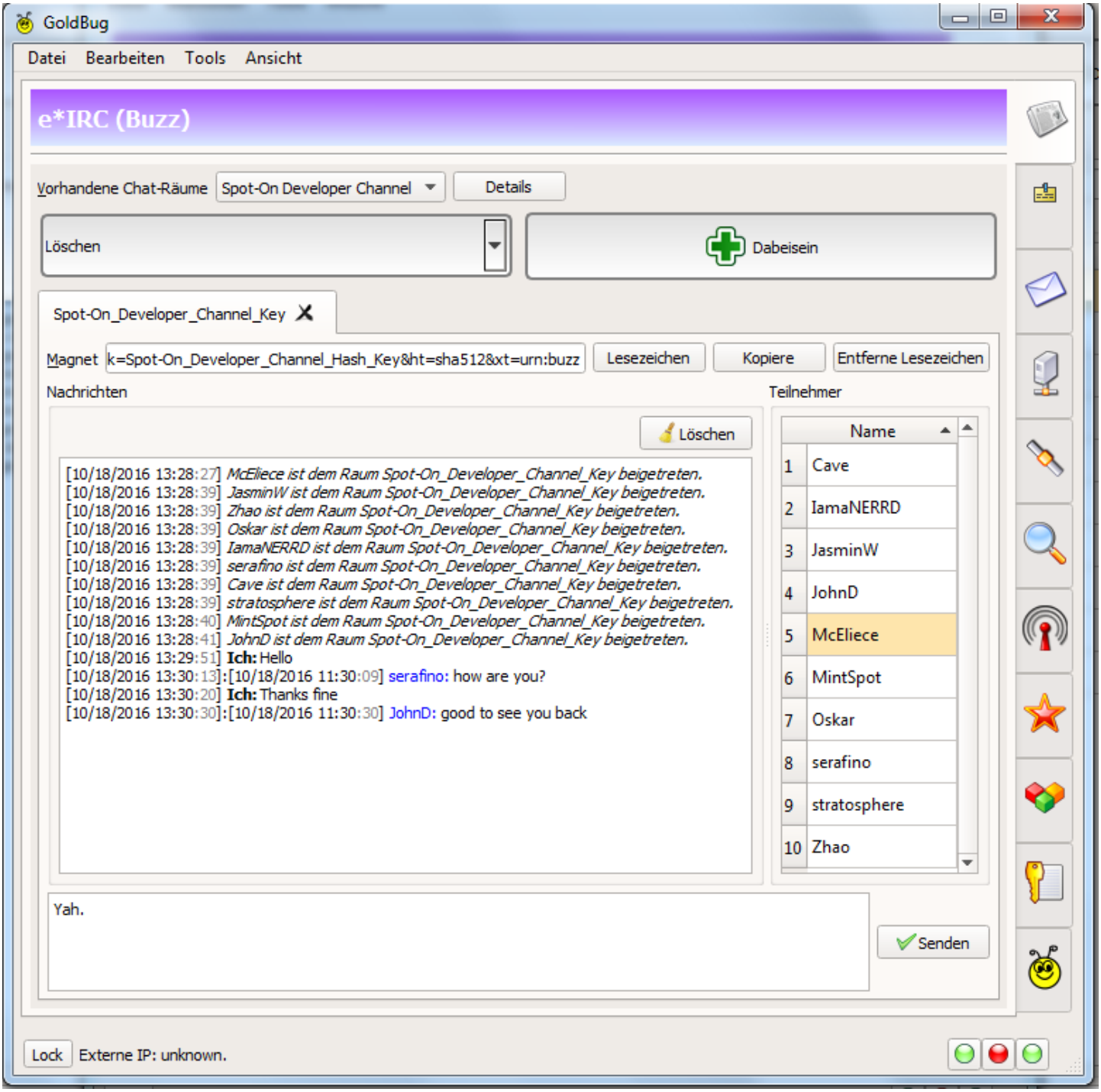



Die Datei wird ebenso wie der Text sicher und verschlüsselt an den Freund übertragen. Die Datei-Transfer-Funktion wird StarBeam genannt und hat auch einen eigenen Tabulator, ist im Chat-Fenster jedoch für eine einfache und direkte Bedienbarkeit bereits integriert. Eben auch ein kleines Hook-up-Menü.

## 7 Gruppen-Chat im IRC-Stil

Der GoldBug Messenger verfügt neben E-Mail und Chat und Datei-Übertragungen zu seinen Kommunikationspartnern auch über eine Gruppen-Chat-Funktion. Diese funktioniert ähnlich einem IRC-Chat. Die Übermittlung der Nachrichten an alle Gruppen-Teilnehmer erfolgt auch hier wieder vollständig verschlüsselt über das Echo-Protokoll. Die Verschlüsselung ist symmetrisch, also ähnlich einem Passwort-String. Letztlich können in dem p2p-Netzwerk oder über den Chat-Server alle Teilnehmer einen Gruppenchat mitlesen, die einen bestimmten symmetrischen Ende-zu-Ende-Schlüssel kennen, der den Chat-Raum definiert. Der Gruppen-Chat basiert ebenso auf dem Echo-Protokoll.

Abbildung: Der e'IRC Gruppenchat




Es wird daher von ge-Echo-tem IRC oder auch kurz e'IRC gesprochen, dass dem IRC-Chat neue Optionen eröffnet, da die Transportwege des e'IRC-Chats ebenso verschlüsselt sind. Wie heute normale POP3- oder IMAP-E-Mails auch zumindest eine Transportverschlüsslung z.B. mit TLS 1.3 aufweisen, kann sich IRC auch als ein verschlüsselter Gruppenchat verstehen. Auch der althergebrachte IRC-Chat kann daher von solchen Sicherheitsmodellen Verbesserungen erfahren: Der e'IRC Chat kann dazu das Modell einer neuen Gruppen-Chat-Generation darstellen.

Die Verschlüsselungs-Details des Gruppenchats werden wieder über einen Magnet-URI-Link (vgl. unten) definiert (definiert im Link mit Endung &URN=buzz). Buzz ist also die technische Bezeichnung im Quellcode für den e'IRC-Gruppenchat.

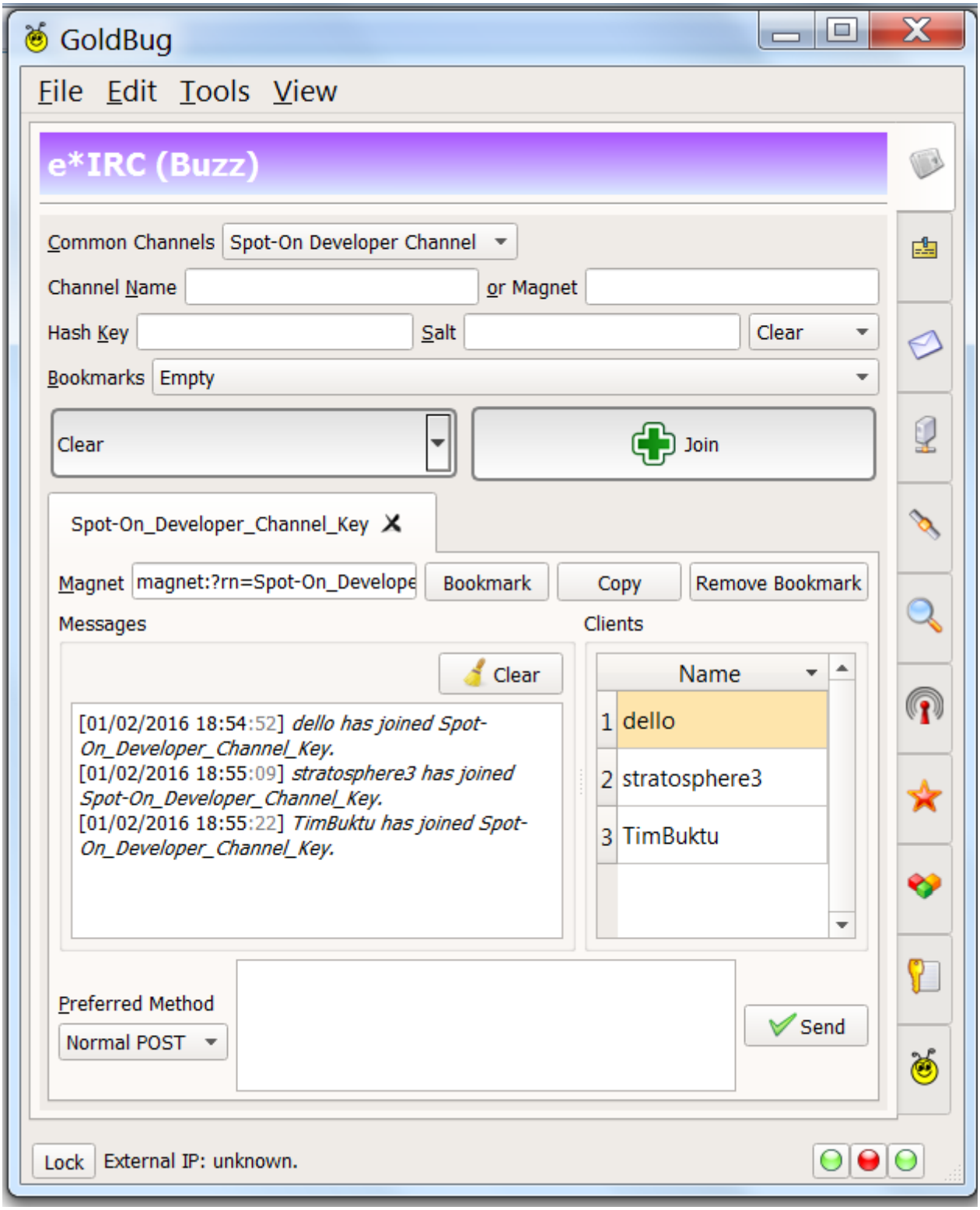
Zum Start des GoldBug-Programms wird der Community-Chat-Raum geöffnet, der als Beispiel dienen kann. Hier kann der Nutzer bei den anwesenden Nutzern auch weitere Anwendungsfragen zum Programm stellen oder diesen Kanal mit einem Freund nutzen.

Um einem eigenen Kanal beizutreten, gibt der Nutzer einfach einen Raum- bzw. Channel-Namen ein oder nutzt die oben angesprochene Methode des Magnet-Links. Der Magnet-Link hat neben dem Raum-Namen zusätzliche Werte für die Verschlüsselung eingebettet wie z.B. Schlüssel, Hash oder Cipher für den Verschlüsselungstyp.

Wenn der Nutzer nur den Raum-Namen eingibt, und keinen Magnet-URI verwendet, werden die zusätzlichen Verschlüsselungsdetails auf den Wert 0000 gesetzt und die Verschlüsselung des Raumes erfolgt auf Basis des Raum-Namens. Wenn der Nutzer alle Werte bzw. den Raum-Namen oder den Magnet Link eingegeben hat, ist der Knopf „Join/Beitreten“ zu drücken.

Sollte der Nutzer einen Magnet als Link eingefügt haben, dann ist zuvor im Pull-Down Menü der Befehl „de-magnetize“ zu nutzen. Der Magnet wird dann wieder in seine Einzelbestandteile zerlegt und der Raum wird auf Basis der im Magnet-Link eingebundenen Verschlüsselungswerte erstellt und betreten.

Abbildung: Gruppenchat im IRC-Stil im e’IRC-Buzz Kanal




Wenn der Raum geöffnet ist, kann der Nutzer den Raum auch als Bookmark abspeichern oder den entsprechenden Magnet-URI auch jederzeit aus seinen Chat-Raum-Bookmarks auskopieren und an seine Freunde senden, um sie in einen Raum einzuladen.

Um eine Nachricht zu senden, gibt der Nutzer sodann im Chat Raum einen Text ein und drückt den Senden-Knopf.

Der e'IRC Chat-Raum kann öffentlich oder privat sein, das hängt davon ab, wie sehr der Nutzer den Magnet bzw. die einzelnen Verschlüsselungswerte bekannt gibt. Als öffentlichen e'IRC Chat Raum kann der Nutzer den Magnet-URI auf seiner Webseite bekannt geben oder verlinken und jeder weiß, wie er in diesen Chat-Raum kommen kann: mit "de-magnetize".

Letztlich funktioniert es für den Nachrichtensender wie ein IRC-Chat, nur mit dem Unterschied, dass der Internet-Provider und weitere Rooting-Server nicht in die Kommunikation hineinsehen können, da sie ja verschlüsselt ist - wie eine Verbindung beim Online-Banking auch.

Es macht also keinen Unterschied mehr, ob ein Nutzer mit Freunden spricht oder seinem online Bank-Berater.

Wenn der Nutzer den Chat-Raum als privaten Raum nutzen will, kann der Nutzer den Magnet-URI auch geheim nur mit seinen Freunden teilen und man bleibt unter sich. Dieses ist eine bequeme Funktion des GoldBug Programmes: Der Nutzer kann einfach verschlüsselt chatten, ohne vorher asymmetrische Schlüssel tauschen zu müssen. Der Nutzer sagt seinem Freund einfach mündlich, er solle in GoldBug in den Raum “Bernsteinzimmer” kommen und beide Teilnehmer können dort sehr einfach sicher und verschlüsselt über einen gemeinsamen Chat-Server chatten.

Tip: Der Nutzer kann einen One-Time-Magnet für einen Raum erstellen und diesen nutzen, um seinen öffentlichen Chat-Schlüssel beim Tausch zum Kommunikationspartner zu schützen, indem der Nutzer diesen über den (selbst definierten) IRC-Channel dann nur seinem Freund bekannt gibt.

GoldBug ermöglicht mit dem Repleo, mit EPKS und dem Schlüsseltausch über einen One-Time-Magneten (OTM) für einen privaten e’IRC Chatraum also mehrere Methoden für einen gesicherten Schlüsseltransfer. Somit müssen öffentliche Schlüssel auch nicht mehr öffentlich sein.

# 8 Smoke Mobiler Chat Client

Während GoldBug ein Desktop-Klient ist, der auf zahlreichen Betriebssystemen und auch Plattformen wie Raspberry Pi kompiliert und einsatzfähig ist, heißt der mobile Klient des Echo Protokolls in Java-Programmierung für Android „Smoke Chat“.

## 8.1. Smoke Android Client

Smoke bietet direkten 1:1-Chat zu einem Freund sowie auch einen Gruppenchat. Dieser wird FireChat genannt und ist vergleichbar mit dem Buzz/e’IRC-Chat in GoldBug.

Der 1:1 Chat von Smoke auf dem mobilen Gerät nutzt nicht die Telefonnummer der Teilnehmer als Identifikationsmerkmal, sondern eine kurze Zeichenkette, ein sogenannter SIP-Hash, wird als Identifier genutzt.

Nutzer von Smoke verbinden also zu einem gemeinsamen Server – dieses kann ein Listener von GoldBug, Spot-On, oder auch Spot-On-Lite sowie SomkeStack-Server sein – und tauschen dann ihren öffentlichen Schlüssel über die SIP-Hash-Verbindung. SmokeStack ist dabei ein Chatserver

für Android und kann rund 500 Nutzer auf einem Android Gerät bedienen – ideal für eine Workshop-Gruppe, Familie oder Schule.

### 8.2. Fire to Buzz Chat

Da die Java und C++ Programmierung aus den Crypto-Bibliotheken heraus keine gemeinsame Schlüsselformate kennen, ist es normalerweise nicht möglich, mit einem quelloffenen Java-Klienten zu einem C++-Klienten verschlüsselt zu chatten.

Smoke hat jedoch hierzu jedoch eine Möglichkeit innoviert und implementiert: Über den sogenannten Fire-Chat in Smoke kann auch ein Nutzer in GoldBug angechattet werden und umgekehrt. Dieses basiert auf einen symmetrisch verschlüsselten Chat (ähnlich einem symmetrischen Crypto Call).

Andere Applikationen nutzen meistens die Java-Skript Crypto Bibliotheken für den Browser oder stellen eine Verbindung zum zentralen Server her, so dass diese Verfahren allgemein als unsicherer gelten.

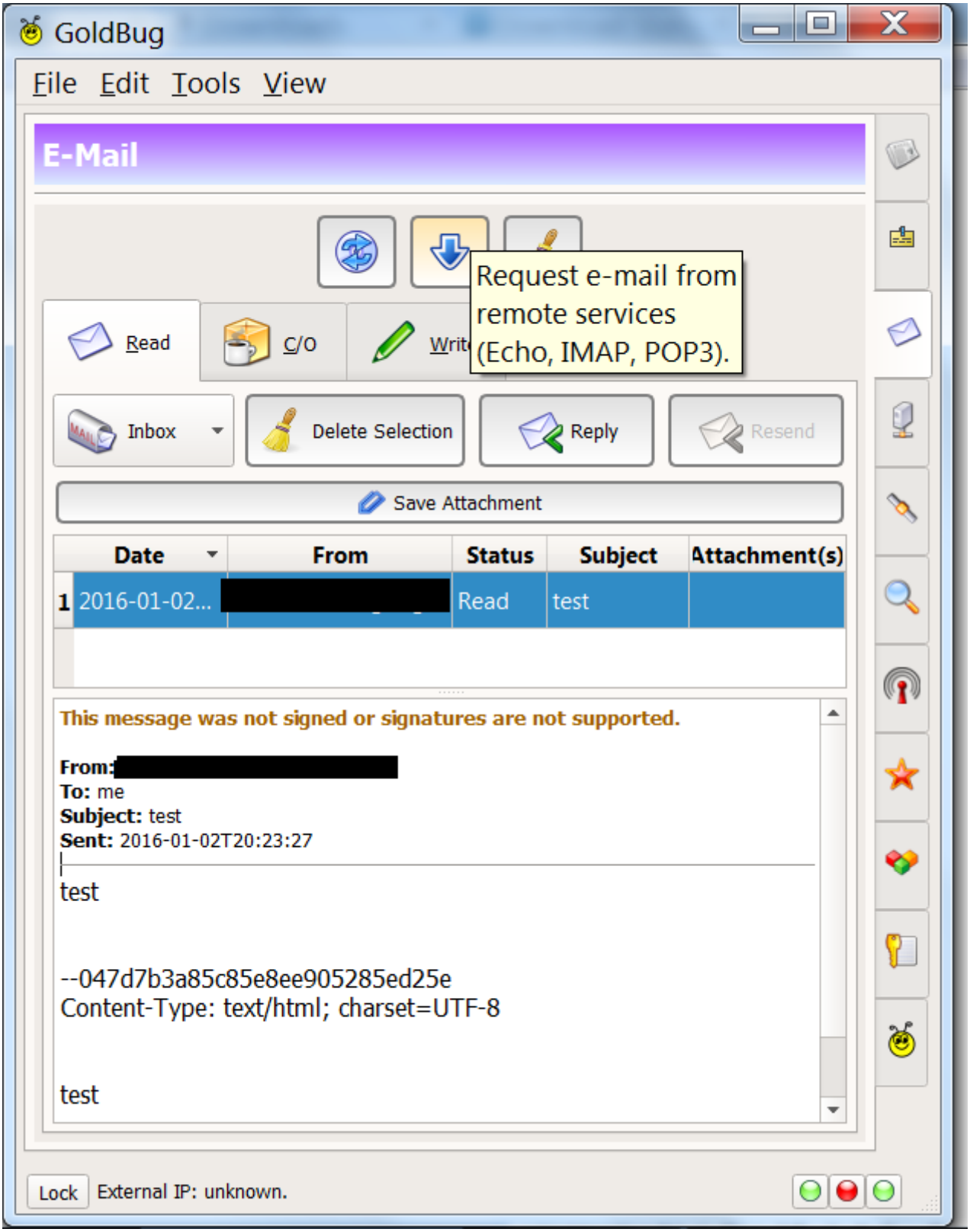
Wer also eine mobile Variante von GoldBug für den Chat einsetzen möchte, findet diese unter Smoke Chat sowie unter der deutschsprachigen Konzeptbeschreibung MOMEDO (beides bei Github: https://github.com/textbrowser/smoke).

## 9 Die E-Mail-Funktion

GoldBug ist ein voll funktionsfähiger E-Mail-Client.

Nicht vollumfänglich - wie seit Jahrzehnten bestehende E-Mail-Applikationen -, hier benötigt er noch weitere Programmierung aus der Community, aber dennoch voll funktionsfähig im Sinne eines vollwertig nutzbaren E-Mail-Klienten. Der Vorteil: das Lesen und Schreiben von E-Mails wird in der Benutzeroberfläche sehr effizient auf einer Seite bzw. in einem Tab dargestellt (vgl. Abbildung). Und: die E-Mails zu anderen GoldBug Nutzern sind immer verschlüsselt.

Abbildung: E-Mail - Lesen Anzeige




Technisch nutzt GoldBug die Bibliothek CURL und unterstützt POP3 mit SMTP sowie IMAP. Schließlich ist es die besondere Funktion in GoldBug, dass er drittens auch p2p E-Mail unterstützt.

Hierbei wird die E-Mail im distribuierten Netzwerk der Teilnehmer gespeichert und nicht bei einem zentralen Provider. Nutzer können also mit GoldBug selbst sehr einfach einen E-Mail Server bereitstellen und darüber kommunizieren. Die Infrastruktur ist nicht nur einfach zu installieren, sondern kann (damit) auch aus eigener Hand erstellt werden.

Perspektivistisch ist dieses auch die (notwendige) Zukunft, dass die Nutzer des Internets sich das Internet wieder stärker selbst organisieren (müssen) und neu, mit Kryptographie ausgestalten und zwar mit ihren selbst angelegten nunmehr verschlüsselten Mail-Postfächern, die nicht bei zentralen Hostern hinterlegt sind, sondern im eigenen Teilnehmernetz.

Denn: Auf Zentralisierung folgt meistens immer Dezentralisierung, auch wenn die Nutzer, die diese Freiheit erkennen und wert schätzen, erst zukünftig ihr Augenmerk auf Dezentralisierungs-Notwendigkeiten und verbleibende Chancen legen werden.

Wie folgt lässt sich die Einrichtung der drei Arten, seine E-Mails zu laden, beschreiben:

## 9.1 POP3

Das Post Office Protocol POP3 ist ein Übertragungsprotokoll, über das ein Klient E-Mails von einem E-Mail-Server abholen kann.

POP3 erlaubt das Auflisten, Abholen und Löschen von E-Mails am E-Mail-Server. Für das Versenden von E-Mails ist komplementär zu POP3 üblicherweise das Simple Mail Transfer Protocol SMTP in Clients und Servern implementiert.

Das POP3-Protokoll ist somit in allen verbreiteten E-Mail-Programmen integriert, so auch in GoldBug. Wie es - neben IMAP - eingerichtet wird, wird folgend und weiter ergänzend unten in der Beschreibung des Fensters von POPTASTIC erläutert (vgl. folgende Abbildung).

## 9.2 IMAP

Das Internet Message Access Protocol ( IMAP ) wurde in den 1980er Jahren mit dem Aufkommen von Personal Computern entworfen, um bei der Mail-Kommunikation die Speicherung der E-Mails auf einzelnen Klient-Rechnern aufzulösen.

D.h., die (PC-)Klienten greifen stattdessen online auf den Servern auf die Informationen zu und erhalten allenfalls Kopien davon.

Während ein Benutzer von POP3 nach Verlust seines PC alle E-Mails verloren hat, stellt ein Mail-Klient bei IMAP die Anfragen an den Server nur nach aktuell benötigten Informationen als Kopie.

Möchte ein Nutzer z.B. den Inhalt seines Eingangsordners sehen, holt sich der Klient eine aktuelle Kopie der Nachrichtenliste vom Server. Soll der Inhalt einer Mail angezeigt werden, wird dieser vom Server als Kopie geladen. Da alle Daten weiterhin auf dem Server verbleiben, wird somit eine “lokale Speicherung der Daten” unnötig und erweiterte Möglichkeiten wie das Durchsuchen von Mails werden ebenso nur server-seitig durchgeführt.

Damit wird auch die lokale Sicherung der Daten - indem sie von Server weggenommen werden - insofern meist unmöglich, als dass die Konfiguration von IMAP in der Voreinstellung nicht darauf ausgerichtet ist. Zugleich rückt bei unverschlüsselten Mails die Frage der Vertraulichkeit

und Sicherheit bei Daten, die auf IMAP-Servern ausgelagert bleiben, in den Vordergrund. Es stellt sich die Frage, ob der Empfänger einer E-Mail selbst die Hoheit über die Vertraulichkeit und Speicherung der E-Mail hat, und z.B. das Recht hat, diese niemandem zu zeigen oder im Geheimen zu löschen, oder ob er lediglich eine Kopie, ein “Ansichtsrecht” seiner Post erhält.

Hinsichtlich der Erkenntnisse aus dem Jahr 2013 - E-Mails besser grundlegend zu verschlüsseln - ist IMAP in diesem Lichte besonders kritisch zu beurteilen: Die Speicherung der E-Mails wird bei IMAP nicht wie bei POP3 in dem Mail-Klienten auf der Maschine des Nutzers vorgenommen, sondern die persönlichen Daten liegen unverschlüsselt weiterhin auf dem Server des Providers. Mit IMAP wurde somit die heute vielfach genutzte Cloud schon in den 80er Jahren im Bereich E-Mail erfunden. POP3 ermöglich eher eine On-Premises - Handhabung der Speicherung der E-Mails auf der lokalen Maschine.

GoldBug unterstützt diesen Standard ebenso wie POP3 und ermöglicht es, Plaintext-Nachrichten über IMAP zu erhalten und zu senden. Wie folgt lassen sich die Einstellungen für einen E-Mail-Account in GoldBug eingeben.

**Detaillierte Beschreibung der POP3/IMAP Einrichtungsoptionen:**

Über das Haupt Menü “View/Ansicht” des GoldBug Messenger werden die eigene E-Mail-Adresse und die POP-3 bzw. IMAP-Server Details hinterlegt. Es sind dieselben Angaben, die auch z.B. im Thunderbird E-Mail-Klienten oder Outlook eingegeben werden, beispielsweise:

- ’”Incomming Server Server:”’ pop.gmail.com
- Port: 995
- TLS
- Username: mygmailname@gmail.com
- Password: ****

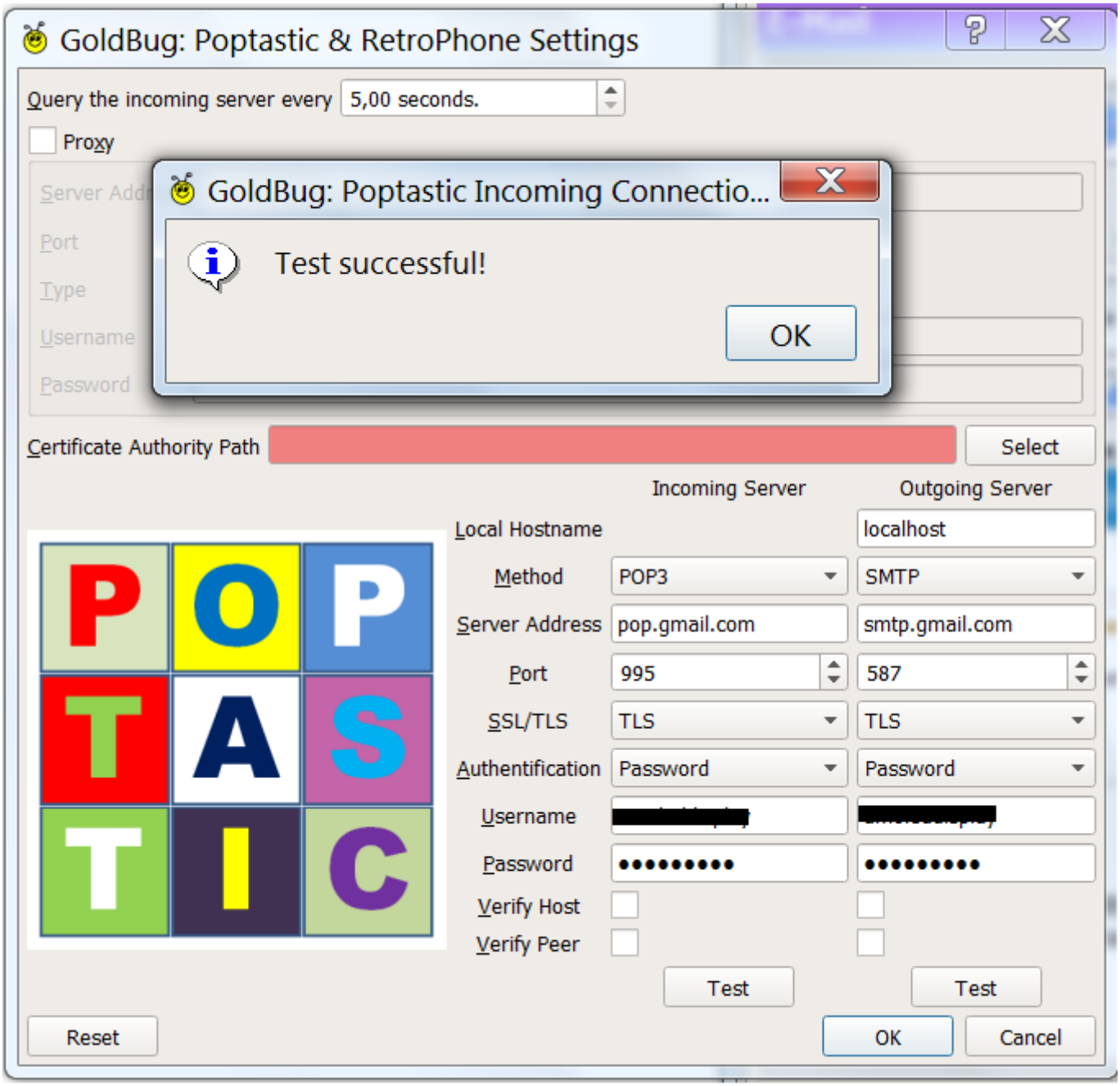
- ’”Outgoing Server Server:”’ smtp.gmail.com
- Port: 587
- TLS
- Username: mygmailname@gmail.com
- Password: ****

Der Nutzer kann jeweils den Test-Knopf drücken, um die Funktionalität der Server-Eingaben zu überprüfen. Mit dem “OK”-Knopf werden dann die Eingaben gespeichert.

(Wenn im Auswahlmenü statt POP3 oder IMAP der Wert “Disabled” genutzt wird, sendet das Programm keine E-Mails mehr: Die Mail-Funktion ist komplett abgeschaltet.)

Nutzer, die die weiter unten beschriebene Funktion des Chats über den POP3/IMAP-E-Mail-Server nutzen wollen (POPTASTIC), sollten daher die Mail-Angaben nicht deaktivieren.

Abbildung: POPTASTIC: Chat über E-Mail-Server




Nach obigen Sicherheitsüberlegungen sollte ein Nutzer seine E-Mails immer gleich vom Server auf seine eigene Maschine laden und sie auf dem Server löschen. Somit scheint vieles für die Nutzung von POP3 anstelle vom IMAP zu sprechen, da IMAP mehr auf die Beibehaltung der E-Mails auf dem Server ausgerichtet ist.

Generell gehören E-Mails in diesem Lichte also nicht auf einen Remote-Server, nicht in die Cloud, nicht in einen browser-basierten Web-Service - sondern sie gehören auf die eigene Maschine des Nutzers gespeichert - oder aber sie sind in jedem Falle zu verschlüsseln.

Doch der Trend ist heute genau anders: Zentrale Server, die ohne Verschlüsselung die Nachrichten speichern ohne eigene Infrastruktur in Nutzerhand. Bis der Trend sich wieder umkehrt und Nutzer die eigene Souveränität wieder entdecken werden.

## 9.3 P2P E-Mail: ohne Vorratsdatenspeicherung

Neben IMAP und POP3 besteht drittens in GoldBug die Option der Nutzung von p2p E-Mail. Damit ist gemeint, dass die E-Mails nicht in einem zentralen Server gespeichert werden, sondern im Klienten eines Freundes.

Hinsichtlich der Verschlüsselung wurde schon dargestellt, dass die E-Mail-Funktion einen anderen Schlüssel zur Verschlüsselung nutzt als die Chat-Funktion. So kann der Nutzer also einen Freund zum Chat hinzufügen, ihm aber das E-Mail verweigern oder umgekehrt. Sinnvoll ist es jedoch, immer alle Schlüssel als Ganzes auszukopieren, dann hat der Nutzer seinen Freund auch in allen Funktionen präsent (weiterhin also noch der URL-Schlüssel, POPTASTIC-Schlüssel und der Rosetta-Schlüssel - drei Funktionen, die später noch beschrieben werden).

Natürlich kann bei der Schlüsselübertragung auch für die E-Mail-Funktion wieder die Sicherheit eines Repleos genutzt werden, wenn man seinen eigenen öffentlichen E-Mail-Schlüssel nicht der Öffentlichkeit preisgeben will.

Egal welche E-Mail-Methode ein Nutzer wählt, ob POP3, IMAP oder P2P, ausgehende E-Mails in GoldBug sind immer verschlüsselt, es gibt nur eine Ausnahme, das ist, wenn der Nutzer im Key-Hinzufügen Tab nicht einen Key (oder ein Repleo) eingibt, sondern die Auswahl: E-Mail-Adresse hinzufügen eingibt. Dann sendet das E-Mail Programm unverschlüsselten Text von @-mail zu @-mail.

**Hinweis:** Wer einen POPTASTIC-Schlüssel eingibt, sieht auch die @-E-Mail-Adresse in der Kontaktliste für E-Mail, sie ist jedoch farblich hinterlegt und hat auch ein Schloss-Symbol, d.h. über POPTASTIC-E-Mail-Adressen wird ebenso nur verschlüsselt gemailt – und zwar der Chat. Denn schließlich wird bei POPTASTIC ein Schlüssel eingefügt und keine @-E-Mail-Adresse. Nur E-Mails an @-E-Mail-Adressen, die kein Lock-Symbol haben, bleiben unverschlüsselt.

Zur Verdeutlichung nochmal:

- Der Nutzer kann beim E-Mail folgende Wege nutzen
- Der E-Mail-Schlüssel: Dieser kann E-Mails über POP3,IMAP und P2P senden.
- Der POPTASTIC-SCHLÜSSEL: Dieser kann Chat über POP3 und IMAP senden
- Die @-Mail-Adresse: Diese kann unverschlüsselte E-Mails and normale @-Mailadressen senden über POP3 und IMAP.

Daher können zwei GoldBug-Nutzer mit normalem @Mail z.B. über die großen E-Mail-Provider wie Ymail, Gmail, GMX, Outlook etc. verschlüsselte E-Mails ohne weitere technische Kenntnisse austauschen: Entweder über @-Mail-Adressen unverschlüsselt, oder verschlüsselt als Chat über den POPTASTIC-Schlüssel, der noch weiter unten erläutert wird. Und drittens kann über den E-Mail-Schlüssel immer verschlüsselt gemailt werden, auch p2p.

Dieses ist insofern sehr komfortabel, als dass es ausreicht, einmal die Schlüssel zu tauschen. Es ist also nicht jedes einzelne Mail, welches der Nutzer schreibt, jedesmal wieder aufs Neue einzeln zu verschlüsseln (wie bislang in anderen Verfahren geübte Praxis). Jeder @-Mail Anbieter kann nun von der Einsichtnahme in die E-Mails des Nutzers ausgenommen werden, indem der Nutzer einfach verschlüsselten Ciphertext über die zentralen Server an seinen

Kommunikationspartner schiebt. Was lediglich benötigt wird, ist die Absprache mit dem Freund, dass dieser ebenso GoldBug oder einen der anderen Echo-Klienten als E-Mail-Klient nutzt, um einmalig die Schlüssel tauschen zu können.

E-Mail-Anlagen können ebenso einer E-Mail als Datei angehangen werden und werden unabhängig, welche verschlüsselnde E-Mail-Methode gewählt wird, automatisch ebenso verschlüsselt. Dieses ist auch mit mehreren Anhängen möglich.

Neben der Verschlüsselung der E-Mails werden in vielen Ländern weiterhin auch die Meta-Daten gespeichert, also wann und wie oft ein Nutzer die Nachrichten aus seinem Postfach abruft. Hier wird nun die weitere Methode der p2p E-Mails interessant:

GoldBug ermöglicht weiterhin, E-Mails auch im Teilnehmernetzwerk oder auf einem eigenen Server zu speichern und entsprechende E-Mail-Postfächer dezentral einzurichten, die darüber hinaus auch ausschließlich und automatisch den Standard der verschlüsselten E-Mails handhaben.

Der E-Mail-Klient enthält somit auch eine peer-to-peer basierte Komponente, d.h. die E-Mails werden über das Netzwerk der verschlüsselten Verbindungen gesandt und in den Knotenpunkten von Freunden zwischengespeichert. Dieses Netzwerk wird durch die integrierte Architektur des Spot-on-Kernels bereitgestellt. Der Vorteil von P2P-E-Mail liegt darin, dass das E-Mail-Postfach nicht bei einem zentralen Hoster und öffentlichen Anbieter von E-Mail liegt, sondern im dezentralen Netzwerk von eigenen Freunden des Nutzers eingerichtet werden kann.

Mit GoldBug ist also jeder in der Lage, für seine Freunde unkompliziert ein E-Mail-Postfach einzurichten. Niemand kann dann mit-loggen, wann und wie oft ein Nutzer seine E-Mails abruft. Das Echo-Protokoll trägt ebenso dazu bei, dass kaum Metadaten anfallen, die offenbaren, wer welche E-Mail gelesen hat und wer für wen eine E-Mail zwischenspeichert (da die Öffnung der verschlüsselten Nachrichten ausschließlich auf der Maschine des Nutzers erfolgt und jeder – gemäß dem Echo Protokoll - jede Nachricht an jeden sendet).

Wie es geht, ein Postfach für seine Freunde einzurichten, zeigt der folgende Abschnitt:

## 9.4 C/O & E-Mail-Institutionen einrichten

Das interessante an der GoldBug E-Mail-Funktion – und hier unterscheidet es sich ggf. von anderen p2p E-Mail Implementierungen - ist, dass es möglich ist, E-Mail auch zu Freunden zu senden, die offline sind.

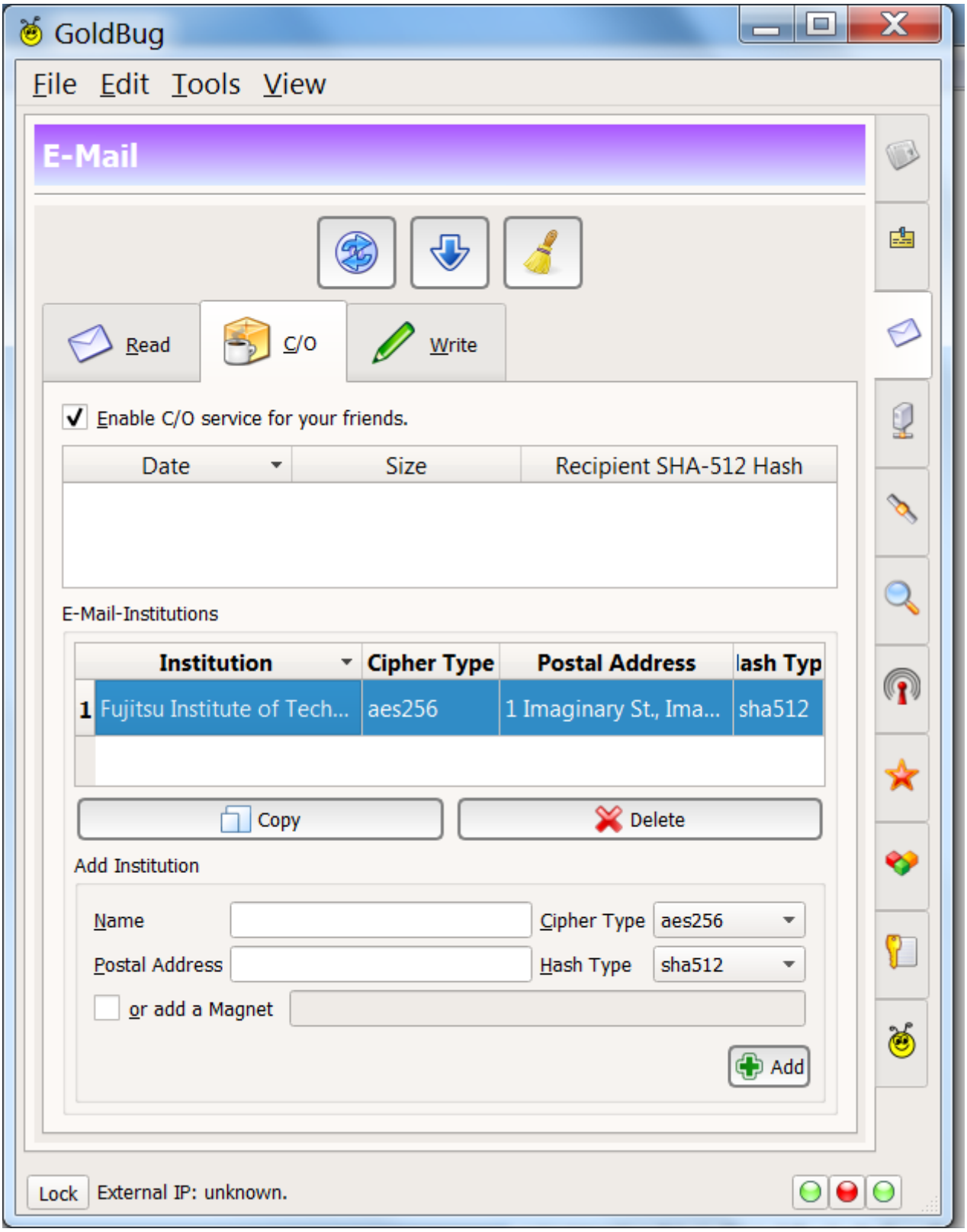
Hierzu bestehen zwei verschiedene Methoden:

### 9.4.1 Care-Of-Methode (c/o)

Die eine Methode ist, dass ein dritter, gemeinsamer Freund genutzt wird, um die E-Mails dort bei ihm zwischen zu speichern. Wenn Alice und Bob also einen gemeinsamen Chat-Server im

Web auf ihrem Webserver einrichten, und alle drei Ihre Schlüssel getauscht haben, fungiert der Webserver wie ein E-Mail-Postfach, wie wir es von POP3 oder IMAP her kennen.

Abbildung: P2P-E-Mail aus der Postbox bei einem Freund: c/o-Funktion

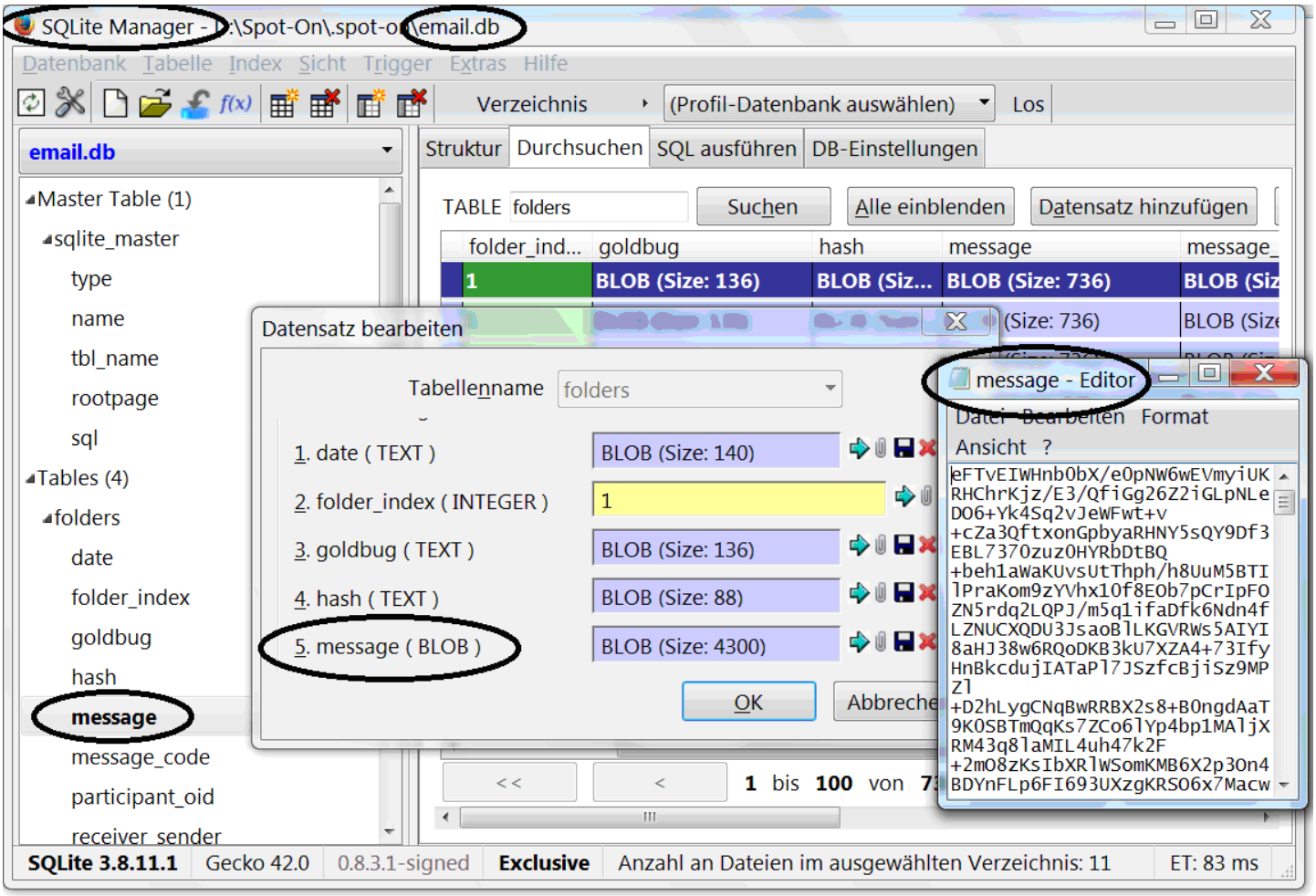



Grundsätzlich benötigen die E-Mails jedoch keine zentralen Server, es kann auch ein dritter Freund oder ein kleiner Raspberry-Pi-Computer zuhause sein, der kontinuierlich online bleibt. Es macht daher Sinn, mehr als einen Freund in seiner Liste zu haben und gemeinsame Freunde mit anderen Freunden zu vernetzen, die als Zwischenspeicher fungieren können. Da alle E-Mails

verschlüsselt sind, können die Freunde, die eine Cache-Funktion zur Verfügung stellen, die E-Mail des Nutzers auch nicht lesen.

Auch werden die E-Mails in verschlüsselten Datenbanken abgelegt. Die Abbildung zeigt, dass selbst mit Einsichtnahmen in die Struktur der Datenbank-Datei nur Ciphertext angezeigt wird.

Abbildung: Datenbank Verschlüsselung




Um diese Zustellanweisung Care-Of (c/o) Caching-Funktion zu aktivieren, muss in dem Sub-Tabulator “E-Mail-Settings” die Check-Box “Care-Of” aktiviert sein, wenn man als dritter Freund zwei anderen Freunden das Zwischenspeichern der E-Mails im eigenen Klienten ermöglichen will und sie beide auch in der E-Mail-Kontaktliste einfügt.

Der Nutzer hat bei GoldBug auch im p2p-E-Mail die Wahl, ob die E-Mails authentifiziert oder nicht authentifiziert gesandt werden, also einfach nur verschlüsselt gesandt werden, ohne Nachweis, dass der Schlüssel auch zu einem, bestimmten Nutzer gehört.

Die Care-of-p2p-E-Mail-Funktion ist eine der einfachsten in der Softwarelandschaft für p2p-E-Mail überhaupt. Wenn drei Nutzer einen gemeinsamen Echo-Server nutzen und sich gegenseitig gespeichert haben, muss nur die C/O-Funktion aktiviert werden, und die E-Mails werden in den Klienten der Freunde zwischen gespeichert, für den Fall, dass man mal offline ist. Einfacher als diese Architektur geht es nicht, man benötigt lediglich ein paar Freunde, die an diesem Verfahren für die interne Kommunikation innerhalb einer Gruppe teilnehmen möchten.

Die zweite Methode ist die Einrichtung einer virtuellen E-Mail Institution. Dieses ist ideal für Leute, die gleich eine ganze Community an Freunden mit einem E-Mail-Postfach ausrüsten wolen.

### 9.4.2 Virtuelle E-Mail Institution (“VEMI”) - Methode

Hierzu ist es ebenso notwendig, die C/O-Funktion mit der Check-Box wie oben geschildert zu aktivieren.

Sodann kann der Nutzer eine virtuelle E-Mail-Institution erstellen.

Für die Eingabe- bzw. Definitions-Felder “Name” und “Adresse” der Institution kann der Nutzer frei kreativ werden und beliebige Bezeichnungen wählen. Sodann sind noch die öffentlichen E-Mail-Schlüssel der Freunde, die in dieser Institution speichern wollen, in diesen Knotenpunkt einzukopieren.

Schließlich kann der Nutzer dann den so erstellten Magnet-URI-Link auskopieren und Freunden zur Verfügung zu stellen, die in diesem Postfach dann zwischen-speichern. (Zum Magnet-URI-Standard und was das ist, vgl. auch unten im Abschnitt Dateiübertragung mit “StarBeam”). Zusätzlich muss der Knotenpunkt, der die E-Mail-Institution einrichtet, immer auch den öffentlichen E-Mail-Schlüssel desjenigen hinzufügen, für den er speichern soll.

Der Vorteil gegenüber der ersten Methode ist somit, dass der öffentliche E-Mail-Schlüssel des Knotenpunktes, der die Institution einrichtet, niemandem bekannt gegeben werden muss. Bei der c/o-Methode ist der öffentliche E-Mail-Schlüssel hingegen auszutauschen. Daher kann man vereinfacht sagen, dass im kleinen Freundesnetzwerk ein gemeinsamer Knotenpunkt mit der c/o-Funktion ideal ist und die VEMI-Methode der Einrichtung von Virtuellen-E-Mail-Institutionen eher auf Anbieter fokussiert, die für eine größere Teilnehmeranzahl Postfächer einrichten wollen.

**Einrichtungsbeispiel:**

Hier wird nochmal anhand eines Beispiels beschrieben, wie bei den E-Mails die c/o-Funktion und die VEMI-Funktion, also die Einrichtung einer Virtuellen-E-Mail-Institution, Schritt für Schritt umgesetzt werden:

- Der Nutzer aktiviert die c/o-Funktion in dem Tabulator für E-Mail-Settings.
- Der Nutzer erstellt eine Institution und wählt einen Namen und eine Adresse für die Institution.
- Beispiel: Name= “GB-Postfach“ und Adresse = „Dotcom“
- Der Nutzer fügt den E-Mail-Schlüssel eines Freundes in seinen Klienten ein. Sodann kopiert er dann den verfügbaren E-Mail-Magneten von seiner E-Mail-Institution aus und lässt die Freunde diesen in ihr Programm einfügen. Der Magnet wird ähnlich wie dieser aussehen:

magnet:?in=GB-Postfach&ct=aes256&pa=Dotcom&ht=sha512&xt=urn:institution

Der Nutzer erkennt einen E-Mail-Magnet an seiner Endung: URN=institution. Dann weiß man, dass der Magnet kein Buzz-Gruppen-Chat Magnet ist und auch kein StarBeam-Magnet für den Dateiaustausch – denn diese hätten die Endung „URN=buzz“ bzw. „URN=starbeam“. So dann wird der eigene Knotenpunkt die E-Mails der Freunde in der eingerichteten Institution zwischenspeichern – auch für Adressaten, die ggf. offline sein sollten.

Der Nutzer (als Ersteller einer E-Mail-Institution) braucht dann wie genannt seinen eigenen E-Mail-Schlüssel nicht mit den Freunden bzw. “Subscribern” seiner Institution zu tauschen. Der Nutzer kann die E-Mail Schlüssel der Freunde auch in einem Gruppen-Chat-Raum über e’IRC/Buzz mit dem Ersteller einer E-Mail-Institution tauschen. Der Austausch-Prozess von Key & E-Mail-Magnet muss also keine weiteren Identitäten vermitteln.

## 9.5 Zusätzliche Verschlüsselung: Ein „GoldBug“ auf ein E-Mail setzen

Gleichgültig, welche Übertragungsmethode ein Nutzer wählt, ob POP3, IMAP oder P2P, die E-Mails sind über den öffentlichen (asymmetrischen) E-Mail-Schlüssel sowie das Echo-Protokoll für die Übertragung ja immer verschlüsselt.

Dieses ist auch der Fall, wenn sie in einer Zwischenstation wie einem Provider-Postfach oder einer Virtuellen Institution oder einem Zwischenknoten eines Freundes zwischengespeichert werden. Transportverschlüsselung und Ende-zu-Ende-Verschlüsslung gehen hier durchgängig ineinander auf.

Als zusätzliche Sicherheit für die E-Mail-Funktion besteht ähnlich wie im Chat beim sog. **“Gemini”**, nun für E-Mails die Option, ein Passwort auf die E-Mail zu setzen: Nicht nur die Software nennt sich GoldBug, sondern auch die Funktion im E-Mail-Klienten, auf das E-Mail ein zusätzliches Passwort zu setzen, nennt sich **“GoldBug”**.

E-Mails, auf die ein „GoldBug“-Passwort gesetzt wurde (vergleiche später unten die Beschreibung der File-Transferfunktion „StarBeam“, hier heißt das zusätzliche Passwort „Nova“) können vom Empfänger nur gelesen werden, wenn sie das entsprechende “GoldBug” – also den Goldenen Schlüssel für das Passwort kennen. Der Nutzer sollte daher seine Freunde über das einzugebende Passwort informieren, wenn der Nutzer ihnen E-Mails sendet, die zur Öffnung noch ein zusätzliches Passwort benötigen.

Das kann z.B. in den E-Mails zu der eigenen Frau sein, dass der Nutzer die E-Mails immer mit dem Namen der Stadt zusätzlich verschlüsselt, in der die Hochzeit oder der Hochzeitsurlaub stattfand.

Auch hier wie beim Chat mit dem Gemini ist es ein wichtiges Merkmal der symmetrischen und durchgängigen Ende-zu-Ende-Verschlüsselung, dass der Nutzer das Ende-zu-Ende verschlüsselnde Passwort selbst und manuell erstellen kann.

Neben der Erinnerung an die Kurzgeschichte von Edgar Allen Poe über ein Kryptogramm und sein Wirken für die Kryptographie schon in frühen Jahren der beginnenden Industrialisierung –

ist das GoldBug auf einem E-Mail eine neue Idee, E-Mail nicht nur automatisiert durch Schlüsseltausch immer a-symmetrisch verschlüsselt zu haben, sondern zusätzlich mit einer symmetrischen Verschlüsselung – dem GoldBug auf einem E-Mail – auch noch weitergehend und pro einzelnem E-Mail zu verschlüsseln, wie es (andernorts durch PGP) bislang der Standard (aber symmetrisch) ist. Dieses erfolgt ohne dass zusätzliche Verschlüsselungssoftware als Plugin installiert werden muss. Das GoldBug auf einem E-Mail ist das symmetrische Sahnehäubchen zu der symmetrischen Verschlüsselung auf einem E-Mail durch PGP – denn die symmetrische Verschlüsselung eines E-Mails ist in GoldBug ja schon standardmäßig automatisiert integriert und mit dem E-Mail-Schlüssel immer zugegen.

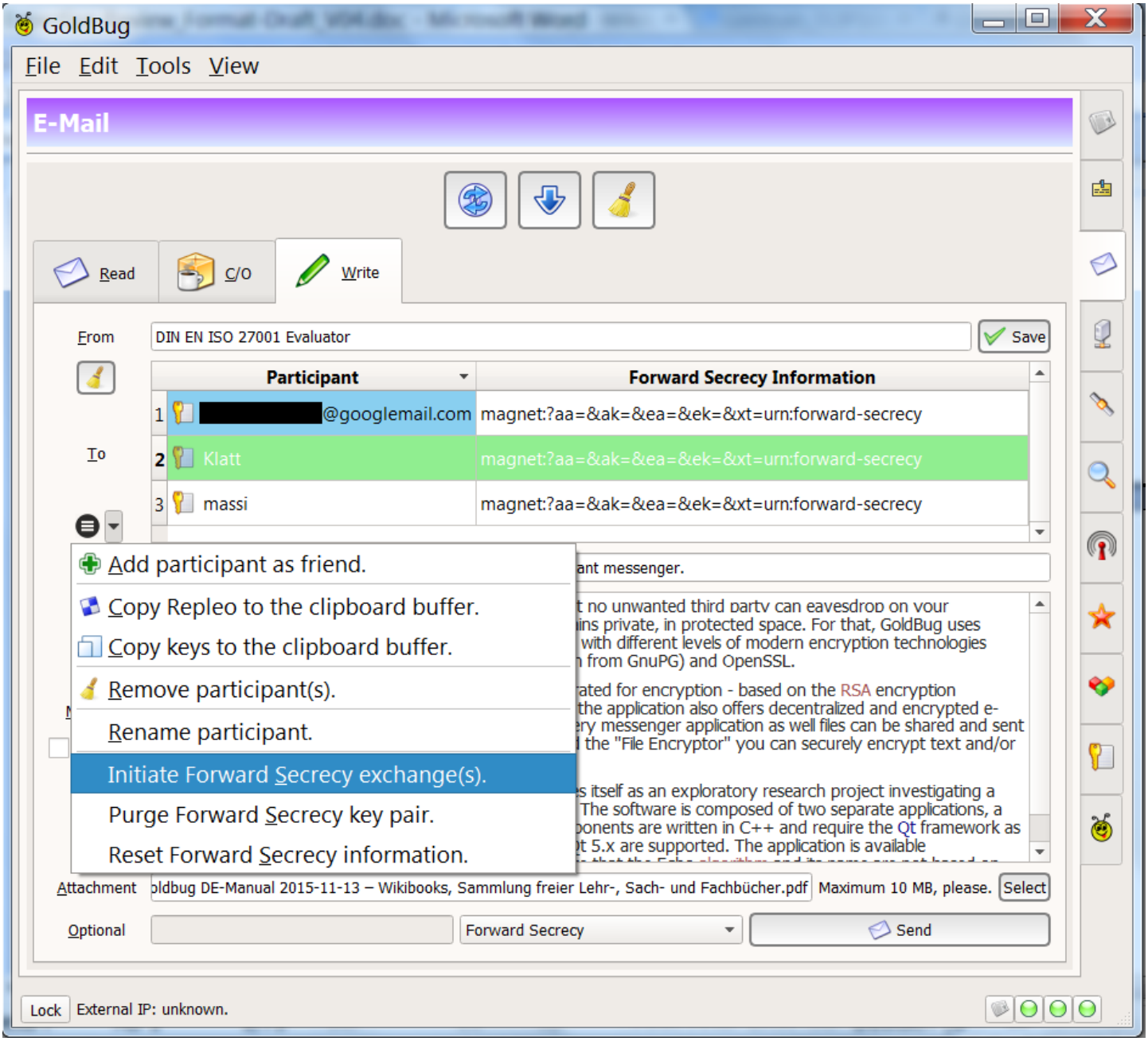
## 9.6 Forward Secrecy

GoldBug ist mit Nutzung der einbezogenen Architektur des Spot-on-Kernels weltweit das erste E-Mail-Programm, dass Forward Secrecy Verschlüsselung anbietet, die sowohl asymmetrisch **als auch** symmetrisch **für E-Mail** umgesetzt werden kann - also beide Verfahren innerhalb eines E-Mail-Programms unterstützt.

Forward Secrecy heißt ja, dass temporäre Schlüssel eingesetzt werden, um das Ende-zu-Ende verschlüsselnde Passwort zu übertragen, so dass, wenn zu einem späteren Zeitpunkt Analysen zur Kommunikation und deren Verschlüsselung gemacht werden sollten, nicht der reguläre (permanente) Schlüssel für die Kommunikation betroffen ist.

Der Nutzer sendet also seinem E-Mail-Partner über die übliche asymmetrische Verschlüsselung des E-Mail-Schlüssels nun einen sitzungsbasierten, symmetrischen (Forward Secrecy) Schlüssel (vgl. Abbildung).

Abbildung: E-Mail mit Forward Secrecy




Wenn der Partner die Anfrage bestätigt und seinen temporären Schlüssel zurücksendet, dann können beide Teilnehmer sitzungsbasierte asymmetrische Schlüssel nutzen, um die E-Mail-Kommunikation weitergehend zu sichern. Diese Methode der asymmetrischen Ende-zu-Ende Sicherung wurde im Übrigen nicht nur für E-Mail integriert, sondern auch auf die Chat-Funktion (siehe oben: Forward-Secrecy-(FS)-Calling).

Der permanente öffentliche Schlüssel wird dann nur für den Transport der sitzungsbasierten Schlüssel genutzt – und nicht mehr für den Transport der Nachricht (bzw. wird der neue Schlüssel zur Nachricht). Das bedeutet, der ephemerale (temporäre) Schlüssel wird über den permanenten öffentlichen Schlüssel mit dem Partner geteilt. Wenn dann der ephemerale, öffentliche Schlüssel von einem Empfänger korrekt akzeptiert wurde, generiert dieser besagte Empfänger ebenso einen ephemeralen Sitzungsschlüssel (symmetrisch), der dann über den öffentlichen Schlüssel des Nutzers ebenso dem Nutzer wieder zugesandt wird.

Der Initiator löscht sodann seine asymmetrischen ephemeralen Schlüssel, alsbald die temporäre Sitzung beendet wurde.

Abbildung: E-Mail Tab: Forward Secrecy im GoldBug E-Mail-Klient




Wenn ein Nutzer nun ein E-Mail schreibt, stehen in GoldBug also vier Forward-Secrecy-Modi zur Verfügung, das E-Mail zu verschlüsseln:

- Normal-verschlüsselt: Die E-Mail wird wie gehabt innerhalb des verschlüsselten Systems (Echo oder POPTASTIC) versandt, das heißt, der reguläre permanente symmetrische E-Mail-Schlüssel wird genutzt, um die Nachricht zu verschlüsseln.

- Forward Secrecy-verschlüsselt: Über die reguläre Verschlüsselung werden sitzungsbasierte Forward-Secrecy-Schlüssel genutzt - das bedeutet, der Nutzer sendet sitzungsbasierte Schlüssel über den Kanal des permanenten E-Mail-Schlüssels und sodann seine Nachricht verschlüsselt mit den temporären Schlüsseln. Das fügt also zu der Nachricht eine weitere asymmetrisch verschlüsselte Ebene zu der bereits bestehenden E-Mail Verschlüsselung hinzu.

- Pure Forward Secrecy-verschüsselt (“Pure FS”): Die Nachricht wird nur über den sitzungsbasierten (ephemeralen) E-Mail-Schlüssel des Nutzers verschlüsselt und gesandt. Der permanente E-Mail-Schlüssel kommt somit beim “Pure FS” nicht zum Einsatz: Dieses kann daher auch als die Option bezeichnet werden, innerhalb des p2p E-Mails “instant”, also sofortige (im Sinne von flüchtige) und einmalige E-Mail-Adressen und Post-Fächer zu erzeugen - die wieder nach der Sitzung gelöscht werden können. So entstehen One-Time-E-Mail-Accounts dank Pure Forward Secrecy.

- GoldBug verschlüsselt: GoldBug setzt wie oben beschrieben ein Password auf die E-Mail (z.B. mit einem AES, symmetrische Verschlüsselung) und der Nutzer muss seinen E-Mail-Partner über das Passwort idealerweise mündlich informieren. Die so ergänzend symmetrisch verschlüsselte Nachricht wird sodann ebenso über die asymmetrische E-Mail-Verschlüsselung gesandt (permanenter E-Mail-Schlüssel).

Für den Fall, dass der Nutzer die Check-Box-Option “Plain” neben dem E-Mail-Text markiert, wird das E-Mail nicht im HTML-Rich-Text-Modus geschrieben, sondern im reinen Text Modus. Das Word Plain hat hier also nichts im Sinne eines Antonym zu Ciphertext zu tun.

Nochmal ergänzend mit folgenden Augenmerk zum Verständnis: durch den permanenten (asymmetrischen) Schlüssel (für E-Mail (oder auch so bei Chat) werden ephemerale (ebenso asymmetrische Schlüssel) getauscht, die dann die Grundlage für die Nutzung von Ende-zu-Ende verschlüsselnden (symmetrischen) Schlüsseln dienen. Somit können die ephemeralen Schlüssel jederzeit nach Gebrauch wieder gelöscht werden und die Kommunikation ist nicht an die Identitäten gebunden.

Hier sollte man sich nicht verwirren lassen, denn auch die Ende-zu-Ende verschlüsselnden symmetrischen Passphrasen sind im eigentlichen Sinne ephemerale Schlüssel, aber es wird deutlicher, wenn nur die asymmetrischen temporären Schlüssel, die durch die permanenten

asymmetrischen E-Mail-Schlüssel geschoben werden, zunächst als ephemerale Schlüssel bezeichnet werden (damit es Personen, die sich das erste Mal mit dem Forward-Secrecy-Prozess bzw. der Vokabel "Ephemeraler Schlüssel" beschäftigen, nicht verwirrt sind).

Die Verschlüsselungsebenen bei Forward Secrecy im E-Mail-Program GoldBug können wie folgt vereinfacht beschrieben werden:

- Äußere Verschlüsselungsebene: SSL /TSL Verbindung
- Mögliche, weitere Verschlüsselungsebene: permanenter asymmetrischer E-Mail-Schlüssel (nicht bei "Pure FS" - ansonsten gilt: First-Ephemeral-then-Permanent)
- Weitere, später ggf. gelöschte Ebene: Ephemeraler, temporärer asymmetrischer Schlüssel (dient nur zur Übertragung der symmetrischen Schlüssel)
- erste Verschlüsselungs-Ebene über Forward Secrecy: Symmetrischer Schlüssel
- Alternative erste Verschlüsselungsebene über ein GoldBug: Symmetrischer Schlüssel über ein manuell definiertes GoldBug auf dem E-Mail. Das Nachrichtenformat ist also (TLS/SSL (Permanenter E-Mail-Key (AES-GoldBug (E-Mail-Nachricht))). Entsprechend Encrypt-then-Mac kann man hier von "GoldBug-then-Permanent" sprechen. Das GoldBug auf einem E-Mail verschlüsselt den Text im Briefumschlag.

Temporäre Schlüssel werden nicht aus permanenten Schlüsseln abgeleitet und stehen auch in der Generierung in keiner Beziehung zu diesen. Sitzungsperioden werden manuell vom Nutzer definiert. D.h. nicht wie bei anderen Programmen ist die Sitzung durch das Online-Kommen und wieder Offline-Gehen automatisch definiert, sondern der Nutzer selbst bestimmt, wann er neue sitzungsbasierte Schlüssel nutzen will. Auch dieses kann wieder jederzeit und "instant" sein. (Siehe oben: IPFS).

Der Prozess oder das Protokoll für Forward Secrecy kann wie folgt mit diesem Beispiel beschrieben werden:

1. Ich sende Dir meine Post-Adresse. Diese ist öffentlich.
2. Du sendest mir auch Deine Post-Adresse. Diese ist öffentlich.
3. Unsere Adressen sind permanent. Diese Adressen wechseln nur, wenn wir umziehen.

— Tage später —

1. Ich erstelle einen einzigartigen Umschlag, einen ephemeralen Umschlag.
2. Ich sende Dir, und nur Dir, meinen einzigartigen Umschlag. Ganz klar, ich nutze Deine Post-Adresse, um Dir diesen zu zusenden. Wir nehmen an, nur Du kannst das Geschriebene lesen. Ich könnte auch das Zugesandte mit meiner Unterschrift signieren.

— Am selben Tag —

1. Du erhältst meinen einzigartigen Umschlag und Du verifizierst dieses auch anhand meiner Unterschrift, wenn Du magst.
2. Du erstellst einen speziellen Brief.

3. Du bündelst den speziellen Brief in den einzigartigen Umschlag, den ich Dir gesandt hatte. Wenn Du ihn erst Mal versiegelt hast, kann nur ich ihn öffnen.
4. Du sendest den einzigartigen Umschlag wieder an meine Postadresse zurück. Optional kannst Du natürlich das erstellte Bündel auch wieder signieren.

— Immer noch derselbe Tag —

1. Ich erhalte Dein Bündel. Im diesem Bündel ist mein einzigartiger Umschlag.
2. In meinem einzigartigen Umschlag, den nur ich öffnen kann, ist Dein spezieller Brief.

3. Wir nutzen den speziellen Brief so oft wie wir wollen...Einmal, Zweimal. Etc.

Ein Satz an sitzungsbasierten Schlüsseln wird zurückgesandt über den ephemeralen Schlüssel. Das erste Bündel wird über die permanenten Schlüssel transportiert. Permanente Schüssel müssen nicht sein, aber sie bestehen nun mal (denn die SSL/TLS Verbindung ist ja auch noch da). Das bedeutet, der Nutzer sendet den ephemeralen Schlüssel (ein Weg) über den permanenten Schlüssel und der Partner sendet den Satz an sitzungsbasierten (symmetrischen Schlüsseln) über den ephemeralen Schlüssel zurück.

Am Ende - nachdem das Protokoll abgeschlossen ist - werden die ephemeralen Schlüssel gelöscht und nur der Satz an sitzungsbasierten Schlüsseln verbleibt.

## 9.7 Secret Streams

Forward Secrecy in einem E-Mail-Klienten sowohl asymmetrisch wie auch symmetrisch anzubieten, wurde oben bereits als innovativ beschrieben. Als noch innovativer kann weiterhin die neu und bislang einzigartig implementierte Funktion der **Secret Streams** gewürdigt werden: Secret Streams sind sozusagen eine Liste an temporären Schlüsseln, die durch das Passwort bei der SMP-Authentifizierung im Socialist-Millionaire-Protokoll erzeugt werden. Der SMP-Prozess ist oben im Chat-Abschnitt ausführlich beschrieben worden und kann auch für das Cryptographsiche Calling eingesetzt werden.

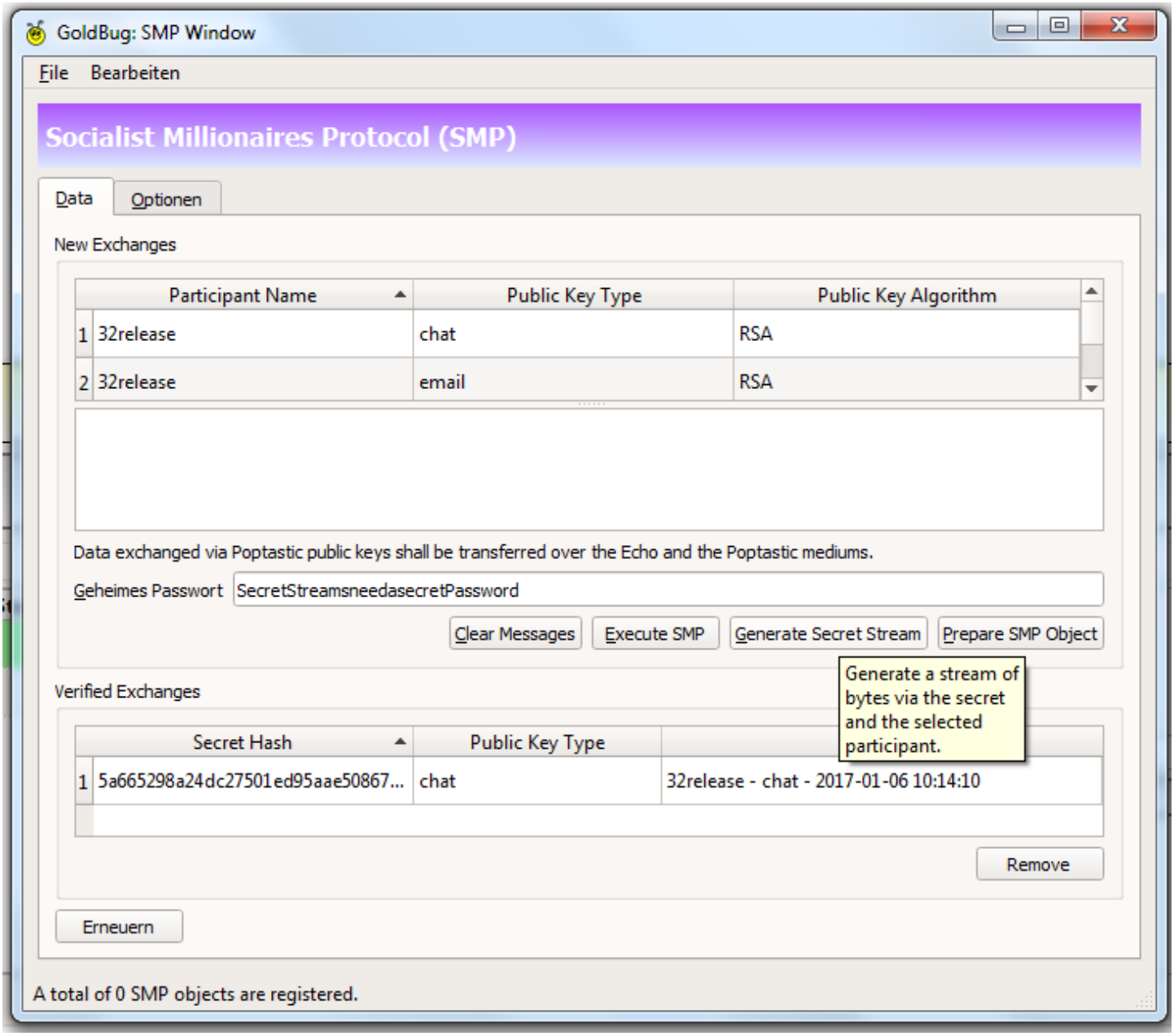
Und nun, diese atemberaubende neu Funktion der Secret Streams hat die Welt bislang so noch nicht gesehen und stellt Nutzer zufrieden und gewöhnt Begleiter in der Landschaft - wenn dieser Ausdruck gestattet sei -, denn es wird damit das Schlüssel-Übertragungs-Problem grundlegend gelöst: Beide Nutzer erhalten durch gegenseitige kontextuelle Hinweise oder ein nur beiden bekanntes Geheimnis ein nur ihnen bekanntes Passwort in den SMP-Prozess ein. Ist diese Authentifizierung erfolgt, können aus diesem Passwort ebenso zahlreiche temporäre, ephemerale Schlüssel abgeleitet werden, die in beiden Klienten gleich sind, ohne dass diese übertragen werden müssen - denn die SMP-Authentifizierung obliegt einem sog. Zero-Knowledge-Prozess.

Der Zweck des SMP Fesnters ist also, Key Streams aus einem Geheimnis zu erzeugen. Das Geheimnis wird über den SMP Prozess mathematisch verhandelt, ohne dass es als solches

übertragen wird. Somit werden die Schlüssel der Secret Stream Funktion abgeleitet aus einem Zero-Knowledge-Beweis!

Die Funktion der Secret Streams steht für Chat, E-Mail und auch POPTASTIC zur Verfügung: Temporäre Schlüssel, die nicht mehr übertagen werden müssen. Und eine kleine Revolution in der Kryptographie darstellen sollten, denn das Passwort-Übertragungsproblem wäre damit partiell gelöst. Es ist nur das SMP-Geheimnis erforderlich, das bislang der Authentifizierung und nicht der Verschlüsselung diente.

Abbildung: Implementierung von Secret Streams im extendierten SMP Protokoll




## Weitere Forschungsperspektiven

Es ist in Erinnerung zu behalten, dass man die permanenten Schlüssel in Transportschlüssel transformiert. Wenn diese kompromittiert werden, wird die Verschlüsselung mit den weiteren Verschlüsselungsinstanzen erkennbar. Dieses Konzept kreiert einen kreativen

Forschungsbereich innerhalb der Echo-Protokoll-Umgebung. Hier sind einige Konzepte-Anregungen, die weitergehend einbezogen werden könnten:

- Teilnehmer könnten konstant ephemerale (asymmetrische) Schlüsselpaare erzeugen und sitzungsbasierte (symmetrische) Schlüssel über die ephemeralen Schlüssel tauschen. Teilnehmer würden benachrichtigt, wenn nicht mehr genügend Schlüssel vorhanden wären. Der erneute Austausch (von ephemeralen Schlüsseln über den permanenten Schlüssel bzw. sitzungsbasierte (symmetrische) über die (sitzungsbasierten) ephemeralen (asymmetrischen) Schlüssel wäre dann automatisch geregelt... ähnlich des Austausches von Status-Nachrichten über den Online-Status im Chat. Nachrichten werden dann nur über sitzungsbasierte Schlüssel im Echo- oder POPTASTIC-Protokoll ausgetauscht.

- Anstelle des Austausches von einem Set an privaten Sitzungsschlüsseln könnten auch mehrere Sets an privaten Sitzungsschüsseln auf Vorrat ausgetauscht werden. Vorratsdatenspeicherung mal anders für eine Vielzahl an anonymen E-Mail-Adressen mit sitzungsbasierten Schlüsseln.

- Das OTR-Konzept (bislang für Chat) könnte innerhalb der permanenten Schlüssel und auch für E-Mail angewandt werden. POPTASTIC mal anders, wenn Chat über E-Mail geht, dann kann auch die Chat-Verschüsselung OTR über E-Mail gehen.

Durch die Nutzung von einzigartigen Schlüsseln können Informationsübertragungen in einer Sitzung ideal geschützt werden - selbst, wenn Versuche zu Kompromittierung bestehen sollten.

Forward Secrecy bietet also eine substantielle Verbesserung des Schutzes bei verschlüsselten Übertragungen für wenig Aufwand und keinerlei Kosten an.

Nachdem E-Mail und seine zahlreichen Optionen zur verbesserten und innovierten Verschlüsselung beschrieben wurden, kommen wir nun zu den schon angekündigten Begriff POPTASTIC – der Funktion von Chat über E-Mail-Server.

## 10 POPTASTIC - Verschlüsselter Chat und E-Mail über POP3 & IMAP

POPTASTIC ist eine Innovation in der Nachrichtenübermittlung - Verschlüsselter Chat über E-Mailserver.

Mit der POPTASTIC-Funktion können alle E-Mail Konten z.B. vom Gmail, Outlook.com oder Yahoo!-Mail mit GoldBug Ende-zu-Ende asymmetrisch - und hybride ergänzend symmetrisch - verschlüsselt werden. Der Clou: Jeder POP3 oder IMAP Server kann nun auch für verschlüsselten Chat genutzt werden. Und das auch durch Firewalls hindurch, wenn E-Mail gegeben ist.

Abbildung: POPTASTIC




Schauen wir uns POPTASTIC hier an dem Desktop-Klienten GoldBug einmal in seien Funktionen näher an.

## 10.1 Chat über POPTASTIC

Warum sollte man also noch einen dedizierten Chat-Server nutzen oder Chat-Protokolle mit Plugins für eine Verschlüsselung absichern, wenn man einfach seine E-Mail-Adresse für E-Mail und zugleich den Chat nutzen kann? Das mehrere Jahrzehnte alte POP3-Protokoll und zahlreiche E-Mail-Servern können nun für verschlüsselten Chat mit GoldBug genutzt werden. Der E-Mail-Server wird einfach als Chat-Server umfunktioniert.

Dazu wird die Chat-Nachricht in ein verschlüsseltes E-Mail gewandelt, über POP3 oder IMAP versandt, und vom Empfänger wird zurück in eine Chat Nachricht gewandelt. Da der GoldBug Messenger zugleich auch ein E-Mail-Client ist, funktioniert der verschlüsselte Nachrichtenaustausch auch über E-Mail. Das Programm erkennt automatisch, ob es sich um eine E-Mail über POP3 handelt oder um eine Chat-Nachricht.

Der Chat und das E-Mail über POPTASTIC sind proxy-fähig und können daher auch vom Arbeitsplatz, der Universität oder hinter einer Firewall betrieben werden, auch über das Netzwerk Tor. Logged man sich mit einem Web-Browser in seinen E-Mail-Account ein, sieht man, wie die verschlüsselte Nachricht aussieht.

Die ergänzend symmetrische Ende-zu-Ende Verschlüsselung über POP3 kann - wie beim Echo-Protokoll - nicht nur als Forward Secrecy eingesetzt werden, sondern kann ebenso “instant” jede Sekunde erneuert werden. Daher wird auch hier (wie oben) von Instant Perfect Forward Secrecy

(IPFS) gesprochen, das nun über POP3 und IMAP für den POPTASTIC-Chat ermöglicht wird! Schließlich besteht auch in POPTASTIC die Option, einen Call für die Übermittlung eines Geminis durchzuführen mit den oben differenziert geschilderten Methoden.

Diese Optionsvielfalt der Chat-Verschlüsselung bei POPTASTIC ist bislang auch nicht bei den Architektur-Derivaten für mobile Endgeräte gegeben.

Wie auch immer, für Anwender sicherlich eine interessante und einfache Methode, über dieses E-Mail-Protokoll verschlüsselt zu chatten.

## 10.2 E-Mail über POPTASTIC

Ebenso wie es E-Mail über den E-Mail-Schlüssel gibt, und wie es den Chat über den POPTASTIC-Schlüssel gibt, besteht auch die Möglichkeit über POPTASTIC zu e-mailen. Da POPTASTIC ein Schlüssel ist, mit dem der Freund eingegeben wird (über das Freund-Hinzufügen Tab), ist der POPTASTIC Kontakt bzw. die E-Mail-Adresse mit einem Schloss-Symbol versehen und zusätzlich auch mit einer Hintergrundfarbe markiert, um anzuzeigen, dass der Nachrichtenaustausch hier immer nur verschlüsselt geschieht.

Wenn man im Freund-Hinzufügen-Tab eine E-Mail-Adresse einfügt, wird dieser Kontakt ebenso in die Kontaktliste beim E-Mail hinzugefügt - jedoch ohne Locked-Symbol und ohne Hintergrundfarbe. Dieses zeigt an, dass hier mit diesem Kontakt die E-Mail-Nachrichten unverschlüsselt erfolgen. Dieses ist insofern der Fall, wenn jemand nicht den GoldBug-Klienten nutzt. Dann sendet POPTASTIC das E-Mail unverschlüsselt an die @-Mail-Adresse.

Das Programm weiss: wenn man vom POPTASTIC-Schlüssel an einen POPTASITC-Schlüssel mailt, dann ist dieses immer verschlüsselt und kann auch Chat sein. Und wenn man vom POPTASTIC-Schlüssel an eine @-Mail-Adresse mailt, dann ist die Nachricht unverschlüsselt. Dieses ist der einzige und seltene Fall, dass die Nachricht den Klienten unverschlüsselt verlässt, da hier nicht da Echo-Protokoll genutzt wird, sondern das reguläre E-Mail-Protokoll SMTP.

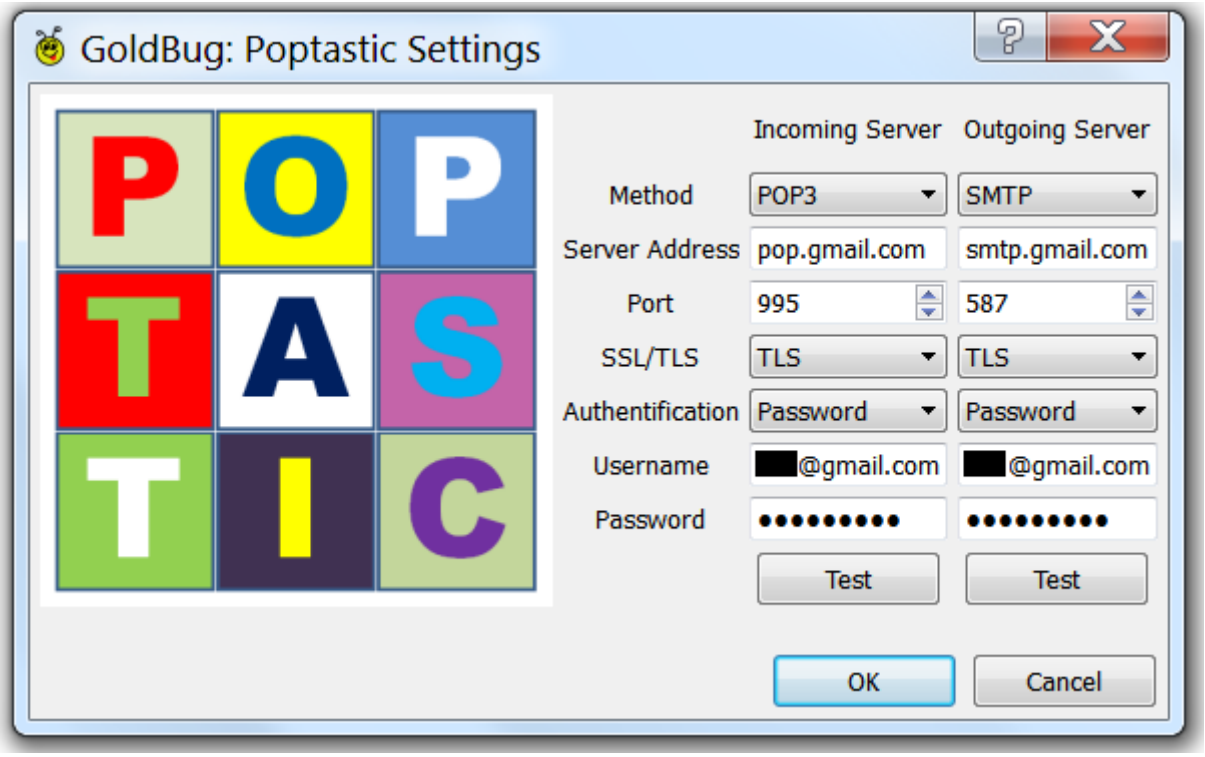
Wenn der Kontakt jedoch auch GoldBug nutzt, können beide dauerhaft verschlüsselt mailen, wenn der POPTASTIC-Schlüssel im Tabulator “Freund-Hinzufügen” eingegeben wird.

E-Mail über POPTASTIC ist also einfaches dauerhaftes verschlüsseltes e-mailen, indem einfach einmal zu Beginn der POPTASTIC-Schlüssel getauscht wird.

## 10.3 Einrichtung von POPTASTIC

Eine detaillierte Beschreibung der Einrichtungsoptionen des E-Mail-Servers findet sich weiter oben im Abschnitt zu POP3 und IMAP (vgl. auch Abbildung).

Abbildung: POPTASTIC Settings: Verschlüsselter Chat und verschlüsseltes E-Mail über für POP3 und IMAP




**Hinweis:** In Gmail sollte man ggf. im Web die Option setzen, dass abgerufene POP3-Nachrichten aus der INBOX gelöscht werden. Um verbinden zu können, empfiehlt es sich auch, die Sicherheitseinstellung in Gmail so zu definieren, dass mit allen lokalen E-Mail-Klienten zu Gmail verbunden werden kann (Funktion: GMail unbekannte Klienten zulassen):

- Settings / Forward and POP & IMAP / POP Download: Enable POP for all Mail
- Settings / Accounts & Import / Change Account Settings: Other Settings / [New window] / Security / Access for less secure/unknown Apps: Enabled.

Es empfiehlt sich, für einen ersten Test und die weitere Nutzung ggf. einen Extra-E-Mail Account einzurichten: Es ist ggf. zu beachten, dass neue E-Mail-Accounts z.B. bei Gmail ggf. für die ersten 30 Tage limitiert sind für die Aussendung von E-Mails (z.B. bei Gmail für 500 Chat-Messages oder E-Mails pro Tag). Das sollte für einen Test oder normalen Bedarf ggf. erst Mal ausreichen. Andernfalls kann man ja auch mit GoldBug seinen eigenen E-Mail-Server einrichten und ist auf die @-Mailkonten der großen Anbieter nicht mehr angewiesen, wenn es sich um kleine interne Nutzerkreise handelt, die das hier vorgestellte Echo-Mail intern mit eigenem Server nutzen wollen.

## 10.4 Weiterentwicklung von POPTASTIC

Diese Idee der Architektur ist vom Entwicklerteam von GoldBug entwickelt, veröffentlich und in der Studie Big 7 von den Auditoren des Programmes beschrieben worden. Sodann ist diese Idee von der mobilen Applikation Delta-Chat übernommen worden, wenn auch dort die Verschlüsselung über PGP und ausschließlich über IMAP-Server läuft. Die ursprünglichen Commits und Veröffentlichungsdaten zeigen den historischen Ursprung, zu dem es Referenzen und Credits herzustellen gilt, um nicht die Nähe einer Bedeutung der Plagiarisierung der Architektur dieser verschlüsselten Kommunikation zu suchen. (vgl. den Release von POPTASTIC

in 2014 und populär Mitte 2016 beschrieben sowie dessen Derivate mit ersten Commits einige Jahre später: https://sf.net/projects/goldbug/files/bigseven-crypto-audit.pdf (p134, 2016), https://sourceforge.net/p/goldbug/wiki/release-history/ (2014), https://delta.chat/en/blog (2016)).

Ein Fork und eine Weiterentwicklung der POPTASTIC Idee, die zu begrüßen (und zu referenzieren) ist in einem mobilen Chat-Klienten mit ansprechender Benutzeroberfläche.

## 11 FileSharing: mit StarBeam

Wie in jedem Messenger kann auch in GoldBug FileSharing zwischen mehreren Personen oder ein File-Transfer zwischen zwei definierten Freunden vertraulich und sicher verschlüsselt umgesetzt werden. Dieses geschieht im Tabulator “StarBeam”. Der Begriff StarBeam (SB) impliziert, dass FileSharing so einfach sei sollte, wie sich das Licht der Sterne durch die Galaxien projeziert oder “beamed”.

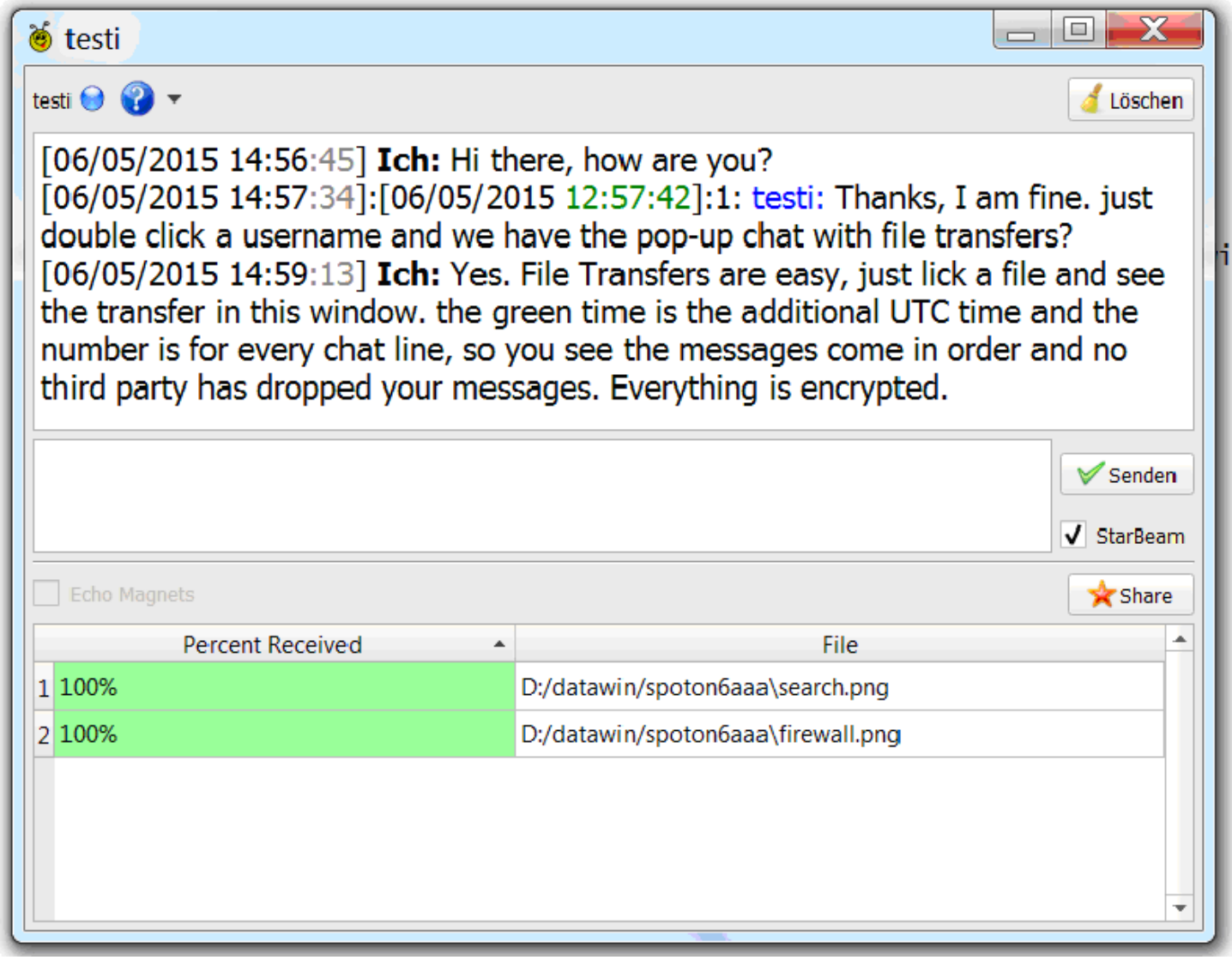
Während althergebrachte File-Sharing Programme wie EMule oder BitTorrent zunächst auf spezifische Links wie den ed2k-Link oder den Torrent-Link gesetzt haben, hat sich inzwischen für Datei-Übertragungen sowohl bei Torrents, also auch bei fast allen fortschrittlicheren Gnutella-Klienten sowie auch für das Edonkey-Netzwerk im Shareaza-Klienten die Verlinkung von Dateien über den Magnet-URI-Standard etabliert.

Die zu GoldBug und dem Spot-On-Kernel gehörige Elaboration hat die Architektur dieses Magnet-URI-Standards weiterentwickelt und um kryptographische Werte ergänzt.

Wenn der Nutzer nun eine Datei über GoldBug von anderen herunterladen will, muss er also einen Magnet-URI-Link in das Programm einkopieren. Und entsprechend: Wenn der Nutzer einen Upload einer Datei vorbereiten will, ist ein Magnet-URI für diese Datei zu erstellen.

Dieser Prozess ist als einfach als möglich gehalten: Wenn der Nutzer mit einem Freund in einem Pop-Up-Chat-Fenster chattet (vgl. Abbildung), ist dort ein Knopf “Share StarBeam”. Der Nutzer kann diesen einfach klicken, sodann die zu sendende Datei auswählen und schon wird sie sicher verschlüsselt über die Echo-Verbindung an den Freund übertragen.

Abbildung: GoldBug Messenger Chat Pop-Up Window mit File-Transfer



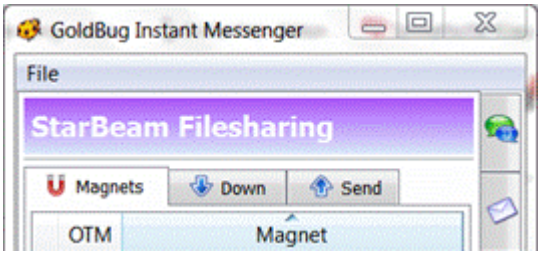

So kann der Nutzer ein ZIP mit Urlaubsbildern einfach und sicher zu Familienmitgliedern über den Chat oder das StarBeam-Tab übertragen.

Um einer ganzen Gruppe eine Datei zukommen zu lassen, kann der Nutzer seinen Magneten auch im Gruppenchat posten. Dieser wird dann automatisch den Downloads hinzugefügt (vgl. Check-Box im Menü Optionen: Buzz/e'IRC-Chat: akzeptiere geteilte Magneten).

Aufgrund des Echo-Protokolls werden die Einzelpakete auch "geswarmed", d.h. die verschlüsselten Pakete, die bei dem Nutzer vorbeikommen, kommen auch bei seinen Freunden und Nachbarn vorbei.

Der Tabulator "StarBeam" für das File Sharing besteht aus drei Sub-Tabulatoren: einen für das Hochladen, einen für das Herunterladen und einen für das Erstellen oder Hinzufügen von SB-Magneten.

Abbildung: StarBeam mit seinen drei Sub-Tabs




## 11.1 StarBeam-Magneten erstellen mit Verschlüsselungswerten

Ein Magnet-URI ist ein Standard, der aus vielen File-Sharing Programmen bekannt ist (vielfach im Gnutella Netzwerk) und ebenso eDonkey/Emule ed2k-Links oder auch Torrent-Links entspricht.

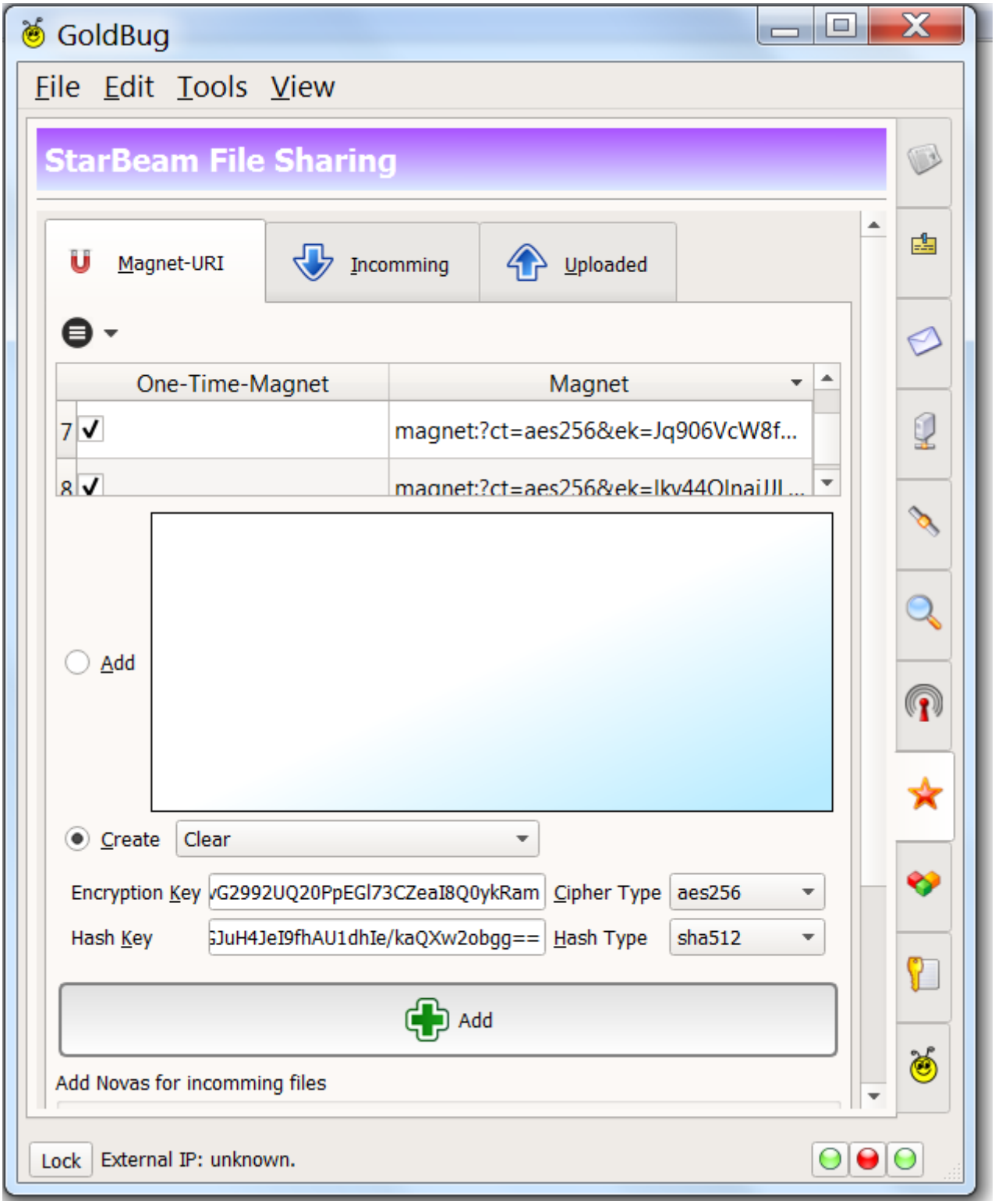
Die Weiterentwicklung des Magnet-URI Standards durch die dem GoldBug Messenger zugrunde liegende Spot-On Bibliothek liegt in der Ausgestaltung des Magnet-URI mit Verschlüsselungswerten. Magneten werden also genutzt, um ein Bündel an kryptologischen Informationen zu erstellen oder zusammen zu halten.

Zwischen den Knotenpunkten im Echo-Netzwerk wird somit ein Ende-zu-Ende verschlüsselter Kanal geschaffen, durch den eine Datei dann gesendet werden kann. Es kann aber auch jede andere Datei gesandt werden. Der Magnet ist somit keiner bestimmten Datei zugeordnet. Der SB-Magnet ist wie ein Channel zu verstehen, durch die eine Instanz laufend und dauerhaft Dateien senden kann - oder aber es wird ein One-Time-Magnet erstellt, der nach der einmaligen Nutzung gleich wieder gelöscht wird.

Durch diesen Dual-Use-Effekt kann ein Magnet nicht einer einzelnen Datei oder einer bestimmten IP-Adresse zugeordnet werden. Auch ein Dateiname taucht in dem SB-Magneten nicht auf (- wie es selbst bei den - gegenüber Gnutella, Emule und Torrent-Links - fortschrittlicheren Verlinkungen beispielsweise von OFFSystem oder RetroShare noch der Fall ist). Somit wird deutlich, dass in StarBeam keine bestimmte Datei getauscht wird, sondern es werden grundsätzlich nur verschlüsselte Kanäle getauscht. Sozusagen ein „Wurmloch“, um bei dem Begriff der „Stars“ zu bleiben. Und dieser Kanal wird eben durch einen Magnet-URI-Link und seine kryptologischen Werte definiert.

Während hingegen zahlreiche Meinungen das Verlinken von Gnutella, Edonkey und Torrent-Links im Web kritisch sehen, besteht bei einer Kollektion aus Verschlüsselungswerten sodann keine Veranlassung dazu, diese Werte zu hinterfragen. Eine Homepage bzw. unabhängige Portale finden mit StarBeam und den Magnet-URIs ein fortschrittliches Konzept vor. Neben den konzeptionellen Entscheidungen der Auswahl eines Link-Standards geht es bei dem Nutzungsaspekt jedoch auch um die Sicherheit des Dateitransfers zwischen zwei privaten Nutzern.

Abbildung: Magnet URI-Standard mit Crypto-Werten für den Dateitransfer




Zusammenfassend: Um eine Datei zu senden, muss also ein verschlüsselter Kanal erstellt werden. Dieses funktioniert mit der Erstellung eines Magneten - am Ende gekennzeichnet durch URN=StarBeam.

Sodann wird die Datei Paket für Paket über diesen Kanal verschlüsselt übertragen mittels des HTTPS-Protokolls (das auf TCP-, UDP- und auch SCTP-Verbindungen aufsetzen kann). Es ist daher eine interessante Fragestellung für einen Praxistest, ob ein Transfer einer großen, verschlüsselten Datei mittels StarBeam über das Echo-Protokoll basierend auf SCTP, TCP oder UDP Verbindungen ceteris paribus fehlerfreier und am schnellsten übertragen wird?

Für den Ablauf der privaten Datei-Übertragung von Freund zu Freund einige weitere Hinweise:

### 11.1.1 "'Option „Nova“: Datei vor dem Dateiversand verschlüsseln?'"

Bevor der Nutzer eine Datei versendet, kann er überlegen, ob er sie einfach per E-Mail an ein E-Mail innerhalb von GoldBug anhängt. Dieses ist die Variante der Wahl, wenn die Datei kleiner als 10 MB ist. Größeren Dateien sollten ausschließlich über die StarBeam-Funktion oder im Chat-Fenster an einen Freund übertragen werden.

Vor einem Versand kann der Nutzer auch überlegen, die Datei auf der Festplatte zu verschlüsseln. Dazu hält der GoldBug Messenger im Hauptmenü unter Werkzeuge/Tools ein Werkzeug für die Dateiverschlüsselung bereit (vergleiche Abschnitt unten: GoldBug-File-Encryptor). Mit einer doppelten Passphrase wird die Datei darin verschlüsselt.

Dieses Werkzeug des GoldBug-File-Encryptors kann man natürlich auch nutzen, wenn man eine Datei irgendwo in einen Online-Hoster der Cloud hochladen will oder sie über einen anderen Weg, einen anderen Messenger oder E-Mail übertragen möchte.

Da diese Online-Hoster wie Dropbox jedoch die Dateien ggf. kontrollieren und verschlüsselte Dateien mit einem Fragezeichen versehen werden, obwohl es ein Ausrufezeichen sein sollte, macht es Sinn, die verschlüsselte Datei gleich von Punkt zu Punkt, von Freund zu Freund über GoldBug zu transferieren und keinen externen bzw. fremden Zwischenspeicher als Host zu nutzen.

Manche packen die Dateien auch in ein Zip und verschlüsseln dieses vor dem Versand oder Upload. Die Zip-Verschlüsselung ist jedoch sehr leicht zu knacken mit 96 Bits, insofern sollte man also einen Schlüssel nutzen wie er heute für RSA mit mindestens 2048 Bits, besser 3072 Bits (wie für GoldBug als Standard definiert) für RSA empfohlen wird. Oder man nutzt gleich den noch sicheren Algorithmus McEliece anstelle von RSA.

Egal wie der Nutzer seine Datei nun auch vorbereitet - so wie sie ist: (1) als plain-Binärdatei, oder (2) verschlüsselt mit dem GoldBug-Tool über StarBeam oder (3) als Datei, die mit einem zusätzlichen Nova-Passwort (s.u.) im Star-Beam-Prozess geschützt ist - in jedem Fall wird sie ja wiederum mit dem Echo-Protokoll mehrfach verschlüsselt übertragen. Ein Optimum an Verschlüsselung und Optionsvielfalt das zahlreiche Wünsche abdeckt.

Genau wie man beim E-Mail ein zusätzliches Passwort auf ein E-Mail setzen kann („GoldBug“ bei der E-Mail-Funktion genannt, siehe oben) kann man auch auf die Datei – bzw. auf den verwendeten Magnet-URI zum Dateitransfer - ein weiteres Passwort setzen. Dieses wird „Nova“ genannt.

Selbst wenn der Dateitransfer erfolgreich geglückt ist oder auch ein dritter Unbekannter die bisherige Vielfach-Verschlüsselung knacken könnte (was erst mal nicht anzunehmen ist), wird mit dem Nova-Passwort eine Ende-zu-Ende Verschlüsselung eingeführt, die so lange sicher ist, wie das gemeinsame Passwort ausschließlich bei beiden Partnern unter Verschluss ist.

Denn, wenn die Übermittlung des SB-Magneten abgehört werden sollte – der Nutzer muss den Magneten ja irgendwie online an seinen Freund übertragen – dann kann jeder, der den

Magneten kennt, auch die Datei ebenso empfangen. Daher macht es Sinn, die Datei mit einem „Nova“ zu schützen – ein Passwort, das beide Freunde ggf. mündlich, in der Vergangenheit oder über einen zweiten Kanal getauscht haben.

Das Nova baut auch auf dem Ende-zu-Ende Verschlüsselungsstandard AES auf (wenn der Nutzer sich nicht eine eigene Passphrase ausdenkt).

Wie genannt, die Möglichkeit, eigene Ende-zu-Ende verschlüsselnde Passworte selbst zu erstellen und selbst manuell einzugeben – in der Fachwelt „Customer Supplied Encryption Keys“ (#CEKS) genannt – wird bislang nur von sehr wenigen Applikationen wie GoldBug Messenger oder Smoke Chat umgesetzt.

Und: Das Nova muss - bevor - der Dateitransfer beginnt, in dem Node des Empfängers hinterlegt worden sein!

### 11.1.2 Nutzung eines One-Time-Magneten

Idealerweise hat der Nutzer für jede Datei einen eigenen Magnet-URI. Das wäre sodann ein One-Time-Magnet (OTM), ein Magnet, der nur einmal genutzt wird für eine Datei. (OTM entspricht also dem Gedanken eines OTP - einem One-Time-Pad: eine Zeichenfolge, die nur einmalig genutzt wird. OTP wird oftmals in kryptologischen Prozessen als entscheidend angesehen, um Sicherheit herzustellen).

Der Nutzer kann einen Magneten auch dauerhaft nutzen, dann ist das wie ein abonnierter Video-Kanal, in dem z.B. jeden Montag eine Datei versandt wird.

Das eröffnet z.B. auch Torrent-Portalen ganz neue Möglichkeiten: es muss gar kein Web-Portal mehr existieren, in dem tausende an Links verlinkt sind! Das Portal selbst braucht nur einen einzigen Magneten im dezentralen Echo-Protokoll, um sodann konsekutiv, nach und nach, eine Datei nach der anderen durch das Wurmloch zu senden.

Alsbald der Nutzer eine Datei über den Magneten transferiert hat, kann der Nutzer den Magnet-URI also löschen oder beibehalten. Erstellt der Nutzer den Magneten als OTM und aktiviert die Check-Box für OTM, dann löscht er sich nach Dateitransfer von selbst. Das ist dann vergleichbar wie im Film Mission Impossible oder bei Apps für Bilder, wo sich Nachrichten und Bilder nach Ansicht selbst zerstören - der Magnet ist sozusagen ein StarBeam-Wurmloch, dass sich nach einmaliger Nutzung wieder schließt.

### 11.1.3 Übersicht der Magnet-URI-Standards für kryptographische Werte

Folgende Übersicht erläutert die gebräuchlichen kryptologischen Werte im Magnet-URI-Standard.

| Kürzel | Beispiel | Bezeichnung |
|---|---|---|

| Kürzel | Beispiel | Bezeichnung |
|---|---|---|
| rn | &rn=Spot-On_Developer_Channel_Key | Raumname |
| xf | &xf=10000 | Exact Frequency |
| xs | &xs=Spot-On_Developer_Channel_Salt | Exact Salt |
| ct | &ct=aes256 | Cipher Type |
| hk | &hk=Spot-On_Developer_Channel_Hash_Key | Hash Key |
| ht | &ht=sha512 | Hash Type |
| xt=urn:buzz | &xt=urn:buzz | Magnet zum IRC Chat |
| xt=urn:starbeam | &xt=urn:starbeam | Magnet zum Dateiversand |
| xt=urn:institution | &xt=urn:institution | Magnet zum E-Mail-Postfach |

Ein Magnet-URI, der in GoldBug, respektive für den Spot-On Entwickler Gruppen-Chat-Kanal genutzt wird, sieht z.B. wie in diesem Format aus:

magnet:?rn=Spot-On_Developer_Channel_Key&xf=10000&xs=Spot-On_Developer_Channel_Salt&ct=aes256&hk=Spot-On_Developer_Channel_Hash_Key&ht=sha512&xt=urn:buzz

Dieser Standard wird genutzt, um symmetrische Schlüssel zu tauschen für Gruppenchat oder E-Mail-Institutionen oder auch Dateiübertragungen mit StarBeam.

Der Magnet-URI-Standard wurde damit weiterentwickelt in ein Format, um Verschlüsselungswerte ähnlich einem Blatt mit Blutwerten weiter geben zu können.

Verschlüsslungen mit sehr individuellen DNA-Werte geben größtmögliche Sicherheit.

### 11.1.4 Rewind Funktion

Wenn ein Empfänger ein Datei-Packet, ein Chunk (oder im Spot-On-Kernel auch „Link“ genannt), empfangen hat, ist er in der Lage, dieses nochmal hochzuladen – auch in andere Magnet-URI-Kanäle – oder es nochmals in den gleichen Kanal zu geben. Dieses ist ähnlich einer Rewind-Funktion: Die Datei wird einfach nochmal wie auf einem Kassetten-Recorder oder MP3-Spieler über das Echo–Netzwerk erneut abgespielt. Die Datei kann auch viele Stunden oder Tage später nochmals versandt werden. Jeder, der eine Kopie über den Magnet-URI-Kanal erhalten hat, wird

zu einem Satelliten, und kann die Daten erneut in den definierten Kanal oder besser: über den oder einen StarBeam-Magneten wieder einspielen.

### 11.1.5 Vergleich mit Turtle-Hopping

Beim Turtle Hopping werden die Datei-Pakete von Freund zu Freund weitergeben, bis sie ein definiertes Ziel erreichen. Es ist eine Wandlung eines Peer-to-Peer-(P2P)-Netzwerkes in ein Friend-to-Friend-(F2F)-Netzwerk. Es hat jedoch den Nachteil, dass Freunde mit nur geringer Upload-Geschwindigkeit zum nächsten Freund in der Kette ein Nadelöhr bilden und den Transport verlangsamen:

Das Turtle-Hopping Protokoll ist erstens nur mit Knotenpunkten verbunden, die sich als Freunde definiert haben und in dieser Verkettung von Freunden kann ein Freund dabei sein, der nur eine geringe Bandbreite umsetzt. Dieser wirkt dann als Flaschenhals und Sender und Empfänger der Datei müssen zwingend durch diesen Flaschenhals.

Die Übertragung einer Datei in der StarBeam-Funktion über das Echo-Protokoll ist daher auch effektiver, als sie über ein Protokoll ähnlich dem „Turtle Hopping“ (derzeit nur im Programm RetroShare implementiert) laufen zu lassen, da hier je nach Ausgestaltung des Echo-Netzwerkes (volles Echo, halbes Echo, Adaptives Echo) und der grundsätzlichen Verschlüsselung Knotenpunkte mit nur geringer Bandbreite nicht als Flaschenhals wirken müssen, sondern über weitere Echo-Wege die gewünschten Download-Geschwindigkeit optimieren.

Beim Datei-Versand über das Echo-Protokoll können daher in den Zwischenstationen auch andere Knotenpunkte wie Peers oder Wege über andere Graphen-Optionen einbezogen werden, wenn irgendwo ein schnellerer Weg besteht:

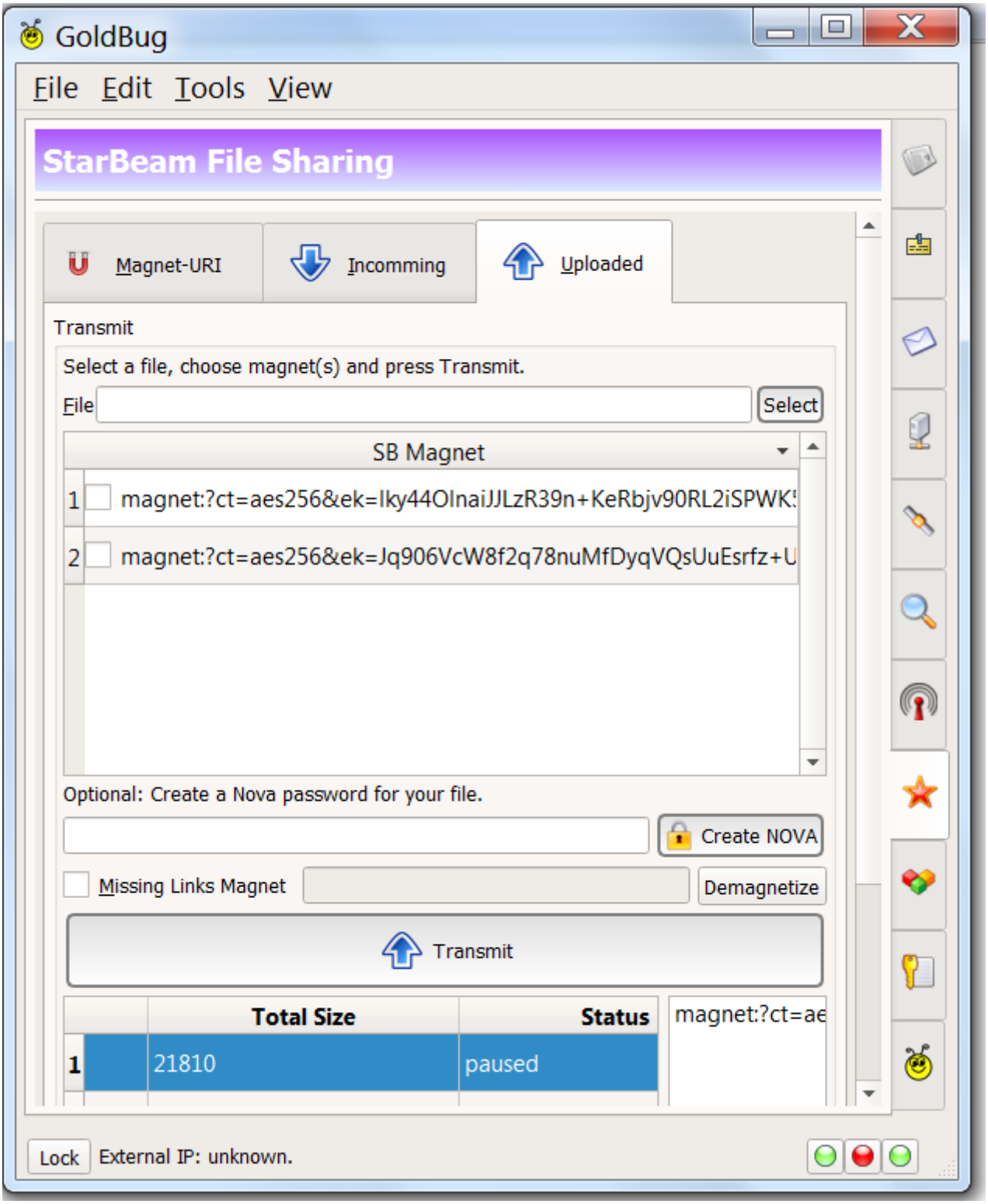
Das Echo-Protokoll erstellt den Fluss im Netzwerk der Knotenpunkte automatisch (einfach indem jeder Knotenpunkt zu jedem verbundenen Knotenpunkt Datei-Pakete verschlüsselt senden kann) und wählt daher auch den schnellsten Weg aller möglichen Graphen zu dem gewünschten Knotenpunkt.

## 11.2 StarBeam-Upload: Eine Datei übertragen

Wie oben beschrieben ist der Versand einer Datei aus dem Chat-Fenster an einen einzelnen Freund sehr einfach: Mit dem Share-StarBeam-Knopf ist nur eine Datei zu wählen und schon wird sie an den Freund übertragen.

Im Folgenden schauen wir uns den Upload-Prozess mit seinen technischen Details nun auch im Sub-Tabulator “Uploads” des StarBeam-Tabulators an.

Abbildung: Starbeam-Dateitransfer: Upload von Dateien




Wenn der Nutzer einen Magnet-URI definiert und generiert hat, erscheint er nicht nur im Sub-Tab für die Magneten, sondern auch in der Tabelle im Sub-Tab für den Upload/Seed.

Auch von hier aus kann der Upload einer Datei gestartet werden. Dazu wählt der Nutzer in diesem Sub-Tab für den Upload in der Check-Box einen SB-Magnet aus. Ebenso wird die Datei ausgewählt.

**Optional**: Ein Nova-Passwort für die zusätzliche Verschlüsselung der Datei: Schließlich kann sich der Nutzer noch entscheiden, ob er auf den Transfer ein zusätzliches Password – wie oben beschrieben „Nova“ genannt – setzen möchte. Die Datei kann der Freund also nur dann öffnen,

wenn er das Nova-Passwort eingibt. Es ist eine zusätzliche symmetrische Verschlüsselung, um einen Datei-Transfer abzusichern.

Sodann ist der Knopf „Transmit“/“Übertragen“ zu drücken.

(Technischer Hinweis: Da das Echo als HTTPS-Post oder HTTPS-Get übertragen wird, entspricht der Transfer dem einer Webseite. Die Chunk-Grösse (Pulse-Size) kann man so wie vor-definiert belassen, sie ist daher in der minimalen Ansicht der Benutzeroberfläche von GoldBug ausgeblendet. Wenn die Pulse-Size größer gemacht wird, wird die zu übertragende Webseite sozusagen länger.)

Schließlich kopiert der Nutzer den Magnet-URI aus und senden diesen seinem Freund. Den Magnet-URI kann man über den Kontext-Menü-Knopf auskopieren.

Wenn der Freund den Magneten einkopiert hat, kann der Nutzer den Transfer mit der Deaktivierung der Pause-Funktion starten (Check-Box “Pause” in der Tabelle).

Sodann wird die Datei an den Freund übertragen.

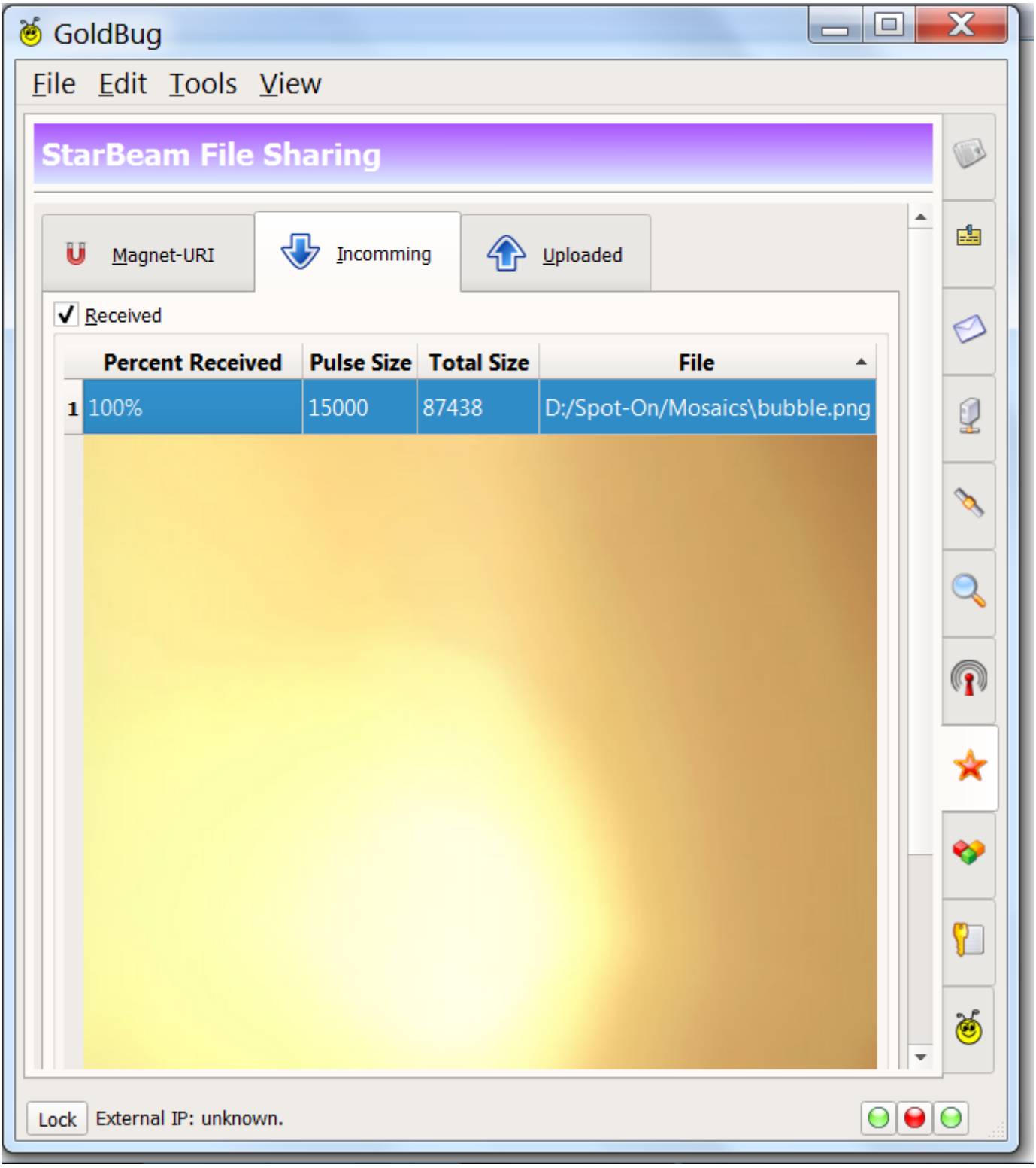
## 11.3 StarBeam-Downloads

Um mit StarBeam eine Datei zu laden, benötigt der Nutzer den StarBeam-Magnet der Datei. Diesen erhält der Nutzer von seinem Freund, der eine Datei senden will.

Der Nutzer kopiert sodann den Magnet-URI in den Sub-Tab für die Magnet-URIs einfach ein. Zuvor sollte der Nutzer im Download-Sub-Tab noch die Check-Box „Receiving“ / „Empfang“ aktivieren. Diese ist per default vorab deaktiviert, damit keine ungewollten Dateien eingehen.

Sodann teilt der Nutzer seinem Freund mit, dass er den Magnet-URI eingefügt hat und dann kann der Freund die Übertragung starten. Der Download startet, alsbald ein Sender über das Echo und durch den Crypto-Kanal des Magneten die Datei sendet.

Abbildung: StarBeam-Dateitransfer - Eingehende Dateien




Mit den weiteren Einstellungen auf dieser Tabulator-Seite für den Upload kann der Nutzer noch die Größe und den Pfad für den Download-Bereich definieren.

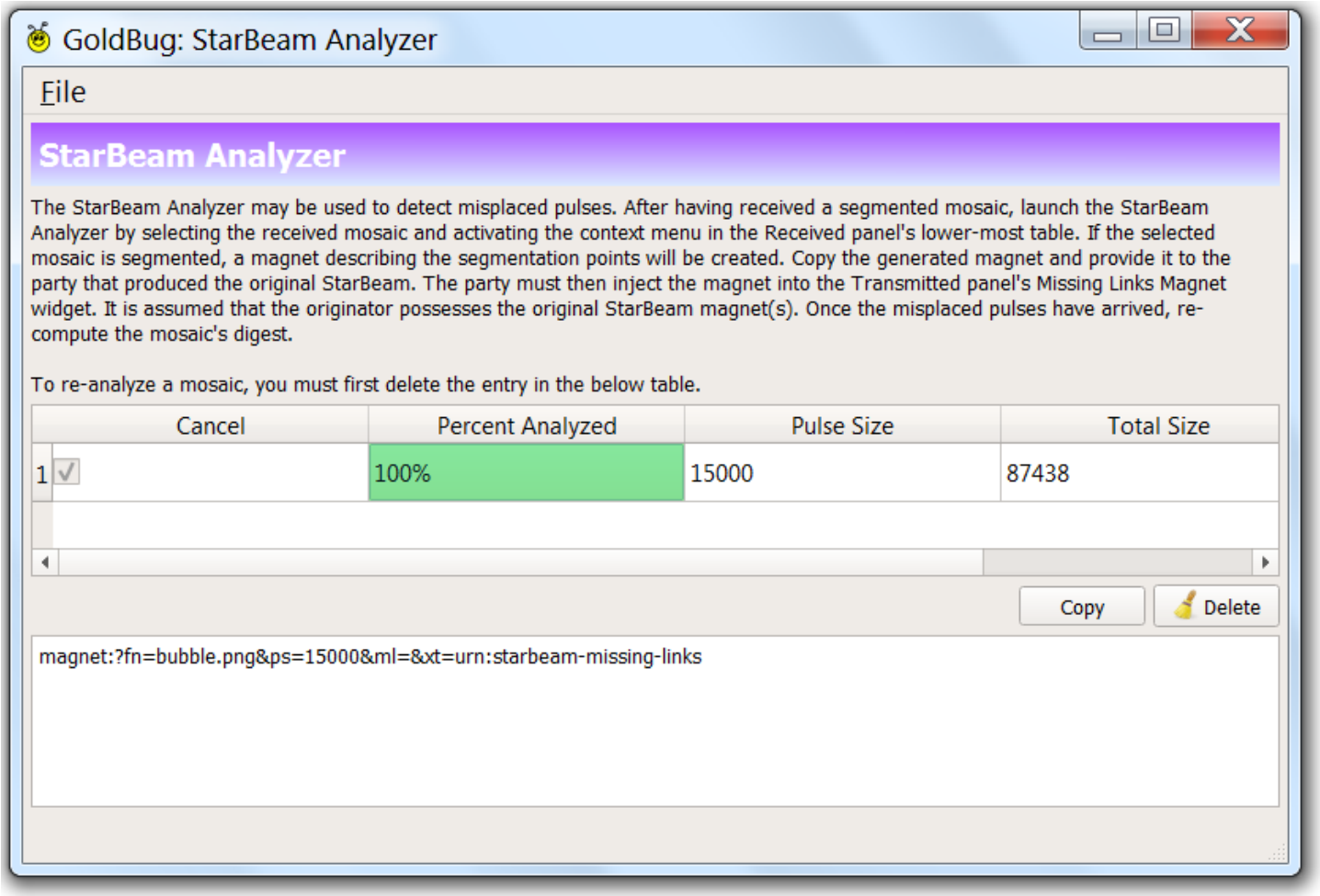
Die erfolgreich heruntergeladen Teile werden beim StarBeam “Mosaics” genannt und in dem gleichnamigen Unterpfad der Installation auf der Festplatte abgelegt. Ähnlich einem Puzzle werden die Mosaik-Teile zu einem vollen Mosaik, der erhaltenen Datei, zusammengesetzt.

Die noch zu übertragenden Datei-Teile werden bei StarBeam „Links“ genannt (vgl. auch den Begriff von “Chunks” im alten EDonkey-Netzwerk oder den Begriff der “Blöcke” im Gnutella-Netzwerk, der durch die Verwendung des damals dort genutzten Tiger-Tree-Hashes geprägt wurde).

### 11.3.1 Werkzeug: Star-Beam Analyzer

Sollte eine Datei mal nicht erfolgreich zu 100 Prozent übertragen worden sein, kann dieses mit dem StarBeam-Analyser Werkzeug überprüft werden. Dieses stellt fest, ob alle Mosaik-Teile vorhanden sind oder ob noch zu übertragende Links/Chunks/Blöcke bzw. Pakete fehlen. Wenn noch Links fehlen, erstellt der SB-Analyser einen Magnet-URI, den der Freund nochmals in seinen Upload-Tab eingeben kann. Dann werden nur die noch fehlenden Links bzw. Mosaike erneut gesandt.

Abbildung: Dateitransfer mittels Starbeam: Analysewerkzeug für die Chunks




Die Datei würde sich auch vervollständigen, wenn der Sender sie dreimal am Tag über das Echo mit der „Rewind“ (= „Erneut senden“)-Funktion sendet.

Es ist zu beachten, dass ein Magnet ein Kanal ist, und vorhandene Dateien in dem lokalen Mosaik-Pfad sodann erneuert werden, wenn kein One-Time-Magnet genutzt wurde und sie nochmals im gleichen Kanal gesendet werden. Ein erneuter Versand der Datei durch den Uploader wird die erhaltene Datei beim Nutzer somit wieder überschreiben, wenn dieser in der Übertragungstabelle keine Lock-Option fest gelegt hat. Mit der Check-Box „Lock“ wird die Datei, die der Nutzer erhalten hat, also nicht gelöscht.

### 11.3.2 Ausblick auf Crypto-Torrents

Aufgrund der Verschlüsselung ist für niemanden einsehbar, welche Datei ein Nutzer herunterlädt, da niemand weiß, ob der Nutzer das Paket erfolgreich entschlüsseln konnte - Und selbst wenn, es weiß niemand, ob der Nutzer sich daraus die Datei erstellt oder abgespeichert hat.

Beim Upload gilt ähnliches. Der Upload ist nur von einer Nachbar-IP einsehbar, wenn dieser Nachbar den Magneten der Datei kennt. Es empfiehlt sich in diesem Fall, wenn man öffentliche Starbeam-Magneten laden möchte, nur zu Nachbarn- bzw. Chat-Servern zu verbinden, denen man vertraut bzw. die man über einen Account-Zugang als Freund definiert.

Auch die oben angesprochene Variante der Setzung eines Nova-Passwortes auf die Datei und die Distribution der physischen Blöcke zeitlich vor der Gewährung der Zugriffsrechte auf das Nova-Passwort in einen zweiten Prozess kann neue Perspektiven in technischen, prozessualen oder auch juristischen Überlegungen bieten.

D.h. der Transfer der Datei erfolgt in der Vergangenheit und der Transfer der Entschlüsselungsoption erfolgt in einem zukünftigen, davon getrennten und nachgelagerten Prozess.

Dann können StarBeam-Magnet-URIs über das Echo-Protokoll eine Rolle bei neuen Wegen des Denkens zum Thema der Entwicklung und Nutzung der in der File-Sharing Community diskutierten “Crypto-Torrents” spielen.

Verschlüsselung bedeutet grundsätzlich, dass Unbefugte nicht wissen, was in dem verschlüsselten Packet ist und dass der Besitzer des Schlüssels selbst entscheidet, wann er die Entschlüsselung durchführt. D.h. Verschlüsselung ist folgerichtig auf den Datei-Transfer und die Souveränität des Nutzers angewandt worden.

## 12 Web-Suchmaschine mit URL-Datenbank

Mit der integrierten Funktion einer Websuche ist GoldBug durch seine genutzte Architektur des Kernels auch eine quelloffene p2p Web-Suchmaschine.

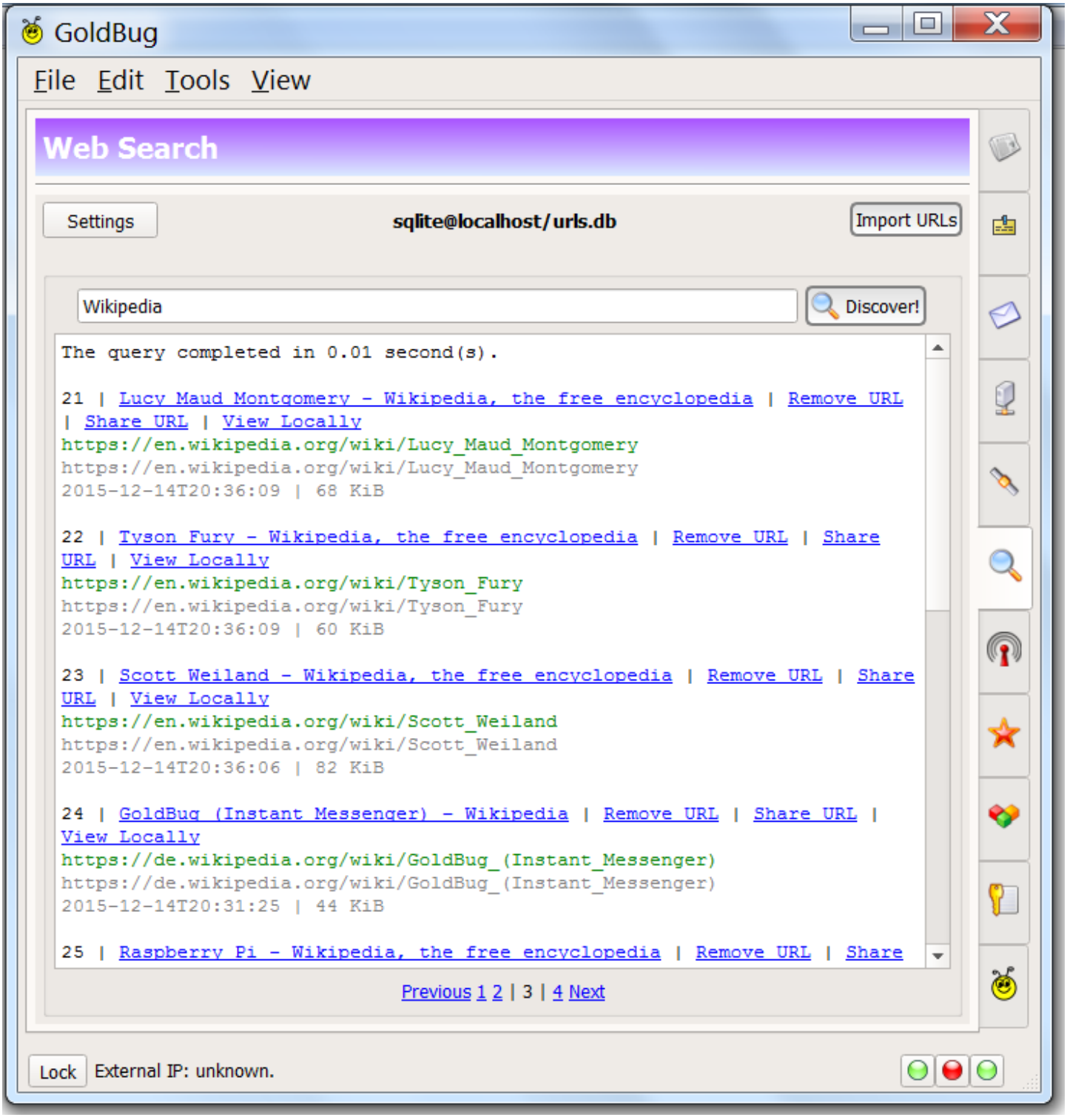
GoldBug ist dabei die (bisherig) einzige und erste unter den wenigen handvollen p2p verteilten Suchmaschinen wie YaCy, Faroo.com, Arado.sf.net oder Grub (die von Wikia-Search bekannt war), die in der Lage ist, den Transfer der URLs über verschlüsselte Verbindungen in ein verteiltes F2F- bzw. P2P-Netzwerk zu gestalten.

Der Anspruch der Web-Suchfunktion in GoldBug ist, nicht nur eine quell-offene Programmierung der Suchmaschine oder des Sortier-Algorithmus anzubieten, sondern auch den URL-Datenbestand quell-offen zu handhaben, so dass jeder Teilnehmer die URLs herunterladen kann. Drittens schließlich geht es darum, dass Transfers und Datenbank-Speicherungen in einer verschlüsselten Umgebung stattfinden. Ein innovatives und exemplarisches Modell, um eine Suche in verschlüsselten Datenbanken durchzuführen.

Titel der Webseite, Stichworte dazu sowie die URL selbst werden verschlüsselt in einer SQLite- oder PostgreSQL-Datenbank abgelegt und über das Echo-Protokoll miteinander vernetzt.

Ein Nutzer kann somit mittels eines Crawlers oder RSS Feeds seine Webseiten und URLs in einem durchsuchbaren Datenbank-Repositorium ablegen und mit anderen Knotenpunkten teilen.

Abbildung: Websuche mit GoldBug in der URL-Datenbank




Der Nutzer kann seine eigene Suchmaschine gestalten: z.B. mit 15 GB an URLs in der Datenbank auf seiner Maschine kann der Nutzer durchaus interessante Suchergebnisse erzielen

hinsichtlich neuer Webseiten, die seine Freunde interessant finden und in das p2p Netzwerk eingegeben haben.

Aber auch als lokale Datenbank für eigene Lesezeichen oder einem eigenen Crawl einer dedizierten Domäne, kann die URL-Datenbank genutzt werden.

Die Websuche in dem URL-Repositorium bleibt anonym, denn die GoldBug URL-Suche erzeugt in anderen Knotenpunkten keine Bekanntgabe der Suchworte, sogenannte Query-Hits.

GoldBug konvertiert die Suchworte in einen Hash und durchsucht die lokalen Datenbanken, ob sie diesen Hash enthalten. Sodann ist dort auch der Hash der URLs verzeichnet, die dieses Suchwort enthalten. Sodann wird die URL-Datenbank nach dem Hash der URL durchsucht.

Auch die Datenbanken sind verschlüsselt, sodass nach dem Suchprozess auch noch ein Entschlüsselungsvorgang angeschlossen wird. Schließlich werden die Suchergebnisse ausgegeben und den Nutzer angezeigt. Die Benutzeroberfläche sortiert derzeit die Ergebnisse zu einem oder mehrerer Suchworte der Einfachheit halber dergestalt, dass die neuesten URLs ganz oben zuerst angezeigt werden.

Wer einen quelloffenen Such-Algorithmus zur Sortierung von URL-Ergebnissen erstellen will, findet bei dieser Funktion in GoldBug also die offene Quellcode-Basis, um sein Algorithmus-Modell nicht nur zu entwickeln, sondern auch gleich einem Praxistest zu unterziehen.

Die Distribution von Webseiten-URLs geschieht nicht über zentrale Server, sondern wird über das verschlüsselte Echo-Protokoll dezentral zwischen den Teilnehmern organisiert: Zwei oder mehrere Nutzer tauschen dazu ihre URL-Schlüssel und nehmen dann am dem p2p Austausch an Webseiten-URLs, z.B. der eigenen Lesezeichen, mit allen ihren Freunden teil. Die online getauschten URLs werden zunächst im Hauptspeicher gesammelt und sodann alle 10 Sekunden in die lokale Datenbank geschrieben.

Weiterhin besteht auch die Möglichkeit, neue URLs in die eigene lokale Datenbank manuell zu importieren. Hierzu wird der Web Browser Dooble.sf.net benötigt. Das erste Symbol in der URL-Zeile des Browsers ermöglicht, eine einzelne URL in eine Zwischen-Datenbank zu speichern: Shared.db. Diese wird dann von GoldBug mit nur einem Klick importiert. Die Shared.db muss im Installationspfad von GoldBug liegen und beide Programme, GoldBug und Dooble, müssen in den Einstellungen den Pfad dieser Datei definieren.

Um eine URL der Webseite, die ein Nutzer gerade ließt, aus dem Web Browser Dooble in die eigene URL-Datenbank von GoldBug zu importieren, ist einfach das erste Icon in der URL-Zeile des Browser zu klicken, um die URL erst mal in der Shared.db abzulegen. Sodann ist in GoldBug im Tabulator der Web-Suche den Knopf “importieren” zu klicken.

Die neue aktuele Version des Browser Dooble unterstützt diese Importfunktion einer einzelnen URL in GoldBug nun jedoch nicht mehr. Denn die neue Version des Browsers Dooble stellt eine komplette Neu-Programmierung dar und wurde erforderlich aufgrund der Änderung bei Qt hinsichtlich des Webkit-Moduls.

In dem weiterhin verfügbaren Source Code des alten Dooble Browser kann mit eigener Kompilierung diese Möglichkeit jedoch nochmal reaktiviert werden. Diese Möglichkeit soll hier nur kurz benannt bleiben, da ggf. auch andere Entwickler einen Import einer URL aus einem (jeglichen) Browser in eine verschlüsselte Bookmark-Datenbank intendieren und sich dieses Modell anschauen mögen.

Die Idee, mit Freunden geteilte Bookmarks für die eigene Historie durchsuchbar und lokal speicherbar zu machen, bleibt also aktuell.

Effizienter sind hingegen die weiteren Methoden, zahlreiche URLs mittels eines Crawlers oder des RSS-Feeds in GoldBug zu importieren.

Doch zunächst zum Setup der URL-Datenbank in GoldBug.

## 12.1 Datenbank Setup

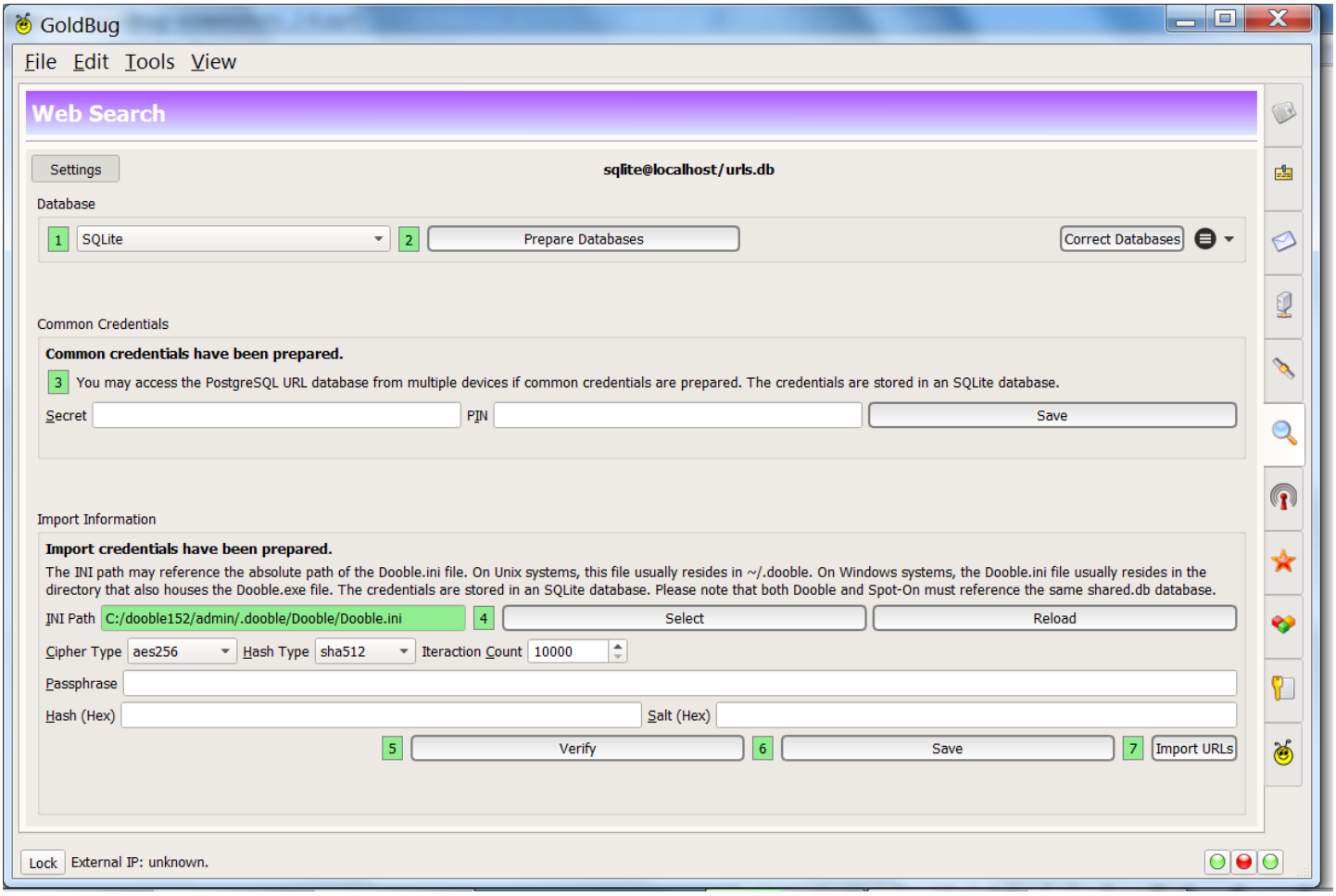
Die URLs können wahlweise in einer SQLite- oder PostgreSQL-Datenbank gespeichert werden. SQLite ist die automatisch konfigurierte Datenbank, die auch Nutzern mit weniger Erfahrung in der Einrichtung von Datenbanken empfohlen wird. Fortgeschrittenere Nutzer können sich auch an eine PostgreSQL-Datenbank-Einrichtung heran begeben. Diese hat Vorteile im Netzwerkzugriff, der Verwaltung von Nutzerrechten und dem Handling großer URL-Datenbestände. GoldBug eignet sich daher mit der Funktion, eine eigene Websuche zu gestalten, auch für den Unterricht, um Lernende für die Einrichtung von Datenbanken zu interessieren.

Die URLs werden in 26x26 bzw. 36x36 Datenbanken (2(16^2) = 512 Tabellen) deponiert, die verschlüsselt sind. D.h. die Suche findet in einer verschlüsselten Datenbank (URLs.db) statt. Die Suche in verschlüsselten Datenbanken ist ein bislang noch wenig bearbeitetes Forschungsfeld.

### 12.1.1 SQLite

SQLite ist eine Programmbibliothek, die ein relationales Datenbanksystem enthält. Die gesamte Datenbank befindet sich in einer einzigen Datei. Eine Client-Server-Architektur ist somit nicht vorhanden.

Abbildung: Installation der URL-Datenbank für die URL/Websuche




Die SQLite-Bibliothek lässt sich direkt in entsprechende Anwendungen integrieren, so dass keine weitere Server-Software benötigt wird. Dieses ist auch der entscheidende Unterschied zu anderen Datenbanksystemen. Durch das Einbinden der Bibliothek wird die Anwendung um Datenbankfunktionen erweitert, ohne auf externe Softwarepakete angewiesen zu sein.

SQLite hat einige Besonderheiten gegenüber anderen Datenbanken: Die Bibliothek ist nur wenige hundert Kilobyte groß. Eine SQLite-Datenbank besteht aus einer einzigen Datei, die alle Tabellen, Indizes, Views, Trigger usw. enthält. Dies vereinfacht den Austausch zwischen verschiedenen Systemen.

### 12.1.2. PostgreSQL

PostgreSQL - auch bekannt unter dem Namen Postgres - ist ein freies, objektrelationales Datenbankmanagementsystem (ORDBMS). Seine Entwicklung entstand in den 1980er Jahren aus einer Datenbankentwicklung der University of California in Berkeley, seit 1997 wird die Software von einer Open-Source-Community weiterentwickelt.

PostgreSQL ist weitgehend konform mit dem SQL-Standard ANSI-SQL 2008. PostgreSQL ist vollständig ACID-konform, und unterstützt erweiterbare Datentypen, Operatoren, Funktionen und Aggregate.

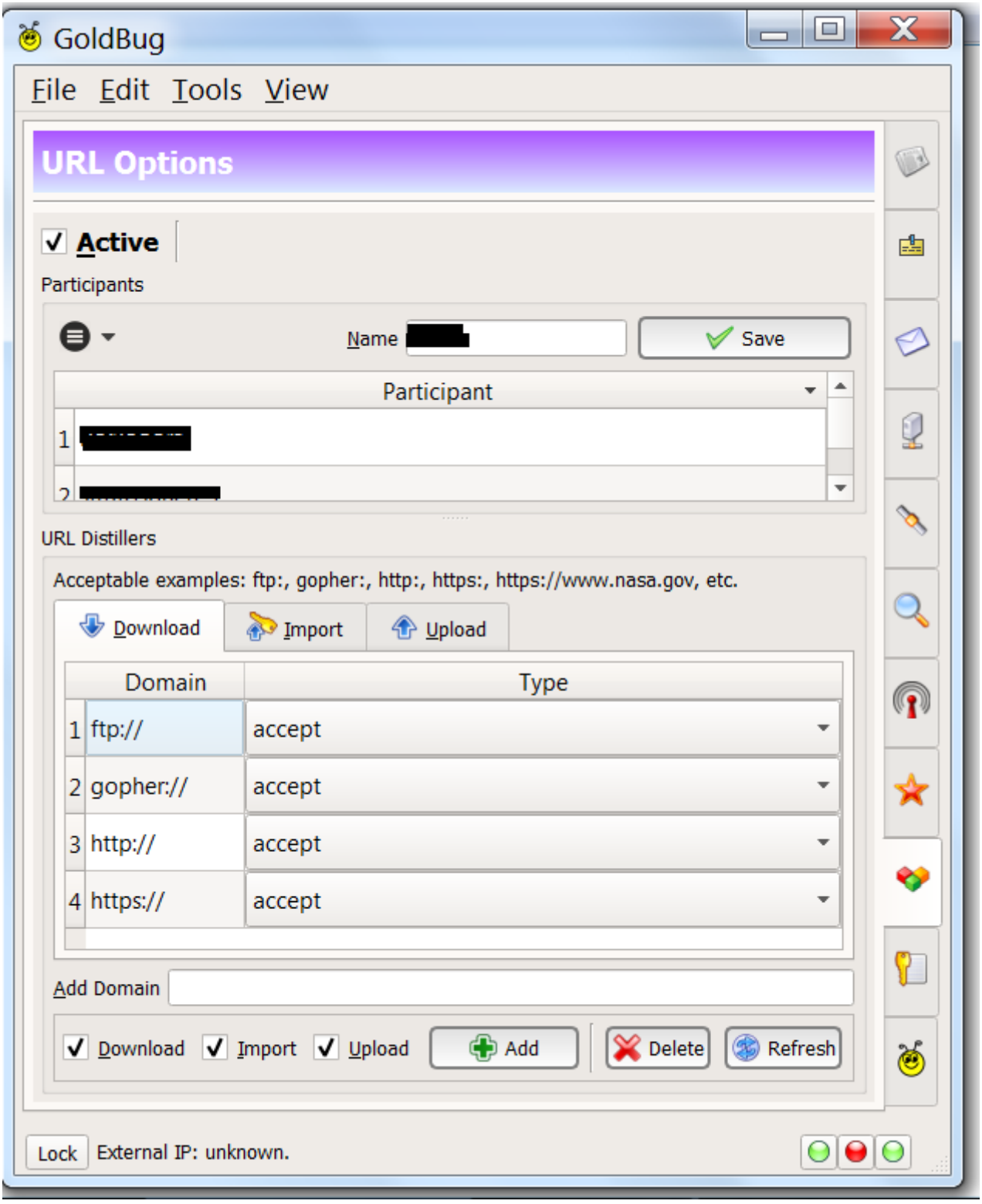
In den meisten Linux-Distributionen ist PostgreSQL enthalten - auch Windows und Mac OS X werden unterstützt. Da er Einrichtungsprozess der PostgreSQL Datenbank umfangreicher ist,

soll hier auch hinsichtlich der eigenen p2p-Fähigkeit dieser Datenbank außerhalb des p2p Echo-Netzwerkes vertiefend auf die Handbücher dieser Datenbank verwiesen werden.

## 12.2 URL-Filter

Wenn der Nutzer nun am dem p2p Prozess des URL-Austausches teilnimmt, erhält er alle URLs, die andere dem System hinzugefügt haben. Um maliziöse URLs auszuschließen, kann der Nutzer in der Websuche URLs auch mit einem Klick wieder löschen - oder aber er nutzt von Beginn an den URL-Filter, der sich in einem eigenen Tabulator findet.

Abbildung: URL Options: Import- und Export-Filter: URL-Distiller




URL-Filter - sogenannte Distiller - können eingehende, ausgehende und zu importierende Datenbestände mit einer Blacklist oder Whitelist filtern. So kann der Nutzer z.B. definieren, nur URLs von der Domain www.wikipedia.org zuzulassen oder dass sein Upload an URLs zu Freunden nur von der Domain seiner Universität erfolgt. Auch kann der Nutzer festlegen, dass er URLs einer bestimmten Länder-Domäne nicht erhalten will.

In dem Fall, dass der Nutzer keine URLs erhalten will, setzt er für die Downloads den Distiller-Filter einfach auf “http://” mit dem Wert “Deny”. URLs werden dann nicht angenommen.

Ganz wichtig: damit der Filter aktiv ist, ist oben in der Check-Box der Filter auf auf “Aktiv” zu markieren.

## 12.3 URL-Community

Damit der Nutzer mit Freuden URLs tauschen kann und sein Datenbestand für die Websuche wächst, kann er den URL-Key entweder manuell im Tabulator URL-Filter in die Teilnehmer-Tabelle einkopieren; oder aber: als zweite Möglichkeit besteht auch die Option, seinen URL-Key an eine Community zu senden.

Wenn der Freund des Nutzers ebenso online ist, und der Nutzer über das Werkzeug “EPKS” - Echo Public Key Share - seinen URL-Key an die dort definierte Community “Spot-On URL Community” sendet, erhält sein Freund den URL-Key des Nutzers automatisch online transferiert.

Dieser Transfer ist verschlüsselt über das Echo-Protokoll und nutzt als symmetrische Verschlüsselung den Namen der URL-Community. Es ist ähnliche einem Gruppenchat-Raum (e’IRC/Buzz-Funktion), in dem die URL-Schlüssel dann hin gesandt und automatisch integriert werden. Wie EPKS funktioniert, ist weiter unten noch ausführlicher beschrieben.

## 12.4 Pandamonium Webcrawler

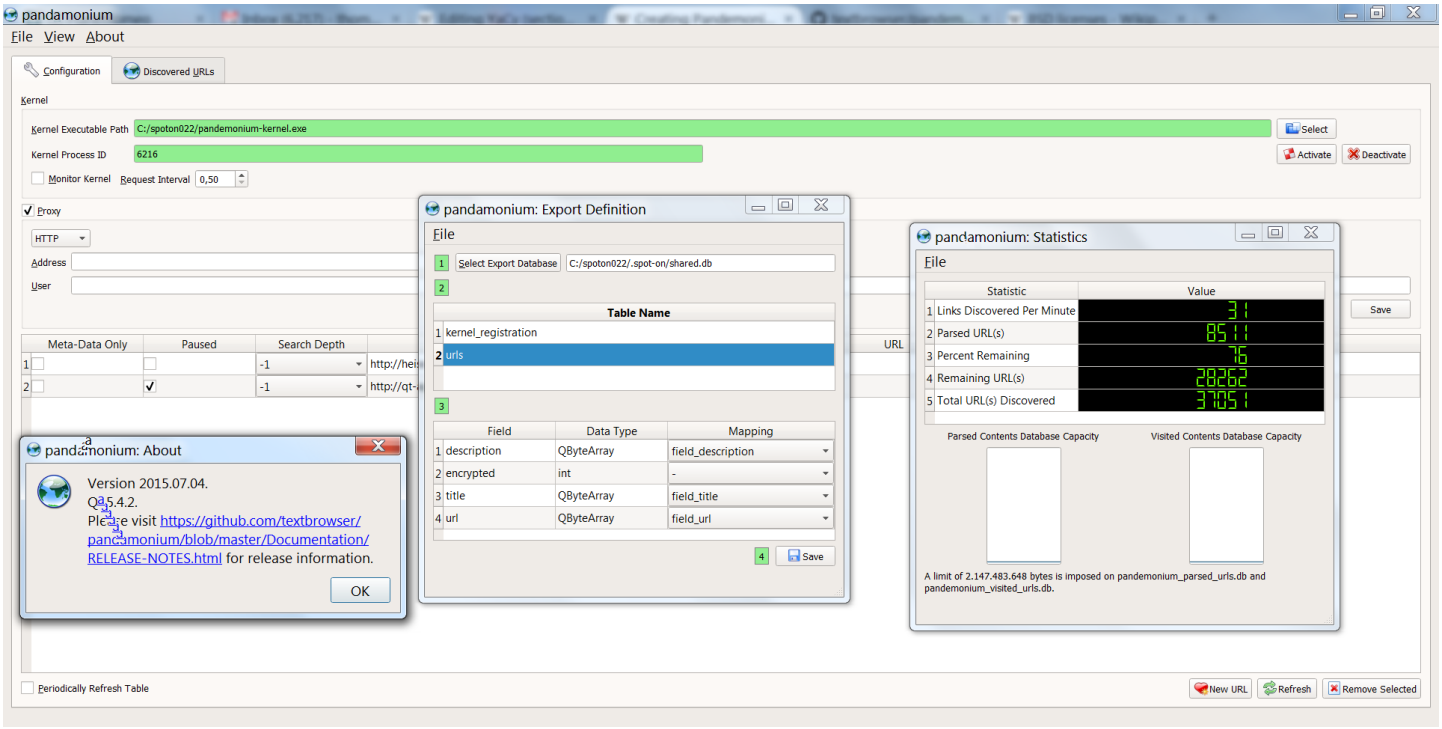
Eine weitere Import-Möglichkeit ist, den Cawler “Pandamonium” zu nutzen.

Die Veröffentlichung von GoldBug Version 2.8 (Christmas-Release 2015) trägt den Namen “Pandamonium Webcrawler Release” und bezieht sich auf den Webcrawler namens Pandamonium, der als Werkzeug der URL-Datenbankfunktion hinzugefügt wurde.

Der Web-Crawler durchsucht eine Domain nach allen verlinkten URLs und kann dann auch auf den so entdeckten Webseiten neue URL indizieren und dem Crawl bzw. Index hinzufügen. Pandamonium funktioniert ebenso wie der Import aus dem Dooble Web Browser über eine zwischengeschaltete Shared.db. Der Web-Crawler Pandamonium ist ebenso quelloffen und kann unter dieser URL heruntergeladen werden:

https://github.com/textbrowser/pandamonium

Abbildung: Pandamonium Web Crawler




Die so hinzugefügten URLs werden dann ebenso mit den Freunden über verschlüsselte Verbindungen geteilt bzw. in der eigenen, lokalen Datenbank ebenso verschlüsselt abgespeichert.

So besteht mit dem Crawler Pandamonium die Möglichkeit, große Mengen an Webseiten gewünschter Domains für eine Websuche im Klienten GoldBug zu importieren.

Neben der URL wird mittels Pandamonium auch die Webseite als Rich-Text (d.h. also ohne Bilder) in der Datenbank mit abgespeichert und dieser Datenbestand kann ebenso mit Freunden geteilt werden. Die Websuche in GoldBug ermöglicht somit das lokale Browsen von Webseiten, ohne das Internet oder die Domain zu kontaktieren und seine IP-Informationen dort bekannt zu geben.

Es ist quasi eine neue Art und weiterentwickelte Idee des Anonymisierungsnetzwerkes Tor: Nicht mehr die Webseite wird über ein p2p-Proxy-Netzwerk live kontaktiert, sondern die URL wird in einer p2p-Websuche bzw. Datenbank gesucht und die Webseite wird gleich als Rich-Text mitgeliefert und kann lokal geladen, gebrowsed und gelesen werden, wie aus einem Browser-Cache oder Proxy.

Java-Scripte, Bilder und Refer-URLs sowie IP-Informationen werden dabei nicht einbezogen. Der Nutzer ist somit vor der Preisgabe seiner Daten geschützt und kann dennoch die gewünschte Webseite einer URL lesen, wenn sie in seinem oder dem geteilten Datenbestand vorhanden ist. Während Webseiten aufgrund von Javascript beim Anonymisierungstool Tor auch zusätzliche Verbindungen aufrufen können oder Spuren hinterlassen, ist es bei dem Webcrawler Pandamonium vorzugsweise ausgeschlossen, dass solche Sicherheitsrisiken bestehen.

Verschiedene Revisionen von Webseiten zu verschiedenen Aufrufzeiten der Webseite (Memento) werden ebenso unterstützt - sowohl beim Crawler als auch bei der Websuche im

GoldBug Klienten. Der Page-Viewer der Websuche in GoldBug zeigt verschiedene Revisonen der Webseite an, wenn diese vorhanden sind.

Die Einrichtung der SQLite Datenbank für den Import der URLs aus dem Pandamonium Webcrawler über die shared.db ist in wenigen Schritten erfolgt:

- Der Nutzer erstellt eine SQLite-Datenbank im GoldBug Programm unter Websuche/Einstellungen.
- Der Nutzer gibt nun noch ein Passwort für “Common Credentials” ein. Dieses ist eine Passwort-Funktion für den Fall, dass dritte, weitere Anwendungen URLs für den Import zur Verfügung stellen.
- Sodann verfiziert der Nutzer alle Eingaben und starte den Import aus shared.db, in die er zuvor die durch den Pandamonium Webcraler gesammelten URLs abgelegt hat: Der Importprozess holt sich die URLs aus dieser Datei ab und fügt sie dem URL-Datenbestand im GoldBug Klienten hinzu (URLs.db).
- Alle so importierten URLs werden gegebenenfalls mit den Freunden des Nutzers online p2p geteilt. Dazu ist der URL-Schüssel des Freundes im Tabulator URL-Filter durch den Nutzer einzugeben, oder er nutzt die URL-Sharing-Community wie oben bereits beschrieben, um den URL-Schlüssel zu tauschen.

## 12.5 RSS-Reader und URL-Import

Die RSS Funktion erweitert den GoldBug Klienten zu einem RSS-Reader. RSS 2.0 Feeds werden unterstützt. Die News-URLs werden in einer Timeline angezeigt, so dass die neueste Nachricht immer obenauf ist.

Zusätzlich werden die News-URLs indexiert, d.h. für die lokale Websuche in GoldBug aufbereitet. Der Import von der verschlüsselten RSS-Datenbank in die verschlüsselte URL-Datenbank kann automatisch periodisch erfolgen, oder auch über einen manuellen Import-Knopf nur auf Aktion des Nutzers.

Die RSS Funktion ermöglicht nicht nur, seine ausgewählten Nachrichtenportale komfortabel auf einer Nachrichtenseite zu lesen, sondern die Nachrichten-URLs auch in seine lokalen URL-Datenbank manuell oder automatisiert zu importieren.

Abbildung: RSS-Feed-Reader zur Importierung von URLs in die URL-Datenbank/Websuche




Die Indexierung der Webseite nutzt die 50 (oder nach Einstellung des Nutzers auch mehr) längsten Wörter der Nachricht, um sie für den Suchindex der URL-Datenbank beim Import aufzubereiten.

Für die Timeline werden die Titel der Nachrichten mit einem Hyperlink erst dann versehen, wenn die Indexierung erfolgt ist. Die Status-Zeile gibt entsprechend eine Statistik wieder, wie viele RSS-Feeds abonniert sind, wie viele URLs schon indexiert sind, wie viele URLs aus der RSS-Datenbank in die URL-Datenbank für die Websuche importiert wurden – sowie die im RSS-Fenster insgesamt lesbaren Nachrichten bzw. URLs.

Das Lesen der Nachrichten erfolgt in einem Page-Viewer, der die Nachrichten nicht in einem Browser darstellt, sondern aus Sicherheitsgründen nur in Textform darstellt. Damit sind auch Javascripte, Bilder und Werbung aus den Seiten entfernt, es werden nur die ASCII-Zeichen der Webseite dargestellt sowie die Hyperlinks zu weiteren Webseiten. Mit dem Kontext-Menü können URLs und Hyperlinks manuell auskopiert werden für eine Ansicht im (externen) Browser.

Der RSS-Reader ist proxy-fähig und kann daher auch hinter restriktiven Umgebungen den Inhalt der Webseiten erhalten und sodann für die Speicherung und Suche in GoldBug zur Verfügung stellen.

Eine Funktion, die heute sicherlich auch einige Browser bieten, Webseiten aus dem Cache des Nutzers heraus aufzurufen oder die URL-Historie durchsuchbar anzubieten.

GoldBug ermöglicht dieses in einer verschlüsselten und p2p Umgebung für die lokale eigene Speicherung in einem dedizierbaren URL-Repositorium.

## 13 Chat/E-Mail-Server einrichten

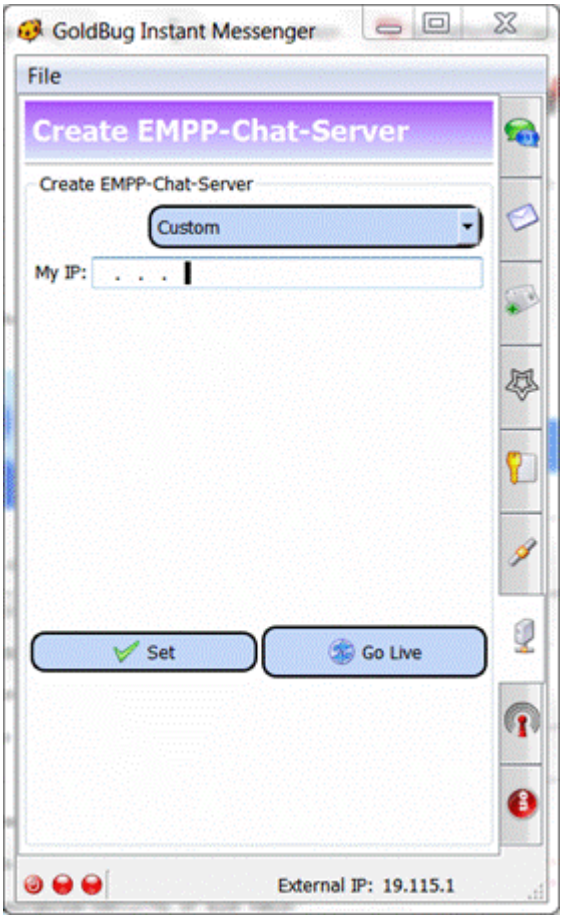
Einen Chat-Server oder Spot-on-Kernel einzurichten bedeutet, einen sogenannten „Listener“ einzurichten, so der technische Begriff.

Wenn der Nutzer sich in der minimalen Ansicht der Benutzeroberfläche befindet, ist ein Chat- & E-Mail-Server bzw. Listener ebenso schnell eingerichtet wie im oben beschriebenen Tabulator eine IP-Verbindung zu einem Server bzw. Nachbarn hergestellt ist. Der Nutzer benötigt keine erweiterten Server-Administrations-Kenntnise, um einen GoldBug-Knotenpunkt auf seinem Webserver laufen zu lassen, einen Chat-Server aufzusetzen oder gar ein E-Mail Postfach für sich und seine Freunde einzurichten. In GoldBug muss dazu lediglich ein sogenannter Listener an einem definierten Port bestätigt werden. Und das ist mit wenigen Klicks möglich. Eine wohl der einfachsten Chat-Server-Administrationen im Vergleich zu anderen Server-Setups überhaupt.

## 13.1 Chat-/E-Mail-Server über einen Listener einrichten

Nochmal zur Erinnerung: Im Tabulator “Verbindung herstellen”, verbindet der Nutzer seinen GoldBug mit einem anderen Knoten oder Nachbarn, und mit dem Tab “Chat-Server” erstellt der Nutzer einen Server bzw. Listener, so dass andere zu ihm verbinden können. Egal welche Methode, Nachrichten kann man immer senden, wenn die zweite oder dritte LED in der Statuszeile leuchtet und ein Nachbar verbunden ist. Entweder zum Nutzer als Server/Listener oder der Nutzer als Klient zum Nachbarn, der einen Listener anbietet.

Abbildung: Einen Chat-Server erstellen (Minimale Ansicht)




Die rechte (dritte) LED in der Statuszeile zeigt also an, dass der Nutzer einen eigenen Chat-Server auf seinem Computer eingerichtet hat.

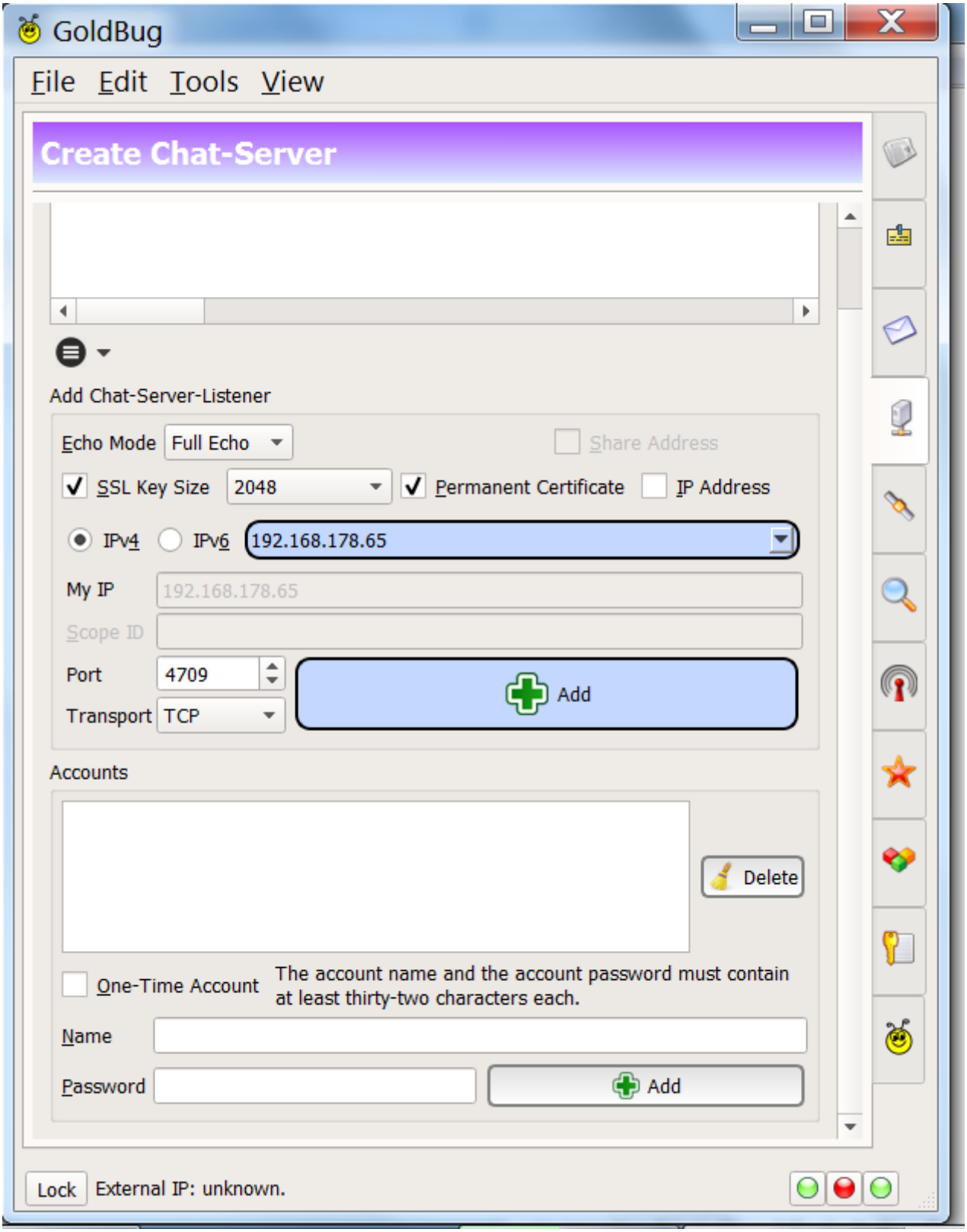
Dazu muss der Nutzer die lokale IP-Adresse seiner Maschine im Tabulator “Chat Server” eingeben. Es handelt sich hier nicht um die (externe) IP-Adresse des Routers, sondern um die lokale Netz-IP-Adresse des Gerätes, auf dem GoldBug Installiert ist. Auch hier erhält man über das Pulldown-Menü eine Auswahl, die IP wird direkt angezeigt und man kann dann die lokale IP wählen. Als Port wird dann wieder automatisch 4710 definiert.

Sodann ist auf den Knopf „Set“ zu drücken und der Eintrag des Listeners war erfolgreich, wenn die dritte LED leuchtet.

### 13.1.1. Server Broadcast

Wenn der Nutzer einen zu seinem Listener verbundenen Klienten hat, oder der Nutzer im “Verbinde-Nachbar”-Tabulator von sich aus zu einem anderen Chat-Server oder Freund verbunden ist, kann der Nutzer sodann auch in der Tabelle mit rechter Maustaste den Befehl “informationen veröffentlichen” klicken. Damit wird sein Chat Server über die bestehenden Verbindungen seinen verbundenen Freunden bzw. Nachbarn sowie auch deren Freunden mitgeteilt. “Server veröffentlichen” meint also “Broadcast IP+Port” des eigenen Chat-Servers an seine (verbundenen) Freunde und Nachbarn. Dann können die Freunde auch zu seinem Chat-Server automatisch verbinden. Der Nutzer muss in diesem Fall also keine IP-Adresse mehr mitteilen oder seine Freunde die eigene IP-Adresse manuell eintragen lassen. Alles geht dann automatisch und der Server des Nutzers steht seinen Freunden und deren Freunden als Peer zur Verfügung. So einfach kann ein Chat-Server erstellt und auch an andere im Netzwerk kommuniziert werden.

Abbildung: Chat-Server einrichten




Der Listener bzw. Chat-Server wird standardmäßig für das TCP Protokoll eingerichtet, GoldBug ist jedoch auch dafür ausgestattet, einen Listener über das UDP oder drittens auch SCTP Protokoll einzurichten. Beide letztgenannten Protokoll sind ideal für VOIP oder Streams. Viertens ist auch ein Chat-Server/Listener über Bluetooth möglich, siehe dazu den Abschnitt weiter unten.

Daher kann in den Verbindungsoptionen auch definiert werden, ob der Klient des Nutzers sich über TCP, UDP, SCTP oder Bluetooth zum Nachbarn bzw. einem anderen Server verbinden soll.

Der Nachbar oder Listener des Servers kann auf SSL-Verbindungen verzichten, dann wird die Übertragung nicht über HTTPS, sondern nur über HTTP geregelt. D.h. eine Verschlüsselungsschicht entfällt, die verschlüsselte Echo-Kapsel wird nicht durch den HTTPS-Tunnel gesandt, sondern über HTTP – und bleibt dennoch verschlüsselt, weil ja die Echo-Kapsel an sich schon verschlüsselt ist.

### 13.1.2. Sicherheitsoptionen

Wenn der Nutzer sich den Tabulator in der maximalen Ansicht der Benutzeroberfläche anschaut, bestehen weitere Einstellungsoptionen:

Ein Listener kann z.B. die Sicherheitsoption setzen, ein permanentes SSL-Zertifikat zu erzeugen. Damit wird der bei SSL bestehende Diffie-Hellman-Schlüsselaustausch bzw. -Verhandlungsprozess nicht in jeder Sitzung neu verhandelt, sondern ein Angreifer müsste schon einen Aushandlungsprozess in der Vergangenheit kennen, um hier einzugreifen.

Es kann aber sein, dass der Server bzw. Listener sein SSL-Zertifikat mal erneuert, daher macht es ggf. Sinn, Ausnahmen („Exceptions“) zuzulassen, wenn man eine Verbindung einfacher erstellen will und diese zusätzliche Sicherheitsebene nicht perfektionieren will.

Ebenso kann man seinerseits die Schlüsselgröße für die SSL-Verbindung definieren und auch bestimmen, dass Verbindungen unterhalb einer bestimmten SSL-Schlüsselgröße gar nicht erst aufgebaut werden. Einmal wird also definiert, was der Nachbar an SSL-Schlüsselgröße anbieten sollte und das andere Mal wird definiert, welche Schlüsselgröße der Nutzer von einem Server bzw. Nachbarn erwartet.

Schließlich besteht die Option, dass der Klient bestimmt, ob er zu dem Nachbarn mit vollem oder halbem Echo verbindet. Bei halbem Echo wird das Nachrichtenpacket nur an den Nachbarn einen Hop über die direkte Verbindung gesandt. Angenommen, der Freund des Nutzers hat den Webserver eingerichtet und sitzt auch davor und der Nutzer möchte nicht, dass seine Echo-Pakete an dritte und seine Freunde gehen, dann kann der Nutzer mit dem Halben Echo definieren, dass seine Pakete nach Erhalt durch den Server nicht weiter verbreitet werden. Damit chatten die beiden Nutzer über eine direkte IP-Verbindung. Beide Teilnehmer sehen beim Halben Echo die IP-Adresse des Freundes und des Chat-Partners. Beim Vollen Echo muss der Chat Freund nicht Administrator des Knotenpunktes sein, sondern kann wie ein zentraler Chat-Server mehrere Klienten miteinander verbinden.

Sicherheitsoptionen erlauben in der erweiterten Ansicht bei der Erstellung eines Chat-Servers/Listeners weiterhin die SSL-Schlüsselgröße zu definieren sowie auch ein permanentes SLL-Zertifikat vorzuhalten.

Auch kann der Nutzer – falls er eine dauerhafte, stabile IP-Adresse hat - diese in das SSL-Zertifikat einbinden.

Diese drei Maßnahmen machen es Angreifern schwerer, das SSL-Zertifikat auszutauschen oder zu „faken“ – denn es würde sofort erkannt, wenn ein untergeschobenes anderes Zertifikat sich als das originäre ausgeben wollte: weil z.B. der Klient kein neues, sondern das alte, permanente Zertifikat erwartet oder weil die IP-Adresse darin fehlt oder nicht stimmig ist. Auch die SSL-Schlüsselgröße definiert dieses.

### 13.1.3. Proxy- und Firewall-Anmerkungen

Wenn der Nutzer GoldBug als Klient über einen Proxy in der Firma, hinter einer Firewall bzw. einem Proxy der Universität oder auch über das Anonymisierungsnetzwerk Tor laufen lassen will, kann er die Proxy-Details für seinen Nachbarn einfügen.

Als Klient kann der Nutzer Dank des HTTP-Protokolls aus jeder IT-Umgebung verbinden, wenn er in dieser Umgebung auch mit einem Browser surfen kann.

Das ist der Vorteil der Programms GoldBug, dass überall, wo der Nutzer mit seinem Browser surfen kann, auch mit dem GoldBug Messenger e-mailen und chatten kann aufgrund des verwendeten HTTPS Protokolls. Viele andere Programme können dieses je nach Firewall-Einstellungen - z.B. vom Arbeitsplatz aus oder im Studierendenwohnheim - nicht.

Wenn der Nutzer einen Proxy z.B. in seiner Firma oder Universität mit dem GoldBug-Messenger nutzen oder austesten will, dann ist dieses unkritisch, denn es wird eine SSL/TLS- bzw. HTTPS-Verbindung aufgebaut – was für die Proxy-Administratoren kaum unterschiedlich ist wie eine SSL/HTTPS Verbindung zu einer HTTPS-Webseite beim Banking oder beim Einloggen in ein Web-E-Mail.

Entscheidend ist, einen Knotenpunkt im Web mit seinem GoldBug zu adressieren, der ggf. nicht vom Port her durch eine Firewall bzw. den Proxy limitiert wird. Falls das der Fall ist, kann der Nutzer seinen Freund bitten, den GoldBug-Chat-Server auf dem Port 80 oder dem Port 443 statt 4710 einzurichten und diesen ggf. mit Login-Daten für einen Echo-Account zu versehen und diese dann zur Verfügung zu stellen.

Verschlüsselter Traffic bleibt verschlüsselter Traffic und über die Ports 443 oder 80 kann jeder GoldBug Freund bzw. Chat-Server im Web erreicht werden.

Da das Echo-Protokoll nur eine einfache HTTP-Verbindung zu einem Nachbarn benötigt (und nicht zwingend Stun-Server oder einen DHT etc.), und damit ideal über einen Proxy, durch eine Firewall oder über das Tor-Netzwerk abgebildet werden kann, ist es eine sehr einfache Architektur, Chat sicher über einen Proxy oder ein Proxy-Netzwerk zu betreiben.

Wenn der Nutzer in der nicht-minimalen Ansicht noch zusätzliche Merkmale definieren will, ist eine oft genutzte Funktion die des Echo Accounts.

Dazu markiert der Nutzer in der Tabelle den Listener, den er erstellt hat und gibt dann die Account Credentials ein, also Name und Passwort. Der Nutzer teilt dann seinem Freund mit, wie

der Accountname und das Passwort dafür lautet und der Freund wird, wenn er den Nachbarkontakt herstellt, über ein Pop-up Fenster gefragt, diese Credentials einzugeben.

Ebenso kann der Nutzer auch wieder zwischen IPV4 und IPV6 wählen, wenn er einen Listener/Chat-Server erstellen will. Auch können mehrere Chat-Server erstellt werden, indem ein anderer Port gewählt wird. Der Nutzer kann verschiedene Listener mit Port 4710 oder 80 oder 443 erstellen und entscheiden, ob er diese Listener für Freunde mit einem Echo Account definieren will (Friend-Modus), oder für einfacher aufzubauende Verbindungen diese im Peer-Modus ohne Account-Login betreibt.

Echo Accounts definieren also, ob der Nutzer ein F2F-Netzwerk oder ein P2P-Netzwerk aufbaut, denn mit den Account Credentials erstellst der Nutzer ein Web-of-Trust, mit dem nur seine vertrauen Freunde mit dem Login-Passwort verbinden können.

### 13.1.4. GoldBug als Lan Messenger

Wenn der Nutzer einen Peer betreibt, kannst er z.B. auch auf einer LAN-Party eines geschlossen Netzwerkes mit der IP-Broadcast-Funktion allen Teilnehmern mitteilen, dass sein Knotenpunkt einen Listener für die Gäste eröffnet hat. Dank des UDP Protokolls funktioniert der GoldBug Messenger aber auch direkt wie ein Lan-Messenger innerhalb einer geschlossenen Benutzergruppe des LANs.

Dazu ist bereits der LAN-Listener als Nachbar in der Nachbartabelle definiert (IP: 239.255.43.21). Dieser ist lediglich zu aktivieren und andere GoldBug-Installationen im selben Windows-Netzwerk finden sich sodann automatisch für eine Verbindung.

## 13.2 Server/Listener-Erstellung Zuhause hinter einem Router/Nat

Wenn der Nutzer keinen eigenen Server im Web hat oder keinen allgemeinen Nachbarn im Web findet, kann er auch einen eigenen Chat-Server zuhause hinter seinem eigenen Router einrichten und den Port im Router weiterleiten. Sein Freund kann dann direkt als Klient zu seinem Listener verbinden.

Einer von beiden jedoch muss einen Listener erstellen, wenn beide hinter einer Firewall sitzen oder keinen Chat-Server im Web nutzen. Wenn der Nutzer also einen Server hinter seinem Router/Nat zuhause erstellen will, ist wie genannt die lokale IP-Adresse der Maschine für den Listener z.B. 192.168.121.1. zu nehmen. Sodann muss der Nutzer in seinem Router auch den Port weiterleiten, d.h. Port 4710 muss vom Router weitergeleitet werden an 192.168.121.1: 4710. Weiterhin sollte der Kernel - Spot-on-Kernel.exe - sowie auch die GoldBug.exe in der Windows Firewall erlaubt sein. Wenn der Nutzer alles korrekt weitergeleitet hat, kann der Freund an der (externen) IP-Adresse des Routers (siehe z.B. unter www.whatismyip.com) und dem Port 4710 seinen Klienten verbinden.

Wichtig ist nur, dass der Router des Nutzers den Kontaktversuch aus dem Internet am definierten Port an seine lokale Maschine weiterleitet. Dieses ist ein übliches und sicheres

Verfahren und öffnet keinen beliebigen Zugang zu einem Rechner, sondern über den Port und die Applikation wird dabei wie bei vielen anderen Programmen definiert, dass auch nur Pakete in diesem Sinne zugelassen werden.

Der Nutzer kann und muss dieses alles selbst definieren und GoldBug enthält keinen Code, der automatisch Ports im Router weiterleitet, oder öffnet oder gar automatisch einen Listener einrichtet!

Somit ist es in GoldBug sicherer und bedarfsgerechter geregelt als bei anderen Applikationen, die im Sinne der Nutzerfreundlichkeit sich selbst konfigurieren und diese Mühe zwar abnehmen und eine Automation im Hintergrund anbieten, jedoch auch den Nutzern, die die technischen Details der Portweiterleitung vom Router und Listener-Definiton kennen, automatisiert Ports öffnen und weiterleiten.

## 13.3 Nutzung von GoldBug im TOR-Netzwerk

Wenn der Nutzer seinen GoldBug-Chat über das Tor-Netzwerk betreiben will, geht dieses ebenso sehr komfortabel, so dass ein Tor-Exit Node nur den Verschlüsselungstext von GoldBug sehen wird. Hierbei liegt der Chat-Server wieder im normalen Web außerhalb des Tor-Netzwerkes.

Bislang kann Tor keine HTTPS-Verbindungen am Exit-Node des Tor-Netzwerkes aufbauen, eine Durchleitung der verschlüsselten Pakete zweier GoldBug-Instanzen sollte jedoch möglich sein: GB -> Tor -> Tor -> Tor -> GB. Ein HTTP-Listener kann ebenso eingerichtet werden.

## 13.4 Spot-On Server

Neben GoldBug, das eine alternative Benutzeroberfläche zum Spot-On-Kernel darstellt, besteht auch noch die original Spot-On Benutzeroberfläche unter spot-on.sf.net. Auch mit Spot-On kann ein Chat Server Listener eingerichtet werden.

## 13.5 Spot-On Lite Server als Deamon

Wer einen Chat-Server ohne Benutzeroberfläche administieren will, somit den Chat-Server als Kernel-Daemon auf einem Webserver einsetzt, kann sich die weitere Server-Software für Echo-Klienten unter github.com/textbrowser/spot-on-lite ansehen.

## 13.6 SmokeStack Server unter Android

Die einfachste Variante zuhause im LAN oder auch mit Port-Weiterleitung im Router einen Chat-Server einzurichten ist, auf einem Android-Gerät die App SmokeStack zu installieren. Auch dieser Android-Server kann die Echo-Pakete der GoldBug-Klienten weiterleiten. Zu finden unter: github.com/textbrowser/smokestack

## 13.7 Bluetooth Server

Schließlich ist auch ein Chat-Server/Listener über Bluetooth möglich (seit Version 2.8, in Abhängigkeit von Qt derzeit nur für Linux). Mit Bluetooth besteht die Möglichkeit, z.B. auf einer Lan-Party die Geräte kabellos über das Echo-Protokoll zu verbinden. Diese Option kann sehr entscheidend sein, wenn kein Internet oder keine entsprechende Infrastruktur mehr zur Verfügung stehen sollte.

Abbildung: Bluetooth Chat Server

![Abbildung: Bluetooth Chat Server] (/images/bluetooth_chatserver.png)

## 13.8 UDP Server

Das User Datagram Protocol, kurz UDP, ist ein minimales, verbindungsloses Netzwerkprotokoll, das zur Transportschicht der Internetprotokollfamilie gehört.

Die Entwicklung von UDP begann, als man für die Übertragung von Sprache ein einfacheres Protokoll benötigte als das bisherige verbindungsorientierte TCP. Es wurde ein Protokoll benötigt, das nur für die Adressierung zuständig war, ohne die Datenübertragung zu sichern, da dies zu Verzögerungen bei der Sprachübertragung führen würde.

Ein Drei-Wege-Handschlag wie bei TCP (dem Transmission Control Protocol) für den Aufbau der Verbindung würde in diesem Fall unnötigen Overhead erzeugen.

UDP ist daher ein verbindungsloses, nicht-zuverlässiges und ungesichertes wie auch ungeschütztes Übertragungsprotokoll. Das bedeutet, es gibt keine Garantie, dass ein einmal gesendetes Paket auch ankommt, dass Pakete in der gleichen Reihenfolge ankommen, in der sie gesendet wurden, oder dass ein Paket nur einmal beim Empfänger eintrifft. Eine Anwendung, die UDP nutzt, muss daher gegenüber verlorengegangenen und unsortierten Paketen unempfindlich sein oder selbst entsprechende Korrekturmaßnahmen und ggfs. auch Sicherungsmaßnahmen vorsehen.

Für das Echo Protokoll eine interessante Grundlage, da hier die Pakete ja auch im Fluss des Netzwerkes eher ungerichtet sind und durch die Multiplikation verlorene UDP-Pakete ja auch wieder durch die Redundanz ausgeglichen werden.

## 13.9. SCTP Server

Das Stream Control Transmission Protocol (SCTP) ist ein zuverlässiges, verbindungsorientiertes Netzwerkprotokoll. Als Transportprotokoll steht SCTP auf derselben Stufe des TCP/IP-Referenzmodells wie TCP und UDP.

SCTP realisiert das Konzept einer Association: Hier wird eine Verbindung aufgebaut, in der mehrere Nachrichten-Datenströme in sich reihenfolgenerhaltend (untereinander aber potentiell

nicht-reihenfolgenerhaltend) transportiert werden. Zusätzlich können einzelne, zum Beispiel dringende, Datagramme separat und außer der Reihe verschickt werden, die dadurch eventuell die In-Order-Datenströme „überholen“.

Auch dieses Protokoll für die Übertragung der Echo-Pakete zu nutzen ist für die Forschung sehr interesant, da die eher ungerichteten Echo-Pakete ggf. mit diesem Protokoll im Vergleich zu UDP eine abgesichertere Übertragung erfahren.

Auch mit diesem Protokoll kann ein Chat-Server für GoldBug eingerichtet werden.

## 13.10 Ncat Verbindung

Während andere Applikationen immer einen Server benötigen, der ggf. sogar nur sehr schwer replizierbar oder administrierbar ist, hat GoldBug auch Möglichkeiten, ganz ohne dedizierte Server-Software auszukommen. Dazu kann Ncat benutzt werden wie folgt:

In einer Übung werden zwei Geräte mit GoldBug durch einen RaspberryPi verbunden, der unter Debian läuft und NCat benutzt. Es wird ein funktionierendes Netzwerk benötigt, ein RaspberryPi und zwei Geräte mit jeweils GoldBug.

Zuerst wird ncat auf dem Pi installiert:

```
sudo aptitude install nmap
```

Sodann wird etwas SSL Material generiert:

```
openssl req -new -x509 -keyout server-key.pem -out server-cert.pem
```

Sodann wird ncat aufgerufen.

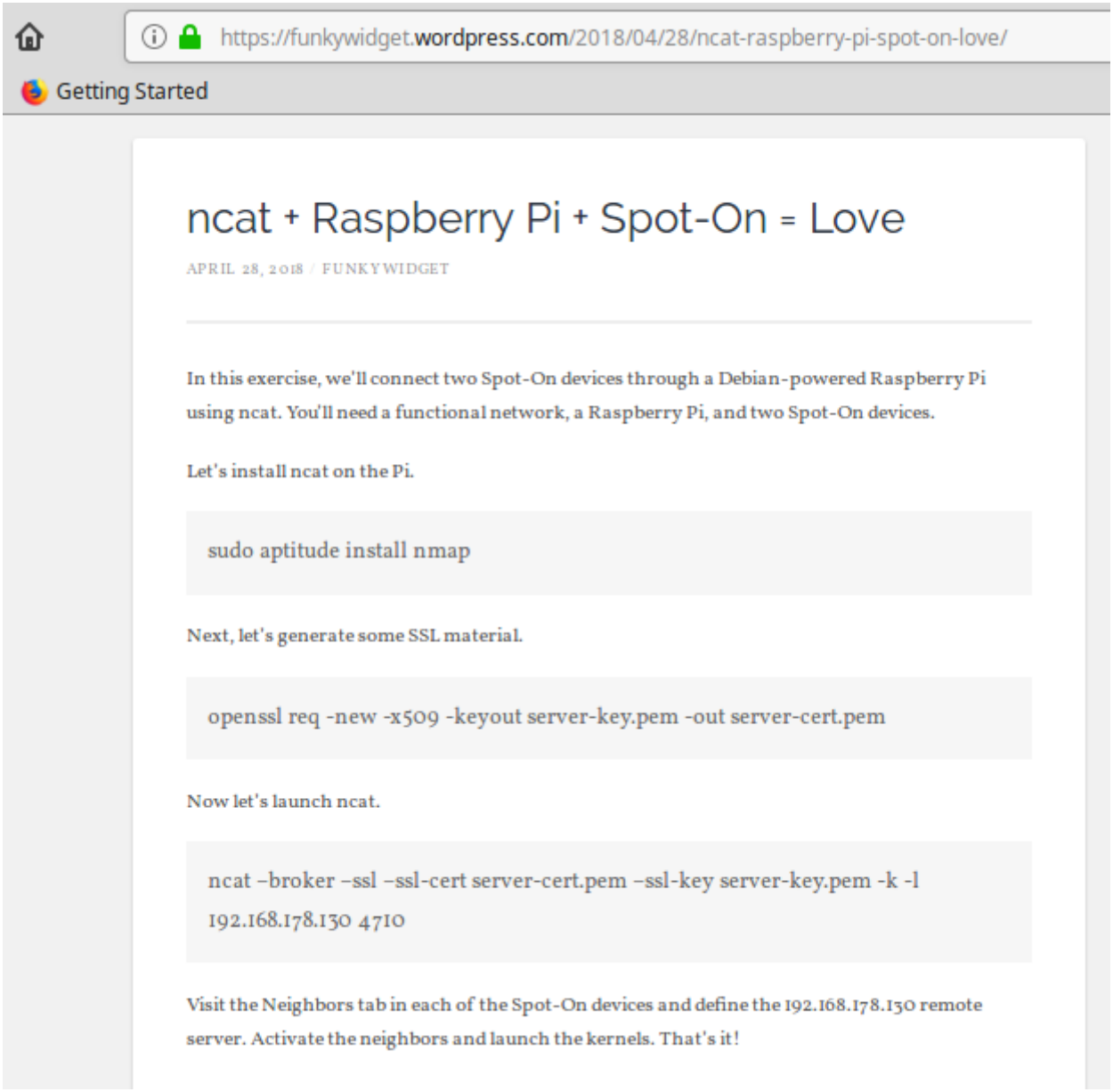
```
ncat –broker –ssl –ssl-cert server-cert.pem –ssl-key server-key.pem -k -l
192.168.178.130 4710
```

Nun besucht der Nutzer den Nachbarn/Server Tabulator in GoldBug und definert unter 192.168.178.130 den remoten Server.

Wenn die Nachbargeräte aktiviert wurden und die Kernels angeschaltet sind, besteht die Verbindung bereits.

Eine schöne Übung, um ein Netzwerk und Ncat für verschlüsselte Kommunikation auszutesten.

Abbildung: Übung Verbindung über Ncat



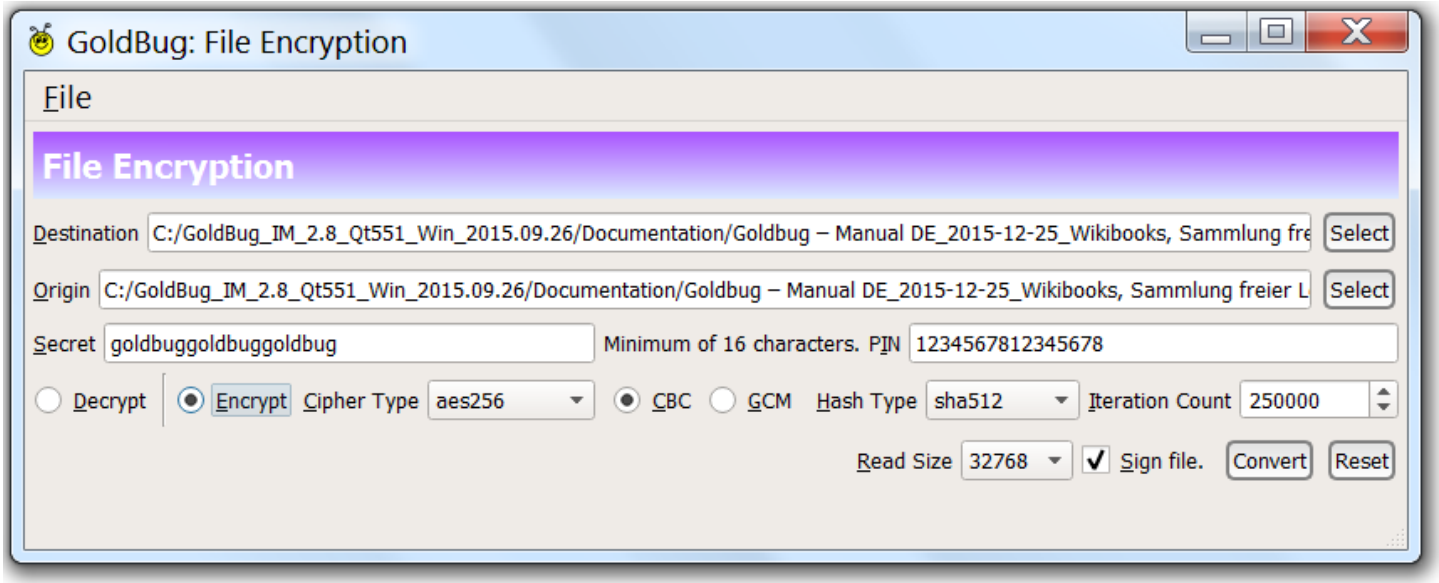

# 14 Werkzeuge

Neben den regulären Funktionen bestehen im GoldBug Messenger auch verschiedene Werkzeuge, die nützliche Funktionen anbieten. Zu nennen sind hier im Wesentlichen die Funktionen der Verschlüsselung von Dateien (File-Encryptor), einem weiteren Werkzeug zum Umwandeln von normalen Text und Cipher-Text (Rosetta-CryptoPad) sowie dem EPKS-Werkzeug, mit dem online die öffentlichen Schlüssel zur Verschlüsselung übertragen werden können. Weiterhin sind die Pass-Through Funktionalität und auch die Werkzeuge für Statistiken und Analysen zu nennen.

## 14.1 Werkzeug: Verschlüsselung von Dateien

GoldBug verfügt über zusätzliche Werkzeuge für die Verschlüsselung. Im Hauptmenü unter Werkzeuge findet der Nutzer das Werkzeug für die Verschlüsselung von Dateien auf seiner Festplatte (“File Encryption Tool”).

Abbildung: File-Encryptor - Werkzeug zur Datei-Verschlüsselung




Damit kann der Nutzer eine Datei von der Festplatte bestimmen, sodann den gleichen Pfad angeben und eine beliebige Endung bzw. Änderung des Dateinamens wählen - sodann Passwort und Pin eingeben (beides natürlich wieder mindestens 16 Zeichen) und mit den Radio-Auswahl-Knöpfen definieren, ob die Datei ver- oder ent-schlüsselt werden soll.

Cipher- und Hash-Type sind ebenso definierbar wie auch, dass eine Signatur in die Verschlüsselung optional eingebaut werden kann, damit sichergestellt wird, dass die Verschlüsselung auch nur durch den definierten Nutzer erfolgte (und niemanden anderes).

Das Werkzeug zur Dateiverschlüsselung ist ein Angebot, um z.B. potentiell unsichere Truecrypt Container zu ersetzen bzw. ergänzend zu verschlüsseln oder um einzelne Dateien zu sichern, bevor der Nutzer sie transferiert - sei es als E-Mail in GoldBug, über StarBeam in GoldBug oder auch über konventionelle, unsichere Wege - oder einfach, um sie auf der Festplatte oder bei Speicherung in Online-Speichern wie Dropbox oder Megaupload zuvor zu verschlüsseln.

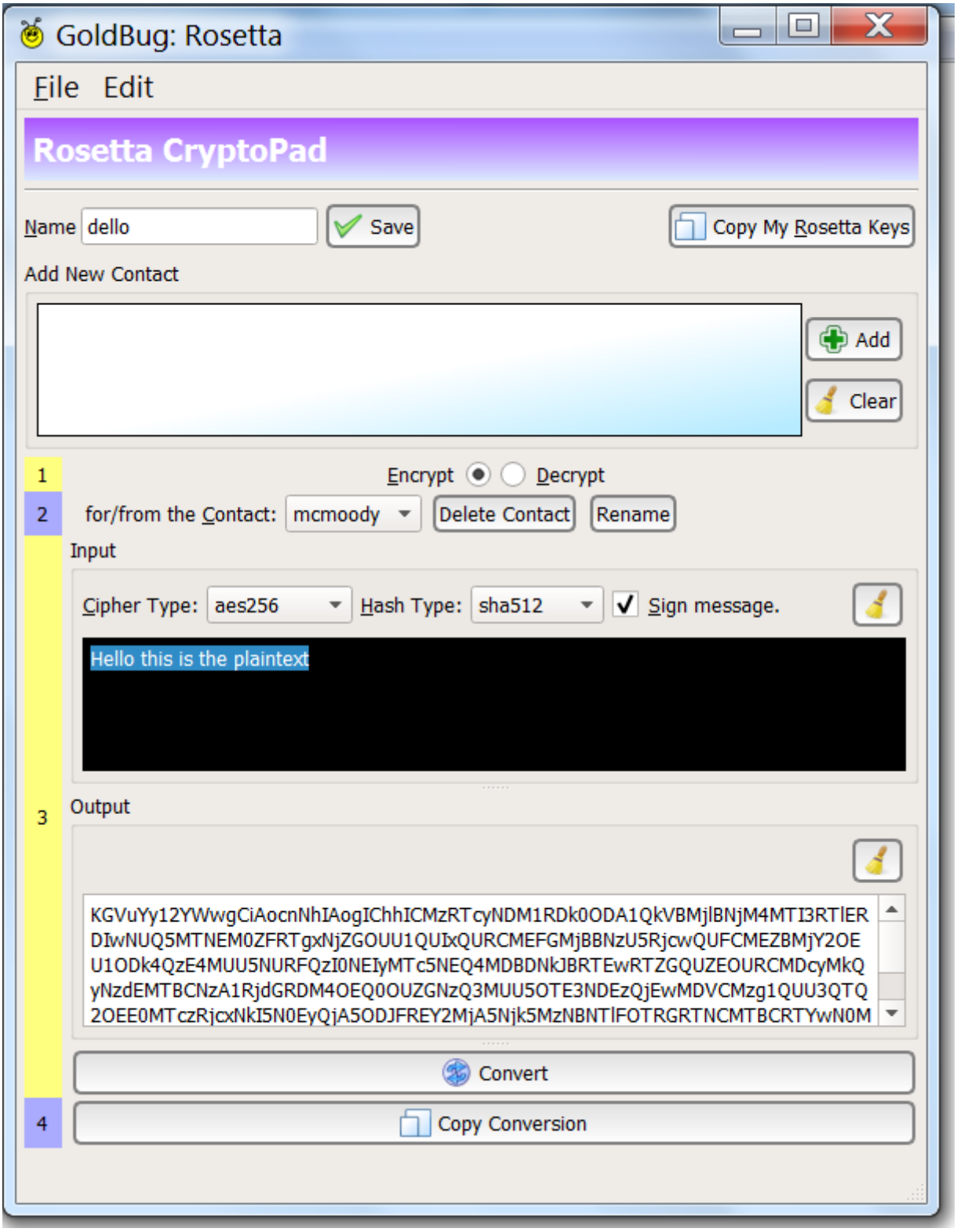
## 14.2 Werkzeug: Das Rosetta CryptoPad

Das Werkzeug Rosetta CryptoPad hat seinen Namen vom Stein von Rosett, der in London im Museum steht. Er gilt als Übersetzungshilfe für ägyptische Hyroglyphen in weitere Sprachen.

Das in GoldBug enthaltene Rosetta CryptoPad hat einen eigenen Schüssel - wie auch Chat und E-Mail und alle anderen Funktionen derartig ihre eigenen Schlüssel haben.

Der Nutzer tauscht auch hier mit einem Freund den Rosetta-Schlüssel, gibt dann Text in das CryptoPad ein, wähle den Freund und, ob es sich um Ver- oder Entschlüsselung handelt, - und drücke den Knopf “konverieren”.

Abbildung: Verschlüsselung vom Text mit dem Rosetta Crypto Pad




Sodann wird unten der Output als chiffrierter Text angezeigt und diesen kann der Nutzer einfach mit der Kopierfunktion auskopieren und über konventionelle Online-Kommunikationswege wie @-E-Mail oder einen anderen Chat versenden. Auch Web-Boards oder Paste-Bins kann der Nutzer so als Ort für seine verschlüsselte Kommunikation nutzen.

Es ist sozusagen „Slow-Chat“ durch eine manuelle Verschlüsselung des Chat-Textes (wobei wohl das Verschlüsseln schneller geht als das Copy Paste in andere Instanzen).

Das Rosetta CryptoPad ist eine Alternative zu GnuPG (bzw. basiert es ja ebenso auf der GnuPG zugrunde liegenden Bibliothek libgcrypt).

Diese Methode des Slow-Chats zeigt ebenso auf, dass Applikationen, die darauf setzten, jedes einzelne E-Mail zu verschlüsseln, eine unbequeme Methode sind. Wer will schon bei jedem E-Mail und jeder Chat-Nachricht den Empfänger auswählen, die Nachricht verschlüsseln, entscheiden, ob der Signatur-Schlüssel noch angefügt werden soll oder nicht, bevor die Nachricht versandt wird?

Abbildung: ROSETTA Crypto Pad Konversion




Bei GoldBug ist der generelle Vorteil gegeben, dass man einmalig mit dem Freund beim Setup den Schlüssel tauscht und sodann ist alles jederzeit verschlüsselt und die gesamte Kommunikation bewegt sich innerhalb der gewählten Verschlüsselung, die mit temporären Schlüsseln und Ende-zu-Ende Passphrasen (Geminis der Calling Funktion) jederzeit instant erneuert werden kann.

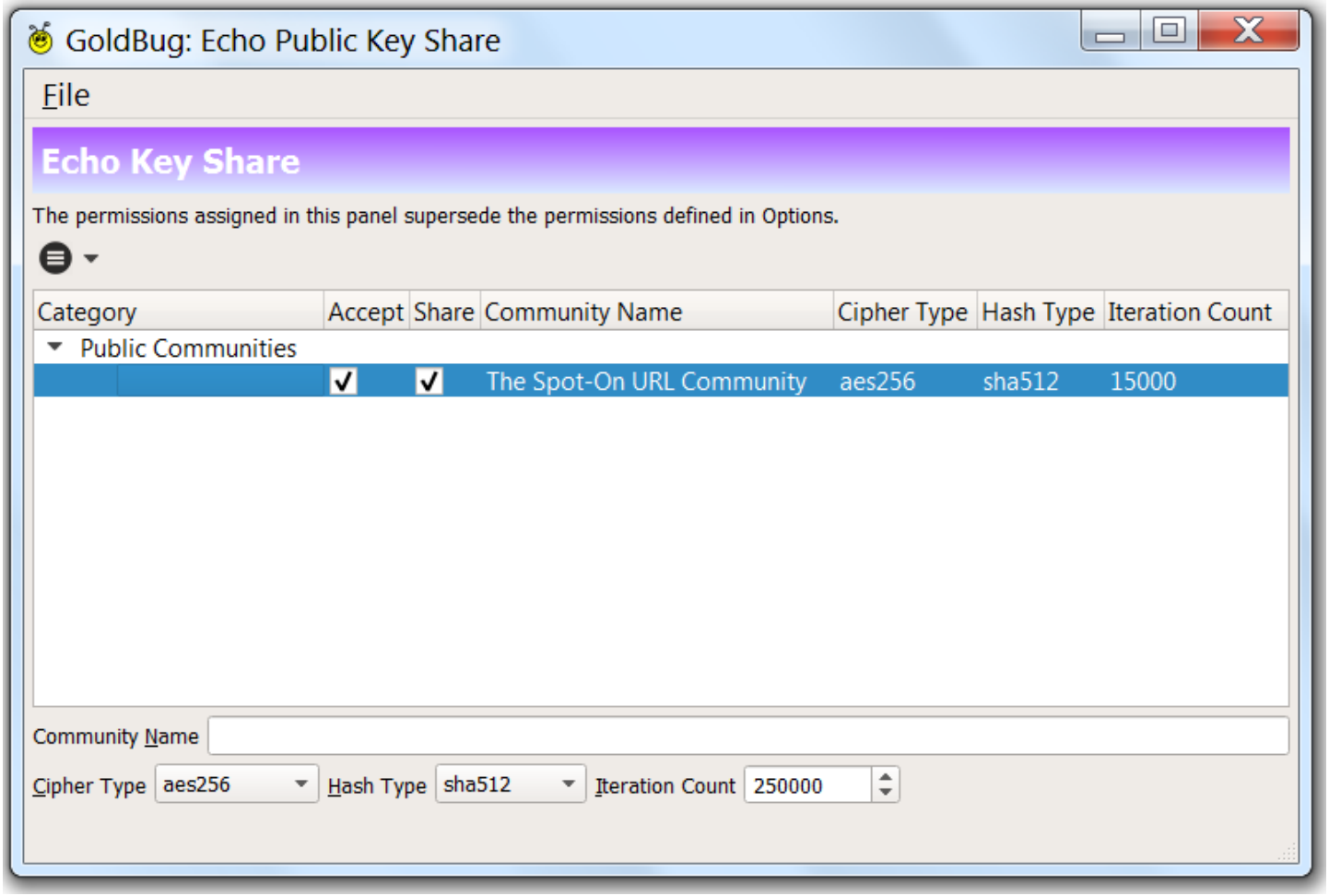
## 14.3 Werkzeug: Echo Public Key Share (EPKS)

Wenn es um Verschlüsselung geht, besteht immer die zentrale Fragestellung, wie man den Schlüssel sicher zum Freund transportieren kann.

Manche Architekturen nutzen dazu Key-Server, in denen der Nutzer seine öffentlichen Schlüssel einstellen kann. Dieses scheint logisch, schließlich ist es ein öffentlicher Schlüssel. Dennoch haben die Key-Server auch massive Nachteile, so weiß man nicht, ob man den richtigen Schlüssel darin gefunden hat bzw. ob dieser gar noch aktuell ist.

Mit der Echo Public Key Share Funktion können im GoldBug Messenger Schlüssel sehr einfach übertragen werden.

Abbildung: EPKS - Echo Public Key Sharing




Dazu wird mit einem Community Namen im p2p-Netzwerk des Echo-Protokolls ein symmetrischer Schlüssel definiert, über den alle Teilnehmer, die den Community-Namen kennen, sodann die öffentlichen Schlüssel tauschen können.

Das Werkzeug ist über das Hauptmenü verlinkt und öffnet ein neues Pop-Up-Fenster.

Ein Beispiel einer Community ist dort für den Tausch von URL-Schlüsseln bereits standardmäßig vorgegeben. Der Nutzer sendet seinen URL-Schlüssel an diese Community und alle weiteren Teilnehmer, die derzeit im p2p Netzwerk online sind, erhalten diesen Schlüssel.

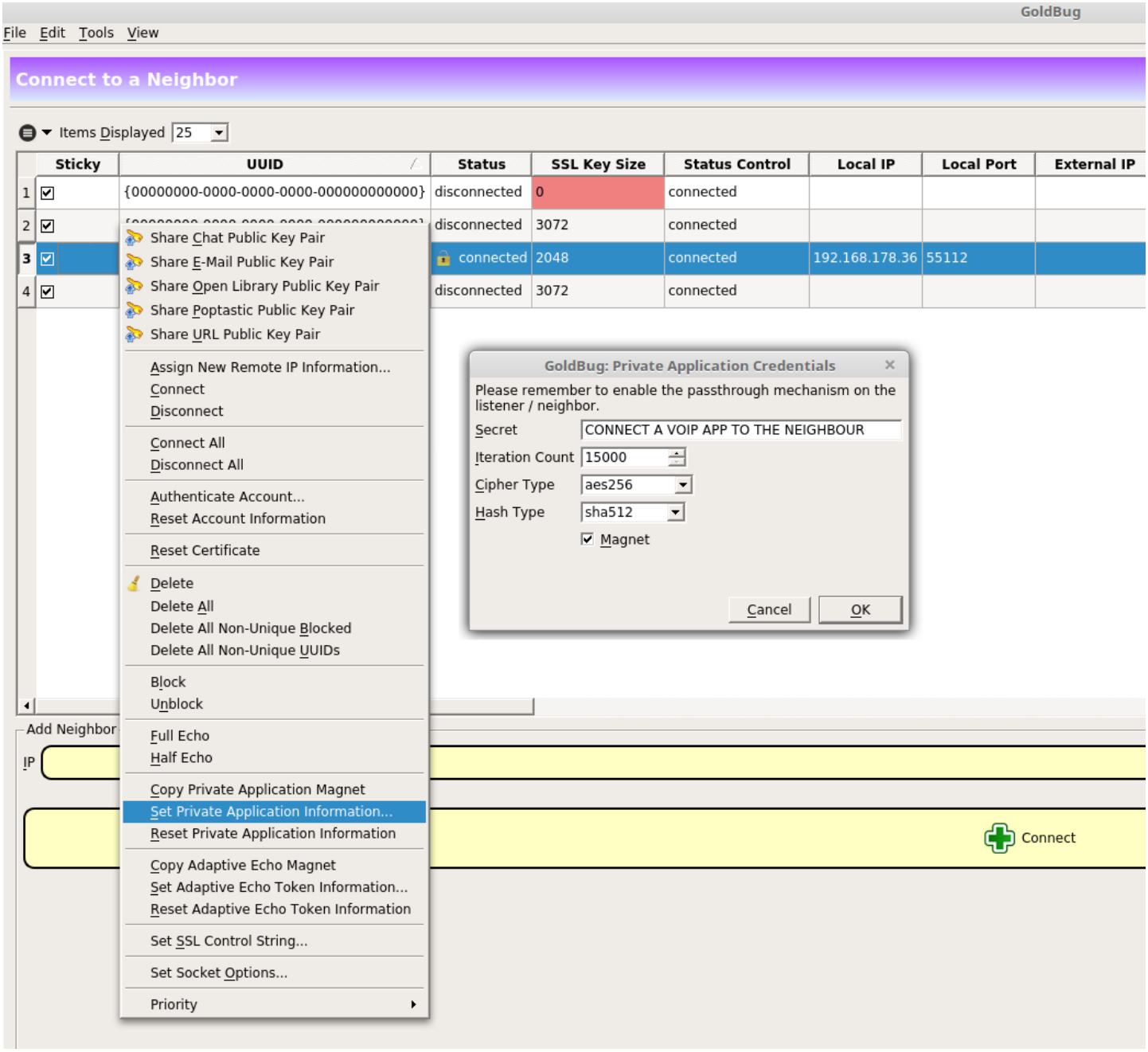
Es ist ein Schlüsselaustausch über einen symmetrisch verschlüsselten Kanal, wobei das Passwort für die Ende-zu-Ende-Verschlüsselung der Name der Community ist. Alle Nutzer, die den Namen der Community kennen, werden dann die Schlüssel, die Nutzer in den Kanal geben, erhalten und ihrem Programm hinzufügen können.

## 14.4 Pass-Through Funktionalität

Wenn zwei GoldBug Klienten über das Internet eine bestehende Verbindung haben, kann diese Verbindung als Tunnel genutzt werden, um die Daten einer anderen Applikation durch diesen Tunnel zu leiten.

Dazu wird der Proxy von GoldBug mit der Pass-Through Funktionalität adressiert.

Abbildung: GoldBug als Proxy: Pass-Through




Diese ist insofern eine interessante Funktion, um zwei Klienten eines anderen Programms ohne Verschlüsselung über das Internet mit der verschlüsselten Verbindung über GoldBug zu schützen.

Applikation => GB => GB-Server => GB => Applikation

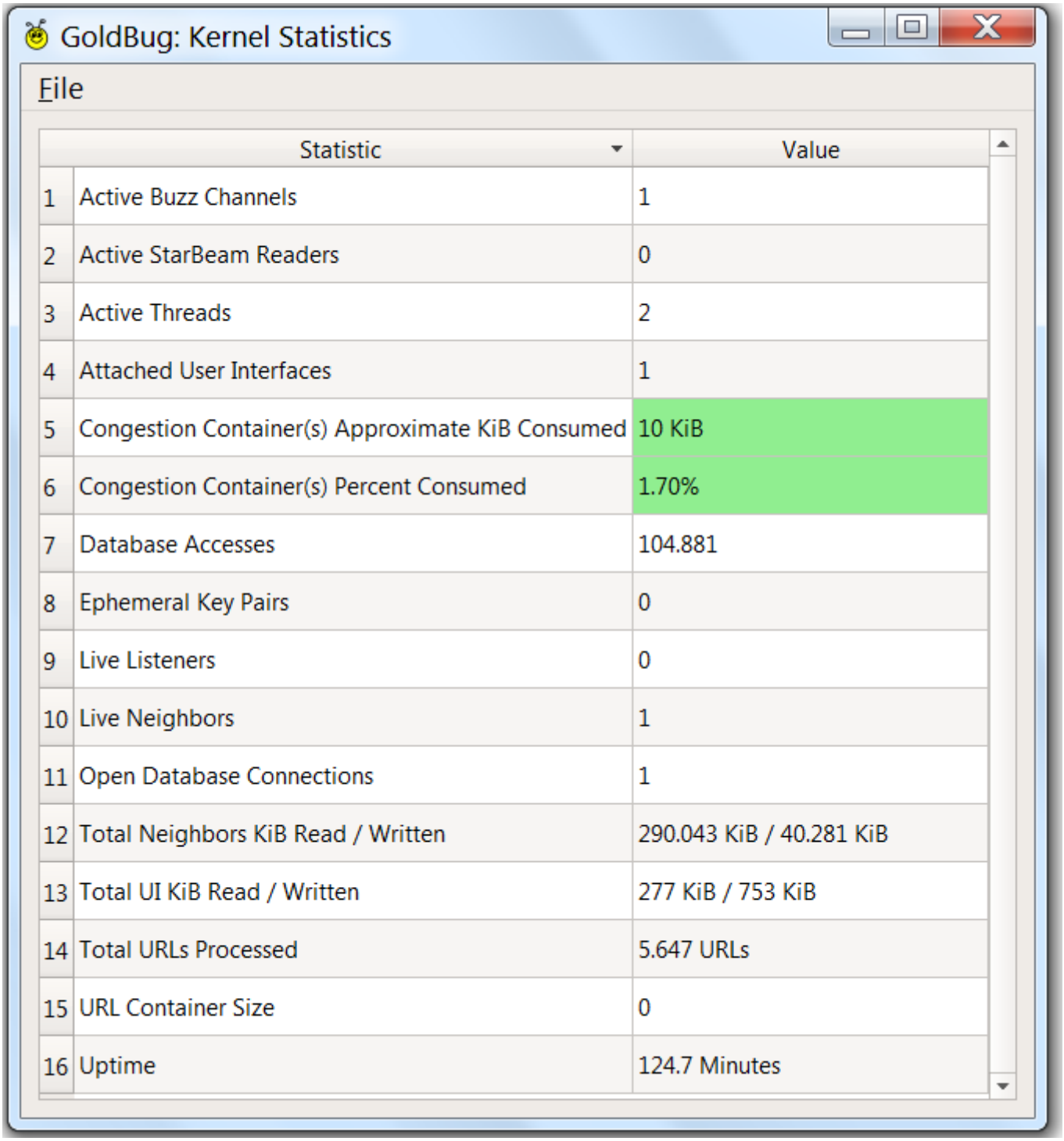
Es wird für eine andere Applikation sozusagen ein VPN-Tunnel aufgebaut, der sogar mit McEliece oder einen der anderen Verschlüsselungs-Algorithmen ausgestattet sein kann. Solange kein VPN Anbieter eine McEliece-Verschlüsselung von Startpunkt zu Endpunkt anbietet, ist die Pass-Through-Funktionalität, die GoldBug als eine Art VPN-Werkzeug darstellt, die richtige Wahl für einen Test. Bislang ist keine weitere Tunnel-Software bekannt, die McEliece einsetzt.

Die anzubindende Applikation sollte dabei tolerant gegenüber der Reihenfolge der gesandten Pakete sein. Ein interessanter Forschungs-Test mit mehreren möglichen Applikationen, die durch geleitet werden können.

## 14.5 Anzeige Analysen und Statistiken

Neben Statistik-Übersichten sind auch Analyse-Werkzeuge in GoldBug enthalten, wie z.B. der oben angesprochen StarBeam-Analyzer. Auch die Listener und Server Tabellen enthalten zahlreiche Dateninformationen zu gesandten Paketen, ebenso bestehen die Statistiken für die URL-Datenbank.

Abbildung: Anzeige von Statistiken




| | Statistic | Value |
|---|---|---|
| 1 | Active Buzz Channels | 1 |
| 2 | Active StarBeam Readers | 0 |
| 3 | Active Threads | 2 |
| 4 | Attached User Interfaces | 1 |
| 5 | Congestion Container(s) Approximate KiB Consumed | 10 KiB |
| 6 | Congestion Container(s) Percent Consumed | 1.70% |
| 7 | Database Accesses | 104.881 |
| 8 | Ephemeral Key Pairs | 0 |
| 9 | Live Listeners | 0 |
| 10 | Live Neighbors | 1 |
| 11 | Open Database Connections | 1 |
| 12 | Total Neighbors KiB Read / Written | 290.043 KiB / 40.281 KiB |
| 13 | Total UI KiB Read / Written | 277 KiB / 753 KiB |
| 14 | Total URLs Processed | 5.647 URLs |
| 15 | URL Container Size | 0 |
| 16 | Uptime | 124.7 Minutes |

Neben der üblichen Benutzeroberfläche lässt sich GoldBug auch in Konsolenform z.B. auf einem RasperryPi installieren und die Statistik-Übersicht mit einem entsprechenden Befehl abrufen.

Abbildung: Statistik Konsole auf einem Raspberry Pi

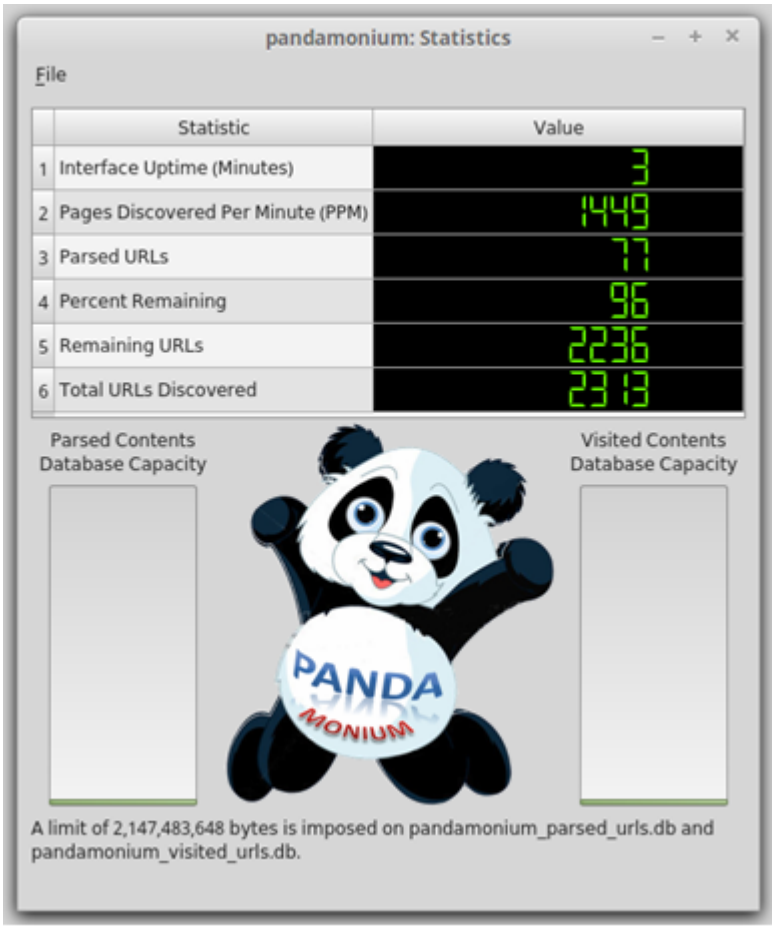
**GoldBug Messenger on the Raspberry PI - Statistics over Console**

```
pi@snoopy:~ $ uname -a
Linux snoopy 4.1.13-v7+ #826 SMP PREEMPT Fri Nov 13 20:19:03 GMT 2015 armv7l GNU/Linux
pi@snoopy:~ $ sqlite3 .spot-on/kernel.db
SQLite version 3.8.7.1 2014-10-29 13:59:56
Enter ".help" for usage hints.
sqlite> .d
PRAGMA foreign_keys=OFF;
BEGIN TRANSACTION;
CREATE TABLE kernel_gui_server (port INTEGER PRIMARY KEY NOT NULL CHECK (port >= 0 AND port <= 65535));
INSERT INTO "kernel_gui_server" VALUES(38328);
CREATE TABLE kernel_statistics (statistic TEXT PRIMARY KEY NOT NULL, value TEXT);
INSERT INTO "kernel_statistics" VALUES('Active Buzz Channels','0');
INSERT INTO "kernel_statistics" VALUES('Active StarBeam Readers','0');
INSERT INTO "kernel_statistics" VALUES('Active Threads','1');
INSERT INTO "kernel_statistics" VALUES('Attached User Interfaces','0');
INSERT INTO "kernel_statistics" VALUES('Congestion Container(s) Approximate MiB Consumed','2 MiB');
INSERT INTO "kernel_statistics" VALUES('Congestion Container(s) Percent Consumed','0.46%');
INSERT INTO "kernel_statistics" VALUES('Database Accesses','465,717');
INSERT INTO "kernel_statistics" VALUES('Ephemeral Key Pairs','0');
INSERT INTO "kernel_statistics" VALUES('Live Listeners','1');
INSERT INTO "kernel_statistics" VALUES('Live Neighbors','3');
INSERT INTO "kernel_statistics" VALUES('Open Database Connections','1');
INSERT INTO "kernel_statistics" VALUES('Total URLs Processed','0 URLs');
INSERT INTO "kernel_statistics" VALUES('URL Container Size','0');
INSERT INTO "kernel_statistics" VALUES('Uptime','875.2 Minutes');
CREATE TRIGGER kernel_gui_server_trigger BEFORE INSERT ON kernel_gui_server BEGIN DELETE FROM kernel_gui_server; END;
COMMIT;
```



Auch der Pandamonium URL Webcrawler verfügt über entsprechende statistische Angaben.

Abbildung: Pandamonium Web Crawler Stats




## 15 BIG SEVEN STUDY: Crypto-Messenger-Audit

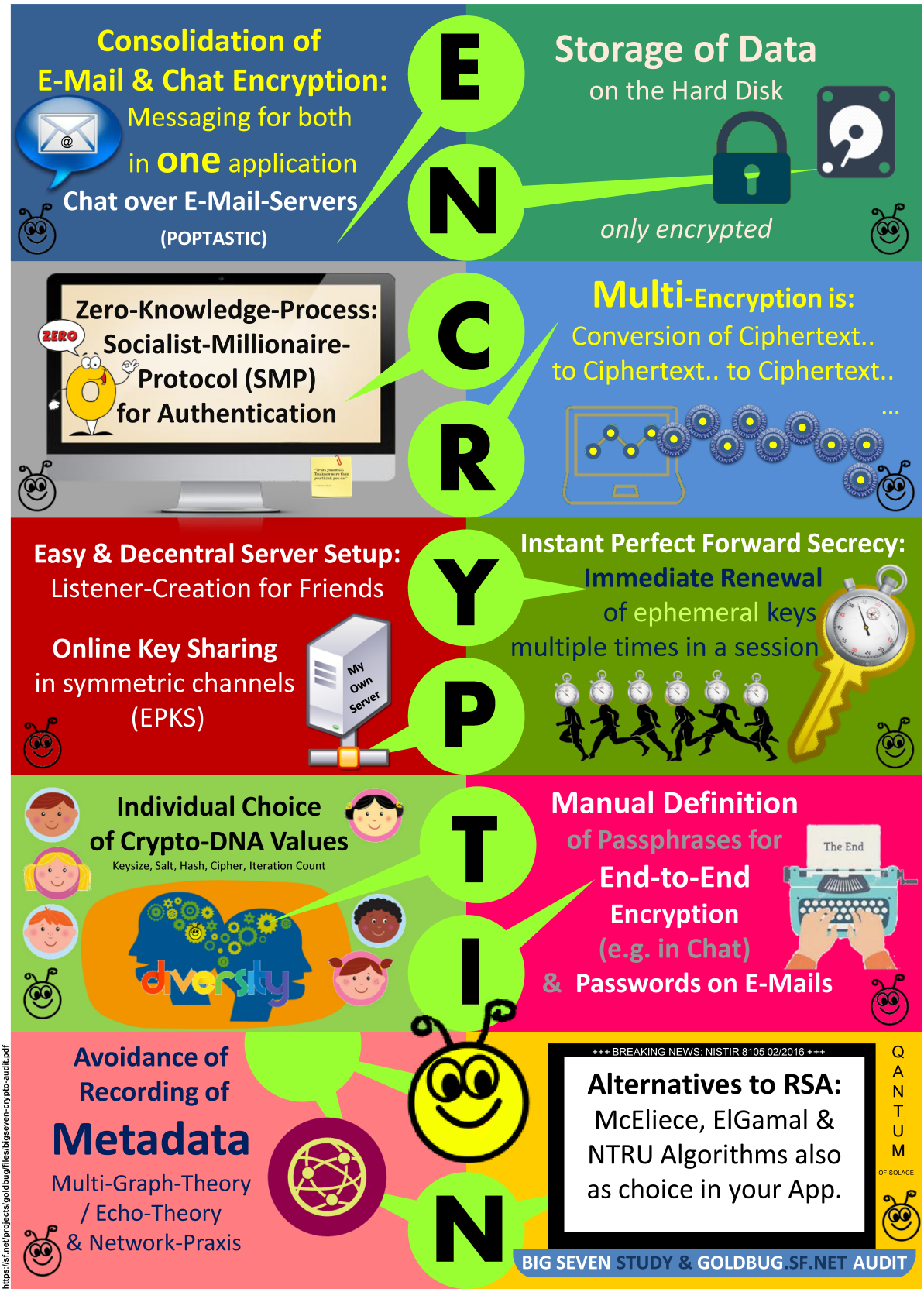
GoldBug wurde in dem Audit von David Adams und Ann-Kathrin Maier über sieben quelloffene Messenger als einer der BIG SEVEN Crypto-Messenger betrachtet, der in einem an internationalen IT-Audit Manualen orientierten Audit sich in mehr als 20 Dimensionen als sehr audit-konform und vertrauenswürdig herausstelle.

Auch die zahlreichen Code-Reviews gaben Hinweise auf eine exzellente Programmierung. Die über alle sieben Messenger identifizierten "'10 Trends in Crypto-Messaging'" (vgl. Adams / Maier 2016) betreffen insbesondere die Möglichkeit in GoldBug, dezentrale Chat-Server einzurichten, sowie auch manuelle Definitionen von Ende-zu-Ende verschlüsselnden symmetrischen Passworten zu treffen.

Abbildung: Trends in Crypto nach der Big-Seven-Crypto-Studie (2016)

# 10 Trends in **Crypto Messaging**

A Study on the open source Applications GoldBug, CryptoCat, OTR+XMPP, RetroShare, Signal, Surespot and Tox.




**Adams, D. / Maier, A.K. (2016)**

Neben der Auditierung des Quellcodes, der Architektur und Prozesse der Verschlüsselung sowie der Funktionen in GoldBug bestehen zahleiche weitere Themenfelder, an denen eine zukünftige

Forschung sich orientieren kann. Exemplarisch wären für weitere Evaluationen und Forschungsbedarf folgende Fragen zu nennen, die insbesondere im Vergleich mit anderen Prozessen und Applikationen eine Rolle spielen können:

- Is the Application open source?
- Is it a tiered application: kernel and user interface processes.
- Are there proxy capabilities?
- Is it possible to send E-Mail messages to offline friends?
- Is it possible to send E-Mail with encrypted attachments?
- Are there different Keys for different functions in place like Chat, E-Mail, Cryptopad, Filetransfer etc.?
- Is the key stuck to the IP Address of the user?
- How is mutual access authentication defined?
- Are there alternatives to RSA, like Elgamal or NTRU? Can a NTRU-user chat to a RSA-user? Is McEliece given?
- Are there selectable SSL ciphers?
- Are there selectable hash algorithms?
- Just need connectivity, no key exchange, keys are optional?
- Is trust needed, or can it be added as the user define it?
- What about technical simplicity?
- is it possible to determine, who is reading which message?
- Local databases store all info in encrypted .db's?
- Is the authentication of messages optional?
- Can the user communicate without public keys, using Magnets?
- Support for TCP and UDP and SCTP communications?
- Support of multi-layers of encryption
- Are multiple listeners possible?
- Is a multi-threaded kernel given?
- Are there IRC-like group chat channels?
- What about simple IP-based firewalls?
- Do scramblers send out fake messages?
- Is it possible to store messages in friends?
- Is there the option to use an individually defined and manually inserted end-to-end key for communication?
- Is there the option to renew the end-to-end key each time user wants (not only session based)?
- Encrypted file transfer protocol (StarBeam)? -Using a onetime magnet (OTM) for a crypto channel?
- Having ipv6 support?
- Having Qt 5 and up deployed?
- Sending a message to a friend to his dedicated connection and not to all connections?
- Hiding the key exchange online?
- Use several encryption keys for one file transfer?

- Adding a passphrase on a file transfer?

# 16 Die digitale Verschlüsselung der privaten Kommunikation im Kontext von...

Dieses Benutzerhandbuch soll nicht nur technisch die Handhabung von Verschlüsslungen, ihren Prozessen oder die Nutzung der einzelnen Tabs und Knöpfe darstellen, sondern auch den Sinn der Verschlüsselung darstellen, wie er sich im Lichte verschiedener Grundgesetze zum Schutz der privaten Freiheit und Kommunikation darstellt. Auf folgende grundlegende Gesetze soll daher hingewiesen werden, die in ihren Originaltexten Referenz dazu nehmen.

## 16.1 Principles of the protection of private speech, communication and life: Universal Declaration of Human Rights, 1948 (Art. 12)

No one shall be subjected to arbitrary interference with his privacy, family, home or correspondence, nor to attacks upon his honour and reputation. Everyone has the right to the protection of the law against such interference or attacks.
http://www.un.org/en/documents/udhr/index.shtml#a12 Universal Declaration of Human Rights Universal Declaration of Human Rights

## 16.2 International Covenant on Civil and Political Rights, 1966 (Art. 17)

1. No one shall be subjected to arbitrary or unlawful interference with his privacy, family, home or correspondence, nor to unlawful attacks on his honour and reputation.
2. Everyone has the right to the protection of the law against such interference or attacks.
   http://www.ohchr.org/EN/ProfessionalInterest/Pages/CCPR.aspx International Covenant on Civil and Political Rights - International Covenant on Civil and Political Rights

## 16.3 European Convention on Human Rights, 1950 (Art. 8)

1.Everyone has the right to respect for his private and family life, his home and his correspondence. 2.There shall be no interference by a public authority with the exercise of this right except such as is in accordance with the law and is necessary in a democratic society in the interests of national security, public safety or the economic well-being of the country, for the prevention of disorder or crime, for the protection of health or morals, or for the protection of the rights and freedoms of others. http://conventions.coe.int/treaty/en/Treaties/Html/005.htm European Convention on Human Rights - European Convention on Human Rights

## 16.4 Charter of Fundamental Rights of the European Union, 2000 (Art. 7, 8)

"'Article 7 - Respect for private and family life'" Everyone has the right to respect for his or her private and family life, home and communications.

"'Article 8. Protection of personal data'" 1.Everyone has the right to the protection of personal data concerning him or her. 2.Such data must be processed fairly for specified purposes and on the basis of the consent of the person concerned or some other legitimate basis laid down by law. Everyone has the right of access to data which has been collected concerning him or her, and the right to have it rectified. 3.Compliance with these rules shall be subject to control by an independent authority. Charter of Fundamental Rights of the European Union - Charter of Fundamental Rights of the European Union (Wikisource) Charter of Fundamental Rights of the European Union - Charter of Fundamental Rights of the European Union

## 16.5 Basic Law e.g. for the Federal Republic of Germany, 1949 (Art. 2 Abs. 1 i. V. m. Art. 1 Abs. 1)

"'Article 2 - Personal freedoms'" (1) Every person shall have the right to free development of his personality insofar as he does not violate the rights of others or offend against the constitutional order or the moral law. Article 1 [Human dignity – Human rights – Legally binding force of basic rights] (1) Human dignity shall be inviolable. To respect and protect it shall be the duty of all state authority. https://www.btg-bestellservice.de/pdf/80201000.pdf Basic Law for the Federal Republic of Germany - Basic Law for the Federal Republic of Germany

"'Further: Article 1 and Article 10:'"

Art. 1 Human dignity – Human rights: Legally binding force of basic rights (1) Human dignity shall be inviolable. To respect and protect it shall be the duty of all state authority. (2) The German people therefore acknowledge inviolable and inalienable human rights as the basis of every community, of peace and of justice in the world. (3) The following basic rights shall bind the legislature, the executive and the judiciary as directly applicable law

## 16.6 Art. 10 - Privacy of correspondence, posts and telecommunications

Secrecy of correspondence - Fernmeldegeheimnis (Art. 10 Abs. 1 Grundgesetz) § 88 Abs. 1 Fernmeldegeheimnis - Telekommunikationsgesetz: (1) Dem Fernmeldegeheimnis unterliegen der Inhalt der Telekommunikation und ihre näheren Umstände, insbesondere die Tatsache, ob jemand an einem Telekommunikationsvorgang beteiligt ist oder war. Das Fernmeldegeheimnis erstreckt sich auch auf die näheren Umstände erfolgloser Verbindungsversuche. (2) Zur Wahrung des Fernmeldegeheimnisses ist jeder Diensteanbieter verpflichtet. Die Pflicht zur Geheimhaltung besteht auch nach dem Ende der Tätigkeit fort, durch die sie begründet worden ist. (3) Den nach Absatz 2 Verpflichteten ist es untersagt, sich oder anderen über das für die geschäftsmäßige Erbringung der Telekommunikationsdienste einschließlich des Schutzes ihrer technischen Systeme erforderliche Maß hinaus Kenntnis vom Inhalt oder den näheren Umständen der Telekommunikation zu verschaffen. Sie dürfen Kenntnisse über Tatsachen, die dem Fernmeldegeheimnis unterliegen, nur für den in Satz 1 genannten Zweck verwenden. Eine

Verwendung dieser Kenntnisse für andere Zwecke, insbesondere die Weitergabe an andere, ist nur zulässig, soweit dieses Gesetz oder eine andere gesetzliche Vorschrift dies vorsieht und sich dabei ausdrücklich auf Telekommunikationsvorgänge bezieht. Die Anzeigepflicht nach § 138 des Strafgesetzbuches hat Vorrang. (4) Befindet sich die Telekommunikationsanlage an Bord eines Wasser- oder Luftfahrzeugs, so besteht die Pflicht zur Wahrung des Geheimnisses nicht gegenüber der Person, die das Fahrzeug führt oder gegenüber ihrer Stellvertretung.

## 16.7 § 206 - Verletzung des Post- oder Fernmeldegeheimnisses

(1) Wer unbefugt einer anderen Person eine Mitteilung über Tatsachen macht, die dem Post- oder Fernmeldegeheimnis unterliegen und die ihm als Inhaber oder Beschäftigtem eines Unternehmens bekanntgeworden sind, das geschäftsmäßig Post- oder Telekommunikationsdienste erbringt, wird mit Freiheitsstrafe bis zu fünf Jahren oder mit Geldstrafe bestraft. (2) Ebenso wird bestraft, wer als Inhaber oder Beschäftigter eines in Absatz 1 bezeichneten Unternehmens unbefugt 1. eine Sendung, die einem solchen Unternehmen zur Übermittlung anvertraut worden und verschlossen ist, öffnet oder sich von ihrem Inhalt ohne Öffnung des Verschlusses unter Anwendung technischer Mittel Kenntnis verschafft, 2. eine einem solchen Unternehmen zur Übermittlung anvertraute Sendung unterdrückt oder 3. eine der in Absatz 1 oder in Nummer 1 oder 2 bezeichneten Handlungen gestattet oder fördert. (3) Die Absätze 1 und 2 gelten auch für Personen, die 1. Aufgaben der Aufsicht über ein in Absatz 1 bezeichnetes Unternehmen wahrnehmen, 2. von einem solchen Unternehmen oder mit dessen Ermächtigung mit dem Erbringen von Post- oder Telekommunikationsdiensten betraut sind oder 3. mit der Herstellung einer dem Betrieb eines solchen Unternehmens dienenden Anlage oder mit Arbeiten daran betraut sind. (4) Wer unbefugt einer anderen Person eine Mitteilung über Tatsachen macht, die ihm als außerhalb des Post- oder Telekommunikationsbereichs tätigem Amtsträger auf Grund eines befugten oder unbefugten Eingriffs in das Post- oder Fernmeldegeheimnis bekanntgeworden sind, wird mit Freiheitsstrafe bis zu zwei Jahren oder mit Geldstrafe bestraft. (5) Dem Postgeheimnis unterliegen die näheren Umstände des Postverkehrs bestimmter Personen sowie der Inhalt von Postsendungen. Dem Fernmeldegeheimnis unterliegen der Inhalt der Telekommunikation und ihre näheren Umstände, insbesondere die Tatsache, ob jemand an einem Telekommunikationsvorgang beteiligt ist oder war. Das Fernmeldegeheimnis erstreckt sich auch auf die näheren Umstände erfolgloser Verbindungsversuche.

- http://www.gesetze-im-internet.de/gg/art_10.html
- Secrecy of correspondence - Secrecy of correspondence
- Briefgeheimnis - Briefgeheimnis
- (Fernmeldegeheimnis - Fernmeldegeheimnis]()
- Postgeheimnis - Postgeheimnis
- http://www.gesetze-im-internet.de/tkg_2004/__88.html
- http://www.gesetze-im-internet.de/stgb/__206.html

### 16.8 United States Constitution: Search and Seizure (Expectation of Privacy, US Supreme Court)

The right of the people to be secure in their persons, houses, papers, and effects, against unreasonable searches and seizures, shall not be violated, and no Warrants shall issue, but upon probable cause, supported by Oath or affirmation, and particularly describing the place to be searched, and the persons or things to be seized. http://www.usconstitution.net/const.html

## 17 Historie der Programm-Veröffentlichungen

Die Liste der Veröffentlichungen zeigt über mehrere Jahre hinweg kontinuierliche Veröffentlichungen der Applikation. Die erste Veröffentlichung geht auf das Jahr 2013 zurück, zuvor sind zudem in einem anderen Projekt ebenso mehrere Jahre Forschungsarbeit eingeflossen. Die Veröffentlichungsdaten der Versionen zeigt im Durchschnitt fast monatlich eine Release-Veröffentlichung. Die entsprechende Anmerkung verdeutlicht, welche Funktion im Wesentlichen hinzugefügt, verbessert oder neu veröffentlicht wurde.

Die Historie der Veröffentlichungen seit 2013 und früher findet sich mit bislang rd. 40 Programm-Veröffentlichen hier im Wiki der Projektseite:
https://sourceforge.net/p/goldbug/wiki/release-history/

## 18 Webseite

Weitere Informationen finden sich auf der Webseite:

- http://GoldBug.sf.net

## 19 Offener Quellcode=

Der offene Quellcode findet sich bei GitHub:

- https://github.com/textbrowser/spot-on

### 19.1 Informationen zur Kompilierung

Wer auf der Webseite von GoldBug schaut, findet hier jeweils die aktuelle Veröffentlichung, insbesondere für Windows. Wer fortgeschrittenere Computer-Kenntnisse hat, ein Programm auch selbst vom Quellcode kompilieren möchte oder es an diesem Beispiel erlernen will, findet hier die Hinweise, wie dabei vorzugehen ist für das Betriebssystem Windows.

Die Kompilierung aus dem Quellcode ermöglicht es dem Nutzer, zu sehen, wie der Quelltext sich in eine Binärdatei (.exe) bildet und welche Programmbibliotheken zu ergänzen sind, damit die ausführbare Datei laufen kann.

1. Zunächst ist das Qt-Tool-Kit herunter zu laden. Zu wählen wäre die Offline (oder Online) Installation von Qt mit MingGW: z.B. Qt 5.X for Windows 32-bit (MinGW 4.9.2, 1.0 GB) unter der URL: http://www.qt.io/download-open-source/#section-2

2. Sodann ist der Quelltext herunterzuladen. Für Windows sind alle benötigten Abhängigkeiten und Programmbibliotheken bereits integriert im Pfad des Qelltextes. Die GoldBug Gui und den Spot-On Kernel findet der Nutzer bei GitHub unter dieser URL: https://github.com/textbrowser/spot-on Um den Quelltext herunterzuladen kann der Nutzer auf der Webseite den Master-Tree als Zip im Browser herunterladen oder einen GIT-Klienten (Windows) für den Source-Download verwenden.

Für Linux sind alle diese Programmbibliotheken zu iinstallieren:

- Qt 5.1.x or higher,
- libGeoIP 1.5.1,
- libcrypto 0.9.8 or later,
- libgcrypt 1.5.x, and
- libssl 0.9.8 or later.
- libsqlite3-dev
- libgcrypt11-dev
- libssl-dev
- libgeoip-dev
- libpq-dev,
- libeay,
- libgpg-error,
- libsshgcrypt-dev,
- libssh-gcrypt-dev,
- libgcrypt-dev,
- libgcrypt11-dev,
- libgl1-mesa-dev,
- libcurlpp-dev,
- libcurl4openssl-dev,
- libsctp-dev,
- libtool,
- libtool-dev,
- libntl,

Die libGeoIP Programmbibliothek ist optional und kann auch umgangen werden, wenn die ausgewählte Qt-PRO-Projektdatei entsprechend konfiguriert wird. Es ist zu prüfen, ob für Linux alle genannten oder aktuelleren Versionen dieser Programmbibliotheken auf der Maschine installiert sind. Für Windows sind wie gesagt die nötigen Programmbibliotheken dem Quellcode bereits beigefügt (DLL-Dateien). Weitere Compiling Informationen finden sich hier: https://sourceforge.net/p/goldbug/wiki/compiling/

1. Nachdem der Nutzer Qt installiert hat, ist das Programm Qt-Creator aus dem Qt-Verzeichnis zu starten.

2. Sodann ist aus dem entpackten Quellcode-Pfad die relevante .pro Datei auszuwählen und die GUI und den Kernel mit Qt Creator zu kompilieren. Für die Kompilierung von GoldBug installiert man also Qt5 und wählt dann die .pro Datei „GoldBug.Qt5.win.pro“. Diese Datei öffnet beide Unter-Pro-Dateien für Kernel und Gui.

Sodann ist in QT-Creator einfach der grüne Forward-Pfeil zu klicken und die Kompilierung startet. Am Ende des Kompilierungsprozesses aus dem Qt-Creator sollte dann die GoldBug.exe startbar sein. Wenn der Nutzer die exe.Datei in einen eigenen Pfad auf seine Festplatte geben will, muss er auch alle benötigten DLL-Dateien (aus der gewählten Qt Version und von allen aktuellen Bibilotheken) hinzufügen sowie die Unterpfade z.B. für die Sound- oder Qt-Dateien, wie sie eben in dem vorgegeben Installations-Zip für GoldBug Windows bereits vorhanden sind. Die Bibliotheks-DLL-Dateienfür Window sind auch im source code der jeweiligen Bibliothekspfade mit abgespeichert zur passenden und einfachen Nutzung.

Der Nutzer kann mit dem Qt-Terminal-Fenster GoldBug natürlich auch mit manuellen DOS-Befehlen kompilieren, ohne Qt-Creator zu nutzen.

KOMPILING-PROZESS mit C++/Qt:

- Source: https://github.com/textbrowser/spot-on

'"Windows:"' qmake -o Makefile GoldBug.win.qt5.pro
make or mingw32-make
or choose in Qt-Creator: GoldBug.win.qt5.pro

GB does not provide checksums for the binary downloads as the source is given for those who want to build on their own. Please notice: GB has a build date in the gui so the sums might differ for each compile!

'"FURTHER INFO for other .pro files: "'

If header (h) or interface (ui) files have changed, please perform a distclean before building the application.

'"Absolute cleaning:"' make distclean or mingw32-make distclean

'"FreeBSD:"' qmake -o Makefile spot-on.freebsd.pro
make

'"Linux:"' qmake -o Makefile spot-on.pro
make

'"OS X:"' qmake -spec macx-g++ -o Makefile spot-on.osx.pro
make

”"Windows:”” qmake -o Makefile spot-on.win.pro
make or mingw32-make

# 20 Schriftenverzeichnis


- Adams, David / Maier, Ann-Kathrin (2016): BIG SEVEN Study, open source crypto-messengers to be compared - or: Comprehensive Confidentiality Review & Audit of GoldBug, Encrypting E-Mail-Client & Secure Instant Messenger, Descriptions, tests and analysis reviews of 20 functions of the application GoldBug based on the essential fields and methods of evaluation of the 8 major international audit manuals for IT security investigations including 38 figures and 87 tables., URL: https://sf.net/projects/GoldBug/files/bigseven-crypto-audit.pdf - English / German Language, Version 1.1, 305 pages, June 2016

- Arbeitskreis Vorratsdatenspeicherung (AKV), Bündnis gegen Überwachung et al.: List of Secure Instant Messengers, URL: http://wiki.vorratsdatenspeicherung.de/List_of_Secure_Instant_Messengers, Mai 2014.

- Banerjee, Sanchari: EFYTIMES News Network: 25 Best Open Source Projects Of 2014: Efytimes ranked GoldBug Messenger # 4 on the overall Top 25 Best Open Source Projects Of 2014, http://www.efytimes.com/e1/fullnews.asp?edid=148831

- Cakra, Deden: Review of GoldBug Instant Messenger, Blogspot, 13 Desember 2014, http://bengkelcakra.blogspot.de/2014/12/free-download-GoldBug-instant-messenger.html

- Constantinos / OsArena: GoldBug: ΜΙΑ ΣΟΥΙΤΑ ΓΙΑ CHATING ΜΕ ΠΟΛΛΑΠΛΗ ΚΡΥΠΤΟΓΡΑΦΗΣΗ, Latest Articles, 25 March 2014, http://osarena.net/logismiko/applications/GoldBug-mia-souita-gia-chating-me-pollapli-kriptografisi.html

- Demir, Yigit Ekim: Güvenli ve Hizli Anlik Mesajlasma Programi: GoldBug Instant Messenger programi, bu sorunun üstesinden gelmek isteyen kullanicilar için en iyi çözümlerden birisi haline geliyor ve en güvenli sekilde anlik mesajlar gönderebilmenize imkan taniyor (Translated: “GoldBug Instant Messenger Application is the best solution for users, who want to use one of the most secure ways to send instant messages”), News Portal Tamindir http://www.tamindir.com/GoldBug-instant-messenger/

- Dragomir, Mircea: GoldBug Instant Messenger - Softpedia Review: This is a secure P2P Instant Messenger that ensures private communication based on a multi encryption technology constituted of several security layers, http://www.softpedia.com/get/Internet/Chat/Instant-Messaging/GoldBug-Instant-Messenger.shtml, Softpedia Review, January 31st, 2016

- Edwards, Scott: Manual of the GoldBug Crypto Messenger, 2018, Review at Github, URL: https://compendio.github.io/goldbug-manual-de/ & https://compendio.github.io/goldbug-

manual

- Filecluster: GoldBug Instant Messenger - Un programme très pratique et fiable, conçu pour créer un pont de communication sécurisé entre deux ou plusieurs utilisateurs, URL: https://www.filecluster.fr/logiciel/GoldBug-Instant-Messenger-174185.html
- Fousoft: GoldBug Instant Messenger, URL: https://www.fousoft.com/goldbug-instant-messenger.html, March 16, 2017
- Generation NT: Sécuriser ses échanges par messagerie: Apportez encore plus de la confidentialité dans votre messagerie, URL: https://www.generation-nt.com/goldbug-messenger-securiser-echanger-communiquer-discuter-messagerie-securite-echange-communication-telecharger-telechargement-1907585.html
- Hartshorn, Sarah: 3 New Open Source Secure Communication Projects, May 28, 2015, http://blog.vuze.com/2015/05/28/3-new-open-source-secure-communication-projects/
- Harvey, Cynthia: Datamation: 50 Noteworthy Open Source Projects - Chapter Secure Communication: GoldBug Messenger ranked on first # 1 position, Posted September 19, 2014, http://www.datamation.com/open-source/50-noteworthy-new-open-source-projects-3.html
- Heise: GoldBug kann Schlüssel selbst encodiert versenden, URL: http://www.heise.de/download/goldbug-1192605.html
- Joos, Thomas: Sicheres Messaging im Web, PCWelt Magazin, Mittwoch den 01.10.2014, http://www.pcwelt.de/ratgeber/Tor__I2p__Gnunet__RetroShare__Freenet__GoldBug__Spurlos_im_Web-Anonymisierungsnetzwerke-8921663.html
- Lindner, Mirko: POPTASTIC: Verschlüsselter Chat über POP3 mit dem GoldBug Messenger, Pro-Linux, 9. Dezember 2014, http://www.pro-linux.de/news/1/21822/POPTASTIC-verschluesselter-chat-ueber-pop3.html
- Majorgeeks:GoldBug Secure Email Client & Instant Messenger, URL:http://www.majorgeeks.com/files/details/goldbug_secure_email_client_instant_messenger.html
- Momedo: Open Source Mobiler Messenger für kommunale und schulische Zwecke mit Verschlüsselung, Github, URL: https://momedo.github.io/momedo/ & https://github.com/momedo/momedo/blob/master/README.md , 2018
- Por, Julianna Isabele: Segurança em primeiro lugar, URL: https://www.baixaki.com.br/download/goldbug.htm
- Qt Digia: Qt Digia has awarded GoldBug IM as reference project for Qt implementation in the official Qt-Showroom of Digia: showroom.qt-project.org/goldbug/, 2015

- Sabtu: Free GoldBug Instant Messenger 1.7, URL: http://bengkelcakra.blogspot.de/2014/12/free-download-goldbug-instant-messenger.html, 13 December 2014

- Schneier, Bruce / Seidel, Kathleen / Vijayakumar, Saranya: GOLDBUG Multi-Encrypting Messenger – in: A Worldwide Survey of Encryption Products, URL: https://www.schneier.com/cryptography/paperfiles/worldwide-survey-of-encryptionproducts.pdf, February 11, 2016 Version 1.0.

- Security Blog: Secure chat communications suite GoldBug. Security Blog, 25. März 2014, http://www.hacker10.com/other-computing/secure-chat-communications-suite-GoldBug/

- Spot-On (2014): Documentation of the Spot-On-Application, URL: https://github.com/textbrowser/spot-on/tree/master/branches/trunk/Documentation, Github 2014.

- Spot-On (2011): Documentation of the Spot-On-Application, URL: https://sourceforge.net/p/spoton/code/HEAD/tree/, under this URL since 06/2013, Sourceforge, including the Spot-On: Documentation of the project draft paper of the pre-research project since 2010, Project Ne.R.D.D., Registered 2010-06-27, URL: https://sourceforge.net/projects/nerdd/ has evolved into Spot-On. Please see http://spot-on.sf.net and URL: https://github.com/textbrowser/spoton/blob/master/branches/Documentation/RELEASE-NOTES.archived, 08.08.2011.

- Theisen, Michaela: GoldBug Instant Messenger - Beliebte Software, Sicherer Instant Messenger, URL: https://www.freeware-base.de/freeware-zeige-details-28142-GoldBug_Instant_Messenger.html, 2015

- Tur, Henryk / Computerworld: GoldBug Secure Email Client & Instant Messenger, https://www.computerworld.pl/ftp/goldbug-secure-email-client-instant-messenger.html, 11.01.2018

- Vaughan-Nichols, Steven J.: How to recover from Heartbleed, ZDNet, April 9, 2014, http://www.zdnet.com/how-to-recover-from-heartbleed-7000028253

- Weller, Jan: Testbericht zu GoldBug für Freeware, Freeware-Blog https://www.freeware.de/download/GoldBug/


Weitere bibliographische Hinweise:


- Bernat, V. (2012, January 1). SSL/TLS & Perfect Forward Secrecy. Web article retrieved March 5, 2015, from http://vincent.bernat.im/en/blog/2011-sslperfect- forward-secrecy.html;

- Schum, Chris: Correctly Implementing Forward Secrecy, SANS Institute InfoSec Reading Room, March 14, 2014,

- Zhu, Y. (2014, April 8): Why the Web Needs Perfect Forward Secrecy More Than Ever. Web article. Retrieved February 2, 2015, from https://www.eff.org/deeplinks/2014/04/why-web-needs-perfect-forward-secrecy

---

**goldbug-manual-de is maintained by compendio.**
This page was generated by GitHub Pages.