SU3 Dateien

Was ist eine SU3 Datei?
Eine SU3 Datei (**s**igned **u**pdate) beinhaltet mehrere Objekte. Zum einem beinhaltet eine SU3 Datei eine andere Datei beliebigen Formats, Metadaten (Datum, Version, ...) und eine Signatur, um die Datei und die Metadaten zu verifizieren.Eine genaue (technische) Beschreibung findet man auf https://geti2p.net/spec/updates.

Vorgängerversionen
Bevor es das SU3 Dateiformat gab, gab es bereits das .sud sowie die .su2 Dateiformat. Allerdings wiesen diese Dateiformate Fehler auf. Zu diesen gehörten, dass es keine Magic Number gab und der Signer (siehe unten) nicht angegeben wurde, wodurch alle bekannten Schlüssel ausprobiert werden mussten. Des Weiteren fehlten noch recht viele Metadaten.

Was beinhaltet eine SU3 Datei genau?
Wie oben bereits erwähnt, beinhaltet eine SU3 Datei eine weitere Datei, eine Signatur sowie Metadaten. Die Signatur wird benutzt, um den Autor der weiteren Datei zu verifizieren. Zu diesem Zweck wird ein Public-Key-Verfahren angewendet.Dies bedeutet, dass es (wie bei GnuPG) zwei Schlüssel gibt, einen öffentlichen und einen privaten. Mit Schlüssel ist eine digitale Datei gemeint, welche es möglich macht, bestimmte kryptographische Funktionen zu nutzen. Der private Schlüssel wird benutzt, um die Datei zu signieren. Dies ist vergleichbar mit einer analogen Unterschrift. Um diese allerdings zu tätigen braucht man digital einen privaten Schlüssel. Der öffentliche Schlüssel wird benutzt, um die Signatur(oder analog Unterschrift) zu verifizieren, also zu überprüfen und deren Echtheit zu bestätigen. Damit soll bestätigt werden, dass die Datei wirklich vom Autor und nicht vom einem Bösen kommt. Dies bedeutet also, dass man den privaten Schlüssel geheim halten muss. Den öffentlichen Schlüssel kann man allerdings öffentlich frei zugänglich machen. Da man diesen auch benötigt und die Signatur zu verifizieren.Bei einer SU3 Datei nennt man die Person, welche die Datei signiert (also unterschreibt) Signer. Ein Signer wird anhand einer E-Mail-Adresse identifiziert.Diese muss nicht echt sein, sondern nur das Format einer E-Mail-Adresse besitzen. Diese E-Mail-Adresse nennt man beim SU3 Format SignerID. Zusammenfassung: Die SU3 Datei wird mit einem privaten Schlüssel (einer Datei)signiert (unterschrieben). Diese Signatur (Unterschrift) kann man mithilfe des öffentlichen Schlüssel verifizieren (bestätigen). Dies wird getan, um die Echtheit des Absenders einer Datei zu bestätigen. Die Person, welche unterschreibt, nennt man Signer. Die E-Mail-Adresse der Person nennt man SignerID. Des Weiteren (wie oben erwähnt) werden in einer SU3 Datei auch Metadaten gespeichert.Zu diesen gehören:

- Die Magic Number. Eine Magic Number steht immer am Anfang einer Datei und gibt den Typ der Datei an (z. B. OpenDocument, PDF oder eben SU3 Datei). Bei einer SU3Datei ist die Magic Number "I2Psu3".
- Die Art der Signatur. Es gibt verschiedene Schlüsselarten und Public-Key-Verfahren. In den Metadaten wird also angegeben, welche dieser Arten verwendet wird.
- die SignerID (siehe oben)
- Die Art der SU3 Datei (siehe weiter unten den Verwendungszweck). Man kann allerdings auch andere Dateien in einer SU3 Datei speichern.
- Den Typ der Datei. Zu den unterstützen Typen gehören:
  - ZIP Dateien
  - XML Dateien
  - HTML Dateien
  - XML Dateien komprimiert mit GnuZip
  - Text (txt) Dateien komprimiert mit GnuZip
- Die Version. Darunter versteht man eine Zahl, welche ein Datum angibt. Normalerweise ist das das Datum, wann die SU3 Datei erstellt wurden ist.
- Verschiedene Daten, damit die Datei richtig gelesen wird.

Verwendungszweck

Es gibt mehrere Zwecke, wofür eine SU3 Datei verwendet wird:

- Unbekannt. Dieser Verwendungszweck ist nicht weiter spezifiziert.
- Router[1] Update. Eine SU3 Datei kann ein Router Update beinhalten. Natürlich könnte man ein Router Update auch in einer „normalen“ einfachen ZIP Datei oder ähnliches zur Verfügung stellen. Allerdings kann man, wenn man eine SU3 Datei verwendet, sicherstellen, dass das Update tatsächlich vom I2P Team kommt.
- Plugin oder Plugin Update. Es gibt verschiedene Plugins für den I2P Router. Diese sollen allgemein dazu dienen, den Umgang mit I2P und verschiedenen anderen Anwendung zu vereinfachen. Zwei bekannte Plugins sind z. B. MuWire, ein Filesharing-Programm und I2P-Bote, ein dezentrales E-Mail-System. Ein Plugin kann entweder in einer .xpi2p oder einer .su3 Datei geliefert werden. Es wird immer empfohlen, wenn man die Wahl hat, die SU3 Datei zu benutzten. Wieder aus dem einfachen Grund, da man so die Echtheit des Entwicklers feststellen kann. Ansonsten könnte eine böse Person, mit einem solchen Plugin, den Router oder den Computer/Server stark beschädigen.
- Reseed Hosts. Damit der Router sich bei der ersten Installation oder nach einer langen Ruhezeit wieder mit dem I2P-Netzwerk verbinden kann, braucht er ein paar Router, mit denen er sich am Anfang verbinden kann. Diese können allerdings nicht von anderen Routern kommen, da sein eigener Router noch keinen anderen kennt. Es wird

1 Hier ist mit Router der I2P Router gemeint, also die eigentliche I2P Software.

also eine Reseed-Datei, welche Router Informationen beinhaltet, aus dem Clearnet oder dem Tor-Netzwerk heruntergeladen. Also man braucht eine Datei, damit man sich nach der ersten Installation von I2P mit dem Netzwerk verbinden kann. Wenn man eine falsche oder manipulierte Datei als Reseed Datei nimmt, kann dies seine eigene Anonymität stark gefährden oder gar entfernen. So ist es also wichtig, dass eine solche Reseed Datei von einer vertrauenswürdigen Quelle kommt. Um allerdings die Echtheit dieser Datei zu überprüfen, wird eine SU3 Datei benötigt. Bei einer Reseed Datei beinhaltet die SU3 Datei eine ZIP Datei. In dieser ZIP Datei sind wiederum Router Informationen hinterlegt, um sich mit dem Netzwerk zu verbinden.
- News(feed). Dies sind die Nachrichten, welche immer auf der Startseite des Router erscheinen. Eine Nachricht kann z. B. sein, dass eine neue I2P Version veröffentlicht wurde. Häufig enthält die Nachrichtendatei auch noch eine Blocklist mit gefährlichen IP-Adressen. Zu diesen verbindet sich dann der Router nicht. Die News werden im XML Format in der SU3 Datei gespeichert.
- Blocklist(feed). Dieses Funktion ist noch nicht implementiert und daher noch nicht dokumentiert. Es gibt hier also leider nichts zu erklären.

Zukunft der SU3 Datei

Achtung: Dieser Abschnitt enthält nicht 100% verifizierte Informationen.
Es scheint nicht so zu sein, dass bald eine neues Format vom I2P Team veröffentlicht wird. Es wird also voraussichtlich kein SU4 Format geben. Außer die Implementierung der Blocklistfeed-Funktion ist dieses Format sehr ausgereift.

Manuelles erstellen einer SU3 Datei mit Java und der i2p.jar-Datei

Dieser Abschnitt ist eventuell für Nicht-Entwickler oder „normale“ Benutzer von I2P uninteressant oder langweilig. Es wird einiges Vorwissen für das Verstehen dieses Abschnitt notwendig sein.
Um eine SU3 Datei zu erstellen, braucht man die i2p.jar Datei. Diese findet man normalerweise im Installationsverzeichnis von I2P im Ordner lib. Bei Linux also ~/.i2p/lib/i2p.jar bzw. /usr/share/i2p/lib/i2p.jar. Bei der Anleitung wird der Pfad zu dieser Datei lib/i2p.jar sein.
Um die Funktionen nutzen zu können, ist folgendes Java-Kommando, welches man in der Befehlszeile ausführen kann, elementar:

```
java -jar lib/i2p.jar su3file
```

Dabei ist es egal, ob man su3file groß oder klein schreibt.

Schlüsselpaar erzeugen

Um eine SU3 Datei erstellen zu können, braucht man erst einmal einen privaten und einen öffentlichen Schlüssel. Diesen kann man mit folgendem Befehl erzeugen:

java -jar lib/i2p.jar su3file keygen -t RSA_SHA512_4096 public.crt private.ks example@mail.i2p

In der Datei public.crt wird der öffentliche Schlüssel gespeichert. Diese Datei braucht der Empfänger auch für die Verifikation der SU3 Datei. In der Datei private.ks wird der private Schlüssel gespeichert. Diesen sollte man nie an irgendjemanden weitergeben. Der private Schlüssel wird wiederum in einer Java Keystore Datei gespeichert. example@mail.i2p ist hier die SignerID. Diese sollte man entsprechend mit seiner eigenen ersetzen. Zur Erinnerung, diese muss keine echte E-Mail-Adresse sein, sie muss nur das Format einer E-Mail-Adresse haben. Sie werden bei der Erstellung nach einem Password gefragt. Dieses wird benutzt, um den privaten Schlüssel noch einmal zu sichern. Man wird diesen Password benötigen, wenn man eine Datei mit dem privaten Schlüssel signiert.
RSA_SHA512_4096 ist das Verfahren, welches für die Signatur verwendet wird. Folgende stehen zur Auswahl:

- DSA_SHA1
- ECDSA_SHA256_P256
- ECDSA_SHA384_P384
- ECDSA_SHA512_P521
- RSA_SHA256_2048
- RSA_SHA384_3072
- RSA_SHA512_4096
- EdDSA_SHA512_Ed25519ph

Wenn das Argument -t weggelassen wird, wird RSA_SHA512_4096 benutzt.

Datei signieren

Dafür verwendet man den Parameter sign:

java -jar lib/i2p.jar su3file sign -c UNKNOWN -f HTML hello.html hello.su3 private.ks 1590434324 example@mail.i2p

-c beschreibt den Verwendungszweck. In diesem Fall ist dies leg endlich ein Beispiel, daher unbekannt. Folgende sind verfügbar:

- UNKNOWN
- ROUTER
- PLUGIN
- RESEED
- NEWS
- BLOCKLIST

Wird keiner angegeben, wird UNKNOWN verwendet. -f gibt das Datei Format an, welches die SU3 Datei beinhaltet. (siehe Was beinhaltet eine SU3 Datei genau? Abschnitt Metadaten) Folgende sind verfügbar:

- ZIP

- XML
- HTML
- XML_GZ
- TXT_GZ

Wird keiner angegeben wird ZIP verwendet.
Als nächstes wird die Datei angegeben, welche signiert und verpackt werden soll. In diesem Beispiel hello.html. Danach wird die Ausgabe Datei angegeben. In diesem Fall hello.su3. Danach die Datei, in welcher der private Schlüssel gespeichert ist. Als nächstes die Versionsnummer. In diesem Fall 1590434324 - Dies ist die typische UNIX-Abgabenformat für einen Zeitpunkt (https://en.wikipedia.org/wiki/Unix_time). Danach wird noch angegeben, mit welcher SignerID die Datei signiert werden soll.
Beim Signiervorgang wird nach einem Password gefragt. Dabei handelt es sich um dass Password, welches bei der Schlüsselerstellung eingegeben wurden ist.

Datei Informationen bekommen

Dazu kann man folgenden Befehl ausführen:

```
java -jar lib/i2p.jar su3file showversion hello.su3
```

Dabei ist hello.su3, die Datei, über die die Informationen eingeholt werden sollen.
Es wird in etwa folgende Ausgabe erscheinen:

```
Version:  1590434324 (25.05.2020, 21:18)
Signer:   example@mail.i2p
SigType:  RSA_SHA512_4096
Content:  UNKNOWN
FileType: HTML
```

Signatur verifizieren

Um die Signatur einer SU3 Datei zu verifizieren, benötigt man die Datei, in welcher der öffentliche Schlüssel gespeichert ist. Man kann dann z. B. folgenden Befehl ausführen:

```
java -jar lib/i2p.jar su3file verifysig -k public.crt hello.su3
```

Dabei ist public.crt, die Datei, in welcher der öffentliche Schlüssel gespeichert ist und hello.su3 die SU3 Datei, von welcher die Signatur verifiziert werden soll.

Inhalt auspacken

Dies kann man mit folgendem Befehl tun:

```
java -jar lib/i2p.jar su3file extract -x hello.su3
```

-x gibt an, dass man keine Verifizierung der Signatur möchte. Wenn man doch eine möchte, kann man wie bei „Signatur verifizieren“ den Parameter -k benutzen. hello.su3 ist die Datei, von der der Inhalt ausgepackt werden soll.