

# Die Internet-Tauschbörse

---

In dieser Abhandlung erkläre ich die grundsätzliche Funktionsweise einer Tauschbörse. Dabei werde ich anhand einiger Beispiele die Parallelen zu Verhaltensweisen des echten Lebens mit denen der digitalen Welt in Bezug setzen.

Ein Paar Fakten zu meiner Person: Mein Name ist Daniel Finger, ich bin Mitte 30 und programmiere seit 26 Jahren aktiv in verschiedenen Sprachen. Seit einigen Jahren bin ich selbst Gesellschafter einer Firma, die sich mit der Erstellung und Vertrieb von Software befasst.

## Warum Internet-Tauschbörsen?

Immer wieder hört man, Internet-Tauschbörsen wären illegal. Das stimmt zunächst nach meinem Rechtsverständnis nicht. Jeder von uns dürfte schon einmal einen Trödelmarkt besucht haben, was würden wir dazu sagen, wenn auf einmal der Besuch eines Trödelmarktes illegal sein sollte? Schon vor hunderten von Jahren gab es Trödelmärkte, das ist selbst aus dem heutigen Leben nicht mehr wegzudenken. Da dort zum größten Teil Menschen ohne gewerblichen Hintergrund sich Ihrer nicht mehr benötigten Waren entledigen, stellt das einen steuerfreien Privatverkauf, bzw. einen Tausch gegen eine andere Ware dar. Der Fiskus verdient hier also nichts.

## Vor- und Nachteile einer Internet-Tauschbörse

Eine Internet-Tauschbörse dient primär der schnellen und dezentralen Verbreitung von Dateien verschiedenster Inhalte. Eine typische Anwendung ist die Verbreitung von freier Software. Es gibt viele geniale kostenlose Programme wie z.B.

OpenOffice, Fedora-Linux, um nur einige zu nennen. Die beiden genannten erfreuen sich einer Vielzahl von Downloads, was natürlich eine gewaltige Menge an Daten verursacht. Würde z.B. die Fedora-Foundation die Server, von denen man die Dateien regulär downloaden könnte, selbst betreiben müssen, wäre das mit einem erheblichen Kostenaufwand verbunden und die freie Software wäre nicht mehr kostenlos zu verbreiten, was wiederum die Beliebtheit einschränken würde.

Ein weiteres Beispiel wäre die Verbreitung eines selbstgedrehten Filmes, z.B. einen Lehrfilm – Vereine machen so etwas gerne. Natürlich gibt es YouTube, aber einen Film in HD Qualität kann man dort nicht einstellen.

Unter Verwendung einer Tauschbörse ist jedermann in der Lage auch große Datenmengen einer Vielzahl von anderen Personen kostenfrei zur Verfügung stellen zu können.

Ein gravierender Nachteil ist die rechtliche Unsicherheit der Verbreitung der Daten, weil systembedingt keine Garantie herrscht, dass vermeintliche Dateien einen ganz anderen Inhalt aufweisen können, was erst nach erfolgreichem Download prüfbar ist.

## Tödelmarkt oder doch nicht?

Eine Internet-Tauschbörse ist sicher nicht zu hundert Prozent mit einem Trödelmarkt vergleichbar, weist dennoch einige Analogien auf. Auch dort wird getauscht, jedoch geht der Tausch in einer Internet-Tauschbörse mit einer Duplizierung eines digitalen Werkes einher. Betrachten wir das Beispiel mit dem Trödelmarkt etwas genauer:

Anton A. möchte gerne das Buch „der Bestseller“ von Berta B. haben, im Gegenzug bekommt Berta B. von Anton A. ein anderes Buch. Nach erfolgreichem

Tausch existieren weiterhin 2 Bücher, denn sie haben sich nicht vermehrt. Hätten beide jeweils das Buch im Handel als Neuware erworben, hätten die Verleger und die Autoren dieser Bücher sicher einen monetären Vorteil gehabt, es wären aber auch 2 weitere Exemplare existent gewesen.

### **Es ist doch ganz anders**

In einer Internet-Tauschbörse funktioniert der Tausch prinzipbedingt etwas anders. Je nachdem welches Tauschbörsen-Netzwerk zum Einsatz kommt, stellt sich die angebotene „Ware“ anders dar. Die beiden größten Netzwerke sind zum einen das BitTorrent-, bekannt durch z.B. das Programm „Azureus“, zum anderen das eD2K-Netzwerk, bekannt durch Programme wie „eMule“, „aMule“, „lPhant“ – um einige der wichtigsten Vertreter zu nennen. Die beiden Netzwerke unterscheiden sich ein wenig in Ihren Prinzipien. Die Programme, die ein Netzwerk implementieren, benutzen dieses auch in der Form wie das Netzwerk dieses vorgibt.

### **Ein Vergleich aus dem Leben**

Um das verständlich zu machen kann man sich verschiedene Automobilhersteller oder Autos vorstellen, diese repräsentieren die Programme. Fast alle verwenden den Ottomotor (Benzinmotor) oder den Dieselmotor, was man den Netzwerken (BitTorrent oder eD2K) gleichsetzen kann. Beide Motoren sind ähnlich, haben aber doch gewisse Unterschiede und verrichten am Ende aber doch die gleiche Arbeit - nämlich Dateien übertragen. BitTorrent z.B. benötigt immer einen sog. Tracker. Das ist ein Rechner, der für das Programm per Internet erreichbar sein muss und dafür sorgt, dem Programm zu „erzählen“, wo man denn das gewünschte findet. Der Tracker ist also der „Platzwart“ auf dem Trödelmarkt, der genau weiß, wer sich noch für das Buch „Der Bestseller“

interessiert und wer es besitzt und abgeben könnte.

Der Platzwart ist hier zunächst nur für dieses eine spezielle Buch zuständig. Er kann aber durchaus weitere andere Bücher oder andere Waren verwalten. Es gibt aber i.d.R. nur einen Platzwart, der die Zuständigkeit für eine bestimmte Sache übernimmt. Das ist eine Eigenschaft des BitTorrent-Netzwerkes, genau wie der Dieselmotor, der gegenüber dem Otto-Motor keinen Vergaser benötigt, aber trotzdem Drehbewegung erzeugt.

Es gibt also normalerweise für jede angebotene Ware nur einen Tracker im BitTorrent-Netzwerk. Natürlich kann man auch mehrere Tracker betreiben, aber wir gehen hier zunächst vom Standardfall aus.

### **Wer sucht der findet**

Um von der Existenz einer Ware zu erfahren, sucht man auf dem Trödelmarkt die Stände ab. Konkret auf die digitale Welt übertragen bedeutet das, man sucht auf einer Webseite im Internet nach sog. Torrent-Files. Das ist eine kleine Datei, die eine Beschreibung der Daten beinhaltet. Konkret könnte es sich dabei um eine Menge von Dateien handeln, deren Länge samt einzelnen Dateinamen dort hinterlegt ist. Das ist eine Stärke der Programme, die das Torrent-Netzwerk implementieren, es ist in der Lage einen Dateiodner auf der Festplatte als komplettes „Paket“ anzubieten oder zu empfangen. Ein prominenter Vertreter, der eine Liste von Torrent-Files zur Vergütung stellt, ist z.B. die Webseite der Organisation „thePirateBay“. Dort werden Torrent-Files angeboten, die einen Tausch von urheberrechtlich geschütztem Material sowohl als auch freiem Material ermöglichen.

### **Spielwiese für Straftäter**

Es liegt natürlich an jedem Benutzer selbst, sich dafür oder dagegen zu

entscheiden, urheberrechtlich geschütztes Material zu laden oder zu verbreiten. Allerdings degradiert die Tatsache, dass Urheberrechtsverletzungen begangen werden können, eine Tauschbörse nicht grundsätzlich zu etwas Schlechtem. Man könnte auf einem Trödelmarkt auch verbotene Dinge tauschen, deswegen ist ein Trödelmarkt aber auch nicht gleich schlecht.

### **eD2K: eDonkey2000 oder eMule**

Verglichen mit dem eD2K-Netzwerk ergeben sich durchaus einige Unterschiede zum BitTorrent-Netzwerk. Das eD2K-Netzwerk verfolgt einen etwas anderen Ansatz. Aus dem Vorteil des einen ergibt sich der Nachteil des anderen. Bezugnehmend auf das Auto-Beispiel könnte man Folgendes sagen: Der Diesel hat i.d.R. zwei Betriebsmodi: Er fährt oder fährt eben nicht. Beim Ottomotor gibt es noch viele Zwischenmodi. Das BitTorrent –Netzwerk ist in dem Fall der Diesel, der wenn der Tracker/Platzwart nicht erreichbar ist, still steht.

Anders bei dem eD2K-Netzwerk, auf das ich nun im Weiteren Verlauf detaillierter eingehen werde, gibt es keine Tracker, sondern Server. Grundsätzlich ist das auf den ersten Blick vergleichbar, jedoch im Detail zeigt sich der Unterschied. Ein eD2K-Server ist grundsätzlich immer für alle Dateien und jeden Benutzer zuständig.

Der Name „eD2K“ ergibt sich im Übrigen aus der Bezeichnung „eDonkey 2000“, der Urvater des bis heute entstandenen eMule, dessen Funktionsweise gleich geblieben ist, aber stark erweitert wurde. Aus dem „Affen“ wurde also der „Esel“.

### **Der große Unterschied**

Zurückblickend auf das eD2K-Netzwerk erkläre ich nun dessen Funktionsweise. Beziehen wir uns auf das Programm

eMule, welches mit Abstand die am meisten verwandte Applikation im eD2K-Netzwerk sein dürfte. Das eD2K-Protokoll ist im Gegensatz zum BitTorrent-Protokoll lediglich in der Lage eine einzige Datei pro Download/Upload zu transportieren. Man benutzt daher i.d.R. einen Packer, wie z.B. (Win-)ZIP oder RAR, um mehrere Dateien zu einer einzigen zusammenzufassen, zu komprimieren und zudem auch noch mit einem Passwortschutz versehen zu können.

### **Ein Ausflug zum Einkaufen**

Nehmen wir an, wir wollen eine echte Schachtel Pralinen in einem Geschäft kaufen. Jedes Produkt hat einen Namen und einen Barcode auf der Schachtel. Der Name könnte auf verschiedenen Schachteln in verschiedenen Sprachen aufgedruckt sein. Die EAN-Nummer (europäische Artikelnummer) - also der Barcode – ändert sich aber nicht, denn der Inhalt ist identisch. Angenommen wir nehmen eine andere Schachtel zur Hand, deren Inhalt anders angeordnet ist, auch wenn der Inhalt identisch zu sein scheint, handelt es sich um ein anderes Produkt, also bekommt dieses auch eine andere EAN-Nummer.

Analog verhält sich mit Dateien, die getauscht werden. Jede Datei bekommt eine nahezu eindeutige Nummer, den sog. FileHash, der eine Prüfsumme der Datei darstellt, sozusagen der Fingerabdruck dieser Datei. Dieser Hash besteht aus 32 alphanumerischen Zeichen, basierend auf dem „Message-Digest Algorithm 4“, kurz MD4-Algorithmus.

### **Angebot und Nachfrage**

Nun genug der technischen Fachbegriffe, wir wollen die Funktionsweise besser kennenlernen. Es gibt i.d.R. zwei Methoden das „Angebot“ zu durchforsten. Bei der ersten Möglichkeit verhält es sich ähnlich dem BitTorrent-Protokoll, Angebot wird auf einer Webseite bereitgestellt. Ein

ebenfalls bekannter Vertreter für dieses Netzwerk wäre die Seite „GoldEsel“. Diese Seite stellt zum großen Teil Informationen über Dateien, die urheberrechtlich geschützt sind, bereit. Das ist natürlich genauso problematisch, wie die Seite der „thePirateBay“, die ebenfalls ermöglicht, viele urheberrechtlich geschützte Werke zu laden. Ein Herunterladen stellt mindestens ein Duplikat her. Das wäre, als würde ich mir von einem Bekannten ein geschütztes Buch oder einen geschützten Film kopieren. Auch wenn das für viele Menschen ein normales Verhalten zu sein scheint, stellt es zunächst eine Urheberrechtsverletzung dar.

### Was ist ein Hash und was ein Link?

Im Gegensatz zu den Torrent-Files des BitTorrent-Protokolls gibt es bei dem eD2K-Netzwerk zwar keine Dateien, die Auskunft darüber geben, ob und wo eine Datei zu bekommen ist, es gibt aber die sog. eD2K-Links. Ein eD2K-Link ist keine Datei, sondern lediglich eine sehr knappe Beschreibung, die folgende Teile enthält: Dateiname, Dateigröße in Bytes und den FileHash. Ein typischer eD2K-Link sieht z.B. wie folgt aus:

```
ed2k://file|datei.txt|123|1234567890abcdef1234567890abcdef|
```

### Ein Paar Grundlagen

Verglichen mit einem „normalen“ Link auf einer Webseite passiert dort Folgendes: Klickt der Benutzer einen Link an, wie z.B.: [www.einetolleseite.de](http://www.einetolleseite.de), dann wird das Standardprotokoll „http“ angenommen und der Link wird intern automatisch auf <http://www.einetolleseite.de> erweitert.

Das wiederum bedeutet, dass der erste Teil, nämlich „http“ das Protokoll definiert, mit dem Webseiten angesehen werden können. Dazu wird das zugehörige Programm „Internet Explorer, Firefox, etc.“ automatisch geöffnet und die Zeichenkette

hinter den zwei Schrägstrichen an den Browser übergeben, der dann die Seite [www.einetolleseite.de](http://www.einetolleseite.de) öffnet.

Beginnt ein Link mit „ed2k“, ist im Gegensatz zu „http“ nun das Programm eMule zuständig, welches ebenfalls die Informationen hinter den Schrägstrichen übermittelt bekommt. Damit wird nun dem Programm eMule mitgeteilt, es soll nach einer Datei suchen die den FileHash „1234567890abcdef1234567890abcdef“ hat und diese versuchen herunterzuladen. Die Angabe der Länge von 123 Bytes dient einer weiteren Prüfung und Aufteilung der Datei in sog. „Chunks“ (Erklärung dazu folgt später). Diese Datei soll dann „datei.txt“ heißen, egal ob das stimmt oder nicht.

### Herunterladen, nur woher?

Jetzt kommt der Server ins Spiel. Auch für den Server gibt es sog. Links auf diversen Webseiten. Folgender Eintrag zeigt einen typischen Serverlink, der in diesem Fall nicht funktioniert, da die IP-Adresse nicht gültig ist:

```
ed2k://|server|123.456.789.012|4661|
```

Dieser Link veranlasst den „eMule“ einen Server mit der (ungültigen) IP-Adresse 123.456.789.012 zu kennen. Dieser Server kennt im Übrigen weitere Server und nachdem der „eMule“ sich mit einem beliebigen Server verbunden hat, bekommt er von diesem eine weitere Liste mit anderen Servern, wenn gewünscht. Damit steht dem „eMule“ eine beachtliche Liste von Servern zur Verfügung, die den Betrieb trotzdem ermöglichen, auch wenn der ursprüngliche Server nicht mehr funktionieren sollte. Dann verbindet sich der „eMule“ einfach mit einem anderen Server.

### Der Download beginnt

Was passiert nun genau, wenn der Download beginnt? Der „eMule“ hat nun wie eingangs beschrieben eine Datei

namens „datei.txt“ zum Download in seiner Liste der herunterzuladenden Dateien eingefügt. Nun stellt der „eMule“ eine Anfrage an den Server, nennen wir ihn passenderweise „eMule-Server 1“.

### Die Sicht des Downloaders

Der „eMule“ möchte nun wissen, woher die Datei „datei.txt“ oder Teile davon bezogen werden können. Also schickt der „eMule“ als Suchanfrage den FileHash der gewünschten Datei, also den Fingerabdruck, den wir von einer Internetseite oder eine Suche erhalten haben, an den Server. Es wird schnell klar, dass hier nicht nach einem Dateinamen oder ähnlichem gesucht wird, denn von dem Dateinamen weiß der Server hier zunächst nichts. Welcher Dateiname bei dem „Downloader“ verwendet wird ist also völlig privat und zunächst von niemandem einsehbar außer von demjenigen, der die Daten per Upload bereitstellt.

Der „eMule“ bekommt von dem verbundenen Server nun die Antwort in Form von Kontakten, bestehend aus IP-Adresse, also der Anschluss, mit dem der Zugang zum Internet realisiert wurde, eines oder mehrerer anderer Teilnehmer, die diese Datei anbieten. Weiterhin wird dazu übermittelt, welche „Chunks“ der Datei bei dem Anbietenden vorliegen.

### Ein Chunk

Ein „Chunk“ ist ein Teil einer Datei. Abstrakt erklärt gehen wir mal von einer Filmdatei einer Länge von 30 Minuten aus, die in Stücke zu je 1 Minute unterteilt wird. Die Stärke einer Tauschbörse besteht in der Aufteilung der Datei, die aus verschiedensten Quellen zusammengeführt wird. Es kann also sein, dass die „erste Minute“ des herunterzuladenden Filmes vom Teilnehmer „A“ geladen wird, die „dritte Minute“ zeitgleich vom Teilnehmer „B“. So setzt sich am Ende eine Datei zusammen.

Der „eMule“ würde also nur die fehlenden Teile nur von Teilnehmern zum Download anfordern, die diesen Teil auch wirklich schon vollständig selbst anbieten können. Konkret verbindet sich der „eMule“ mit anderen „eMule“-Teilnehmern auf „Empfehlung“ des Servers.

### Die Sicht des Uploaders

Aus Sicht des Anbietenden stellt sich der Informationsfluss etwas anders dar. Es gibt 2 zu unterscheidende Aspekte wenn es um das Anbieten von Dateien geht. Zum einen werden die Dateien im sog. „Incoming-Verzeichnis“ automatisch angeboten, das sind diese Dateien, die kürzlich heruntergeladen wurden und nun vollständig vorliegen. Desweiteren können natürlich beliebige weitere Verzeichnisse zum Anbieten freigegeben werden.

Damit diese Tauschbörse schnell funktioniert, wird jeder, der mindestens einen „Chunk“ (s.o.) vollständig vorliegen hat, unfreiwillig und zwangsläufig zunächst zum Anbieter. Ein anderer Teilnehmer kann also dann genau diesen „Chunk“ anfordern und laden.

Sobald also eine Datei, egal ob vollständig oder nicht, zum Anbieten zur Verfügung steht, werden nun Informationen darüber an den verbundenen Server gesendet. Diese beinhalten im Wesentlichen wieder den FileHash, also den Fingerabdruck, die Dateilänge und den Dateinamen.

### Namen sind Schall und Rauch

Spätestens hier wird klar, dass die individuelle Vergabe von Dateinamen jedem frei steht. Ein konkretes Beispiel wäre z.B. wenn ein Nutzer eine Datei mit Namen „PrivatesVideo.avi“ auf seiner Festplatte liegen hat, die z.B. die privaten Aufnahmen des letzten Geburtstages beinhalten. Nun möchte der Nutzer aber nicht, dass dieses Video auf seinem Rechner sofort von jedem gefunden werden kann. Eine gängige, wenn noch

primitive Praxis wäre, die Datei einfach umzubenennen, z.B. in „langweilige-witzesammlung.zip“. Will man nun dieses vermeintliche Archiv öffnen, wird dies mit einer Fehlermeldung „ungültiges Archiv“ quittiert. Natürlich kann man diese Datei mit einem Videoplayer direkt öffnen, dann funktioniert das Anschauen dieses Videos – sofern man auf diese Idee kommt oder Kenntnis davon hat“

### **Falsche Dateien in Tauschbörsen**

Mancher unbedarfte Nutzer einer Tauschbörse gibt gerne unwissentlich seine ganze Festplatte zum Tausch frei. Es können somit derartige Dateien wie „langweilige-witzesammlung.zip“ plötzlich im Tauschbörsen-Netzwerk zur Verfügung stehen. Andere Nutzer können diese Datei per Suche finden und die selbige herunterladen ohne zu wissen, dass dort keine langweiligen Witze enthalten sein werden. Leider passiert es mittlerweile sehr häufig, dass sich hinter Dateien nicht das verbirgt, was man sich erhofft. Es wird davon ausgegangen, dass mittlerweile bis zu 10% aller Dateien im eD2K-Netz fehlerhaft benannt sind oder - noch schlimmer – Malware oder Trojaner enthalten.

### **Jetzt wird's vernetzt.**

Angenommen Nutzer A verbreitet unwissentlich sein Video als „langweilige-witzesammlung.zip“, Nutzer B lädt es von Nutzer A und stellt fest, es ist ein interessanter privater Film. Er benennt die Datei von „langweilige-witzesammlung.zip“ in „SpannendesVideo.avi“ um und bietet dieses wiederum an. Will jetzt ein dritter die „langweilige-witzesammlung.zip“ herunterladen, wird er von Nutzer A und von Nutzer B Teile des Filmes bekommen. Der Name ist also völlig irrelevant, die Datei wird ausschließlich durch den FileHash, den Fingerabdruck identifiziert.

### **KADemlia**

Es gibt eine weitere Technik, die verschiedene Tauschbörsennutzer miteinander „bekannt“ macht. Historisch gesehen ist diese Technik eine Weiterentwicklung des regulären serverbasierten eD2K-Netzwerkes und wurde im Jahre 2003 in Form des Clients „Overnet“ von Jed McCaleb ins Leben gerufen. Kademlia selbst kann keine Dateien übertragen. Es funktioniert nach dem „einer kennt den nächsten“-Prinzip. Somit fungiert jeder Benutzer in einer Art und Weise als Server, oder besser formuliert als „Wegweiser“ um Dateien zu finden. Hier wurde eine Alternative zu dem serverbasierten Modell gesucht, um von Betreibern der Server unabhängig sein zu können. Bis heute macht, wenn man statistischen Auswertungen glauben kann, der Anteil der KAD-basierten Anfragen weniger als 1 Prozent aus. Die mangelnde Durchsetzung der serverlosen Verbindungssuche liegt wohl der unterdurchschnittlichen Geschwindigkeit zugrunde.

### **Lokale, globale oder KAD-Suche**

Die Suche über den „eMule“ wird normalerweise immer gegenüber dem verbundenen Server durchgeführt. Zur Erinnerung: Der Server kennt die Datenamen von anderen verbundenen Nutzern in Bezug zu einem FileHash. Es können also ohne Weiteres mehrere verschiedenartig benannte Suchergebnisse auf dieselbe Datei verweisen. Es besteht außerdem die Möglichkeit der globalen Suche, deren Ergebnisse auf weitere Server ausgeweitet werden. Die KAD-Suche funktioniert nach dem Prinzip, dass der „eMule“ andere Nutzer „kennt“ und diese auch befragt. Diese Nutzer wiederum kennen weitere Nutzer, sodass sich die Anfrage wie eine Flut durch das Netz der Teilnehmer fortbewegt.

## Download = Upload?

Oft wird die Frage gestellt, ob das Herunterladen einer Datei zwangsläufig auch immer einem Upload (Anbieten) gleichzusetzen ist. Die Antwort lautet: Jein!

Zunächst hat der Upload mit dem Download nichts direkt zu tun und muss technisch auch differenziert betrachtet werden. Unter dem Upload versteht man eigentlich die Menge der Daten, die ein Anschluss in Richtung eines anderen überträgt.

Lädt man beispielsweise von einer Webseite eine Datei herunter, ist das ein Download. Dazu wird i.d.R. das sog. TCP/IP-Protokoll verwendet, welches die Daten Paketweise empfängt. Nach jedem Paket sendet der empfangende PC eine kurze Statusantwort, damit dem Sender signalisiert wird, dass alle Daten ordnungsgemäß angekommen sind. Davon bekommt der normale Anwender i.d.R. nichts mit. Diese kurzen Antworten können ca. bis zu 10% der heruntergeladenen Datenmenge als Upload auf dem Volumen-Zähler erscheinen lassen. Es sind aber keine Nutzdaten sondern lediglich Statusdaten, die keinen Upload im Sinne einer Tauschbörse darstellen. Aus diesem Grund, aber auch weil andere Nutzer Dateianfragen an einen Teilnehmer stellen, kann die Geschwindigkeitsanzeige des „eMule“ zeitweise eine geringe Upload-Geschwindigkeit anzeigen, obwohl kein Upload einer Datei stattfindet.

## NoUpload-Client und andere Mods

Die schwierige Gesetzeslage erfordert es manchmal, dass neue Methoden erdacht werden, der Upload-Falle zu entgehen. Die Verwendung eines sog. Leecher-Mods widerspricht zwar der Netzkultur, es bleibt aber jedem selbst überlassen, ob man sich einfach nur schlecht benimmt oder

aber tatsächlich, wenn auch unbewusst, einer Straftat bezichtigt werden kann. Technisch ist es ohne Weiteres möglich, den Upload zu unterbinden. Ob alle im Netz frei verfügbaren NoUpload-Clients auch tatsächlich den Upload unterbinden, ist eine Frage des Vertrauens. Eine exemplarische Abhandlung folgt am Ende dieses Dokumentes.

## Gefälschte „Chunks“

Der Erstellung der Prüfsummen, dem sog. FileHash liegt der MD4-Algorithmus zugrunde, der durch seine betagte Implementierung als sehr unsicher angesehen wird. Es ist also ohne weiteres möglich, von einem anderen Nutzer einen falschen „Chunk“ ungewollt zu laden. Zwar wird bei der endgültigen FileHash-Errechnung die Datei nochmals geprüft, damit bis dahin fehlerhaft heruntergeladene Teile erneut geladen werden können, jedoch wird das Netzwerk durch ständige „Störungen“ dieser Art unnötig belastet.

## Manipulierte Server

Es existieren eine Menge Server im Netz, die nicht so funktionieren, wie man es erwarten würde. Die Flut neu hinzugekommener Server ist für den Benutzer nicht mehr zu überblicken, da die Namen der Server von deren Betreiber frei vergebbar sind und bedingt durch Ähnlichkeit im Namen für den Durchschnittsanwender nicht zu unterscheiden sind.

Desweiteren locken manche Server mit angeblich hohen Datei- oder Nutzerzahlen, die es interessant machen, Dateien mithilfe dieser Server schneller Laden zu können.

Ich möchte noch kommentarlos den folgenden Absatz zum Thema „manipulierte Server“ aus Wikipedia hier einbringen:

Zitat: „Der Betreiber des Servers hat nun die Möglichkeit, Dateianfragen mitzuschreiben, zu filtern und zu fälschen. Dies kann über Dateitypen, Schlüsselwörter oder über Hash-Wert und Dateigröße geschehen. So versuchen etwa einige Server, Nutzer zum Download von Malware zu verleiten, indem sie jede Suchanfrage beantworten und der Malware einen zum gesuchten Begriff passenden Namen geben.

Einige Nutzer vermuten, dass Manipulationen teils im Auftrag einer Interessengemeinschaft der Film- oder Musikindustrie geschehen, zum Beispiel der RIAA, die auf diesem Weg dem Netzwerk schaden und Informationen über urheberrechtsverletzende Aktivitäten bekommen kann.“ Zitat Ende.

## Wer betreibt diese Server und warum?

Angefangen hat der Betrieb der Server schon im letzten Jahrtausend. Einige Personen, die damals einen undenkbar schnellen DSL- oder Kabel-Anschluss besaßen, konnten diesen kaum ausreizen. Aus Spaß an der Entwicklung solcher Netze fanden sich genügend Personen oder auch Firmen, die einen solchen Server kostenfrei betrieben haben oder auch noch betreiben. Da sich nachdem der Nutzer mit einem Server verbunden war, beliebiger Text als „Begrüßungsmeldung“ hinterlegen ließ, erkannte man schnell die Möglichkeit, diese als Werbefläche zu vermarkten um den Betrieb von noch stärkeren Servern zu gewährleisten.

## Rechtliche Problematiken

Um die Frage zu beantworten, wie viele Daten eindeutig geladen werden müssten um eine eindeutige Identifizierung zu ermöglichen, erkläre ich nachfolgend die technischen Rahmenbedingungen für eine zu übertragende Datei:

Um anhand der geladenen Daten tatsächlich den Inhalt des Datenstromes zu ermitteln, muss m.E. mindestens ein Chunk geladen werden. Das bedeutet 9,28 Mb an Daten, um überhaupt einen MD4-PartHash erstellen zu können, der als Grundlage für den zu errechnenden FileHash dient. Da jedoch wie zum Thema „gefälschte Chunks“ bereits beschrieben, die Daten manipuliert werden können, ist auch bei lediglich einem Chunk keine Garantie gegeben, dass die Dateninhalt mit dem übereinstimmt, was man anhand des FileHashes erwarten würde.

## NoUpload-Client im Detail

Nachfolgend wird beispielhaft erläutert, welche genauen Änderungen an einem „eMule“-Client durchgeführt werden können und welche Auswirkungen das auf andere Teilnehmer oder Server hat.

Es gibt 3 wesentliche Funktionen, die in der Tauschbörsensoftware eMule oder aMule verändert werden müssen. Sicherlich ist nicht jedermann in der Lage diese Modifikationen durchführen zu können, es sind aber genügend modifizierte eMule-Clients im Umlauf.

Die Netzwerk-Kernfunktion von aMule entspricht weitestgehend der von eMule, da diese ein Fork (Weiterentwicklung) ist. Die Nachfolgenden Ausführungen gelten also gleichwohl für beide Programme. Im Übrigen handelt es sich sowohl bei aMule als auch bei eMule um freie, quelloffene Software, die jedermann nach Belieben ändern und weiterentwickeln kann. Der Quelltext ist frei verfügbar, so dass jedermann die nachfolgend beschriebenen Änderungen nachvollziehen und prüfen kann.

## Einen NoUpload-Client zu Testzwecken bauen

Folgende Dinge müssen im Quelltext geändert werden, um einen modifizierten

eMule- oder aMule Client zu kompilieren, der weder Uploads zulässt, noch andere Informationen preisgibt:

Die Klasse "UploadQueue.cpp"

Es gibt dort die Funktion „AcceptNewClient“, die dafür sorgt, dass ein anderer Nutzer bei unserem TestClient Zutritt zur sog. Warteliste erhält. Diese wurde soweit modifiziert, dass der Zutritt grundsätzlich immer verwehrt bleibt. Die Warteliste ist eine Liste, in den ein anderer Client Zutritt erlangen muss, um von diesem Daten zu laden.

Ein anderer Nutzer, der Zutritt zu der Warteliste unseres Test-Clients erlangen möchte, muss zunächst erst einmal Kenntnis davon haben, welche Dateien von unserem Test-Client angeboten würden, da sonst kein Grund für den Zutritt zur Warteliste bestehen würde.

## Real Life

Ein kleiner Vergleich aus dem echten Leben: Freddy F. als Rechteinhaber sieht, dass der Laden XY ein Produkt verkauft, an dem er Urheberrechte besitzt und möchte prüfen ob es sich dabei tatsächlich um sein geistiges Eigentum handelt. Er wird also einen Probekauf in dem Laden machen wollen, der mit dem Produkt wirbt. Ob dieses Produkt tatsächlich dort käuflich zu erwerben sein wird, weiß er erst nach einem Testkauf. Er möchte nun das Geschäft betreten, um dies zu prüfen. Er steht vor der Tür und kann in den Vorraum sehen, durch die Tür wird er aber nicht eintreten können. Er kann keinen Testkauf durchführen. (Testkauf = Upload)

```
-----  
bool CUploadQueue::AcceptNewClient()  
{  
    // NEVER ALLOW OTHER CLIENTS  
    return false;  
}  
-----
```

## Heute im Angebot

Die Klasse "SharedFileList.cpp"

„Befragt“ man einen „eMule“-Client, wird man folgende Daten erhalten:

- ID (Benutzerkennung aus der IP-Adresse berechnet)
- UserHash (eindeutige Benutzerkennung des Clients)
- Client (Name der Software, z.B. "eMule" mit Version)
- Liste der angebotenen Dateien.
- usw.

Die Liste der angebotenen Dateien wird in der Klasse "SharedFileList" verwaltet. Das kann man sich etwa wie in einem „Werbeprospekt“ vorstellen, in dem alle „Angebote“ aufgeführt sind. Wenn diese Liste leer ist, kann niemand von diesem Client Dateien empfangen. Der Client hat also nichts im Angebot. Da diese Liste tatsächlich leer bleibt, ist die obige Funktion "AcceptNewClient" im Grunde überflüssig - oder auch: doppelt gesichert, denn es gibt keinen Grund, sich mit diesem Client zu Verbinden.

An den folgenden Stellen können weitere Modifikationen durchgeführt werden, um den Upload und die Informationsfreigabe über vorgehaltene Dateien zu verhindern:

### Funktion 1: FindSharedFiles

Diese Funktion durchsucht die im Client freigegeben Verzeichnisse nach Dateien, die angeboten werden sollen. Die Funktion zum Durchsuchen der Verzeichnisse ist hier unterbunden. Das betrifft sowohl das „Incoming-Verzeichnis“, als auch alle sonstigen freigegebenen Verzeichnisse. Es kann nichts, was auf der Festplatte vollständig vorliegt angeboten werden!! Desweiteren hätte diese Funktion auch die noch unfertigen Dateien (Part- oder Temp-Files) zum Upload angeboten, was durch den Abbruch der Funktion ebenfalls unterbunden wird.

```

-----
void CSharedFileList::FindSharedFiles()
{
    //DON'T SCAN FOLDERS
    return;
}
-----

```

### Funktion 2: RepublishFile

Die Funktion "RepublishFile" teilt dem verbundenen Server mit, das eine bestimmte Datei (KnownFile) ganz oder in Teilen vorliegt. Die Details der Datei könnten an den Server oder andere Nutzer (KAD) übermittelt werden, wenn diese Funktion nicht unterbunden wäre. Selbst wenn die Funktion „FindSharedFiles“ Dateien finden würde, wüsste der verbundene Server oder direkt anfragende andere Nutzer nichts über den Inhalt dieser Liste. Um den Vergleich mit dem „Angebotsblättchen“ nochmal heranzuziehen: Wären in dem „Werbeprospekt“ Angebote enthalten, würde das nichts daran ändern, da die einzelnen Angebote nicht auf dem Prospekt aufgedruckt würden.

```

-----
void
CSharedFileList::RepublishFile(CKnownFile* pFile)
{
    //NEVER SEND FILE DETAILS
    return;
}
-----

```

### Funktion 3: SendListToServer

Diese Funktion "SendListToServer" teilt dem verbundenen Server die Liste der angebotenen Dateien mit. Hier gilt ebenfalls, wäre der „Prospekt“ mit Angeboten gefüllt, würde dieses ebenfalls nichts daran ändern, dass der „Prospekt“ niemals „verteilt“ werden würde.

```

-----
void CSharedFileList::SendListToServer(){

```

```

std::vector<CKnownFile*> SortedList;

//NEVER PUBLISH FILELIST
return;
}
-----

```

## Probieren geht über Studieren

Ziel war es also nun eine Lösung zu finden, in der kein Server etwas über eventuelle Dateien auf dem Client wissen darf.

Um diese Verhaltensweise zu gewährleisten, bietet es sich an zunächst das „normale“ Verhalten eines regulären „eMule“-Clients nachzuvollziehen. Dazu werden 2 uploadfähige Clients mit dem gleichen Server verbunden. Es wird bei Nutzer A eine Datei zum Upload angeboten, die einen außergewöhnlichen Namen tragen sollte. Die Datei kann nun von Nutzer B per Suche gefunden und heruntergeladen werden.

Nutzt man den modifizierten „eMule“-Client, kann diese Datei per Suche nicht gefunden werden, was die gewollte Funktionsweise des modifizierten Clients untermauert.

Völlig serverunabhängig, unter Verwendung des "Freundschaftsmodus", bei dem sich die Clients beidseitig aktiv bestätigen müssen, Nutzer B also als "Freund" von Nutzer A dessen Freigaben einsehen darf - sofern erlaubt - bleibt die Liste der angebotenen Dateien, bedingt durch die vorausgegangenen Modifikationen, ebenfalls verweigert.

Die Funktion zur direkten Abfrage ohne Server (z.B. über das KAD-Netzwerk) bleibt ebenfalls ergebnislos, da diese Funktion sich ebenfalls auf die "SharedFileList" – unser Angebotsprospekt - stützt, die wie bereits hinreichend erläutert wurde, leer bleibt.

## Fazit

Kenntnis davon, welche Datei mit einem dieser Art modifizierten Client heruntergeladen würde, könnte nach sorgfältiger Prüfung des Quelltextes nur derjenige erlangen, der Anbieter derselben sein müsste.

## Spekulationen

Wie bereits als Auszug der Wikipedia eingefügt häufen sich die Vorwürfe, Rechteinhaber der Musik- und Filmindustrie würden selbst illegal Dateien im Netz anbieten. Möglicherweise kann diese aufklärende Fassung einer Beschreibung zur Funktionsweise von Internet-Tauschbörsen dazu beitragen, diese Vorwürfe entweder zu entkräften oder zu untermauern.

Die stetig steigenden Mengen der Abmahnungen zeigen deutlich, dass dieser „Zukunftsmarkt“ noch viel Potential aufzuweisen scheint.

## Weiterführende Links

Statement von Dieter Bohlen zum Thema „Urheberrechtsverletzungen“.

<http://www.youtube.com/watch?v=hGEnQ8NRgE>

## Haftungsausschluss

Der Autor übernimmt keinerlei Gewähr für die Aktualität, Korrektheit, Vollständigkeit oder Qualität der bereitgestellten Informationen. Haftungsansprüche gegen den Autor, welche sich auf Schäden materieller oder ideeller Art beziehen, die durch die Nutzung oder Nichtnutzung der dargebotenen Informationen bzw. durch die Nutzung fehlerhafter und unvollständiger Informationen verursacht wurden, sind grundsätzlich ausgeschlossen, sofern seitens des Autors kein nachweislich vorsätzliches oder grob fahrlässiges Verschulden

vorliegt.

Alle Angebote sind freibleibend und unverbindlich. Der Autor behält es sich ausdrücklich vor, Teile der Seiten oder das gesamte Angebot ohne gesonderte Ankündigung zu verändern, zu ergänzen, zu löschen oder die Veröffentlichung zeitweise oder endgültig einzustellen.

## Copyright

© 2009 Daniel Finger

Mail: [contact@daniel-finger.com](mailto:contact@daniel-finger.com)