SMW-Hacking für Anfänger

Ein umfassendes Tutorial

von Eric Kaiser, aka WhiteYoshiEgg, aka WYE

Inhalt

Bevor's richtig losgeht

Ein kurzes Vorwort

Erst einmal **herzlich willkommen** zu meinem Tutorial. Schön, dass du hierher gefunden hast – ich hoffe, du machst dir die Mühe, auch ein bisschen weiter zu lesen als bis hierhin. (Wenn du dir nicht sicher bist, worum es hier eigentlich gehen soll, hab Geduld, deine Fragen werden sofort beantwortet.)

Mein Name ist **WhiteYoshiEgg** oder kurz **WYE**, zumindest im Internet. Ich habe das SMW-Hacken im Frühjahr 2007 entdeckt, und mittlerweile ist es zu meinem mit Abstand liebsten

Hobby geworden. Für einen Veteranen oder Experten halte ich mich nicht, aber ich glaube, ziemlich viel darüber zu wissen – auf jeden Fall genug, um dieses Wissen weitergeben zu können und zu wollen.

Jeder von uns war mal ein Anfänger, und du, der du das hier liest, bist vielleicht auch einer. Wenn man aber dieses Stadium überwunden hat und sich als Fortgeschrittenen bezeichnen kann, ist man die vielen, ewig gleichen Anfängerfragen schnell leid. Zumindest mir ging und geht das so, und ich habe zwar das Bedürfnis zu helfen, aber oft nicht die Lust, so viel Zeit und Tipparbeit für eine Antwort aufzuwenden. Ich habe mir oft gewünscht, es gäbe ein großes Tutorial, ein einzelnes riesiges Dokument, in dem alles verzeichnet ist und auf das man Anfänger einfach nur zu verweisen braucht… tja, und nun schreibe ich selbst eins. Es ist zwar ein Haufen Arbeit, zum SMW-Hacking wirklich alles zu erklären, was ich weiß, aber ich glaube, es wird sich lohnen.

Kleine Warnung: Was dich erwartet

Wie sich vielleicht schon gezeigt hat, ist mein Schreibstil ziemlich seltsam – ich neige zu langen, aneinandergereihten Sätzen, übermäßig vielen Kommas und Gedankenstrichen, ungewöhnlichem Vokabular und manchmal abstrusen Stilmitteln. **Du wirst eine Menge zu lesen haben**, und bevor das eigentliche Tutorial losgeht, musst du noch ein paar Seiten an Vorwissen über dich ergehen lassen.

Ich werde mein Möglichstes tun, mich zurückzuhalten, was Schreibstil und Textlänge angeht, aber komplett verstellen kann ich mich nicht. Ich hoffe, du steigst durch meinen Stil durch.

Was dieses Tutorial ist

…besser gesagt, was dieses Tutorial sein will. Hier eine Liste meiner Ziele:

- Dass verschiedene Leute zu verschiedenen Themen verschiedene Tutorials schreiben, ist verständlich und auch kein bisschen schlimm. Ideal wäre es aber doch, wenn man **ein einziges großes Tutorial** für alle Themen hätte, einheitlich und umfassend – ein Nachschlagewerk quasi. Das versuche ich hiermit zu erreichen.

- **Die Qualität vieler anderer Tutorials lässt leider zu wünschen übrig.** Ich habe schon viel zu viele Texte und Videos gesehen, die sich "Tutorial" nennen, aber eigentlich nur bloße Anleitungen sind. Ein schlechtes Tutorial ist auch nichts anderes als eine Anleitung – es erklärt dem Nutzer nur, *was* er tun muss *("Zuerst klickst du hier, dann gibst du dort das ein, dann kopierst du diesen Text, fügst ihn dort ein, und fertig")*. Ein gutes Tutorial hingegen erklärt immer auch, *warum* der Nutzer das tun soll. Es erklärt bei jedem Schritt, warum dieser wichtig ist und was genau dabei

eigentlich passiert, und liefert genug Informationen, dass der Nutzer wahrhaftig *versteht*, was er tut. Wer ein schlechtes Tutorial gelesen hat, der kann von sich nicht unbedingt behaupten, diese Tätigkeit zu *können* – er hat gelernt, die Anweisungen zu befolgen und das gelernte Schema immer wieder anzuwenden, aber er versteht das Thema nicht wirklich. Wer ein gutes Tutorial gelesen hat, kennt sich mit dem Thema (bis zu einem gewissen Grad) wirklich aus, und weiß nicht nur, was er tut, sondern auch warum. Er kann dann von sich behaupten, die Tätigkeit zu *beherrschen*. Und genau darum geht's mir auch – lange Rede, kurzer Sinn: **Ich will ein gutes Tutorial schreiben.** Es wird mehr Mühe machen – auch für dich als Leser – aber ich bin davon überzeugt, dass es sich lohnen wird.

- Dieses Tutorial soll nicht nur umfassend sein, sondern auch **gut geschrieben** und nicht allzu langweilig. Ich werde mein Bestes geben, verständlich zu schreiben, wo ich normalerweise zu überlangen Sätzen neige. Ich werde **möglichst wenig Vorkenntnisse** voraussetzen – außer gesundem Menschenverstand natürlich. Auch komplette Neulinge sollen verstehen können, worum es hier geht.

Was dieses Tutorial nicht ist

Wie gesagt, ich halte mich nicht für einen Vollprofi, der alles weiß, was es nur zu wissen gibt. Ich will dir lediglich beibringen, was ich weiß. Es kann durchaus vorkommen, dass ich an manchen Stellen sage **"keine Ahnung"** – das ist dann eben mein momentaner Wissensstand. In solchen Fällen werde ich aber alles, was ich dazu doch weiß, umso genauer erklären.

Außerdem ist dieses Tutorial ja nicht die einzige Wissensquelle – es gibt dutzende Datenlisten und Dokumente, die ich hier nicht mit einfüge, weil es einfach sinnlos wäre, sie eins zu eins zu übernehmen. (Auf sie verweisen werde ich natürlich trotzdem.) Was ich damit sagen will: Mach dich gerne auch woanders schlau, **dieses Tutorial ist nicht der Weisheit letzter Schluss**.

So, und nun endlich zur Sache.

Was ist SMW-Hacking?

Eine kurze Definition vorweg

Lass es dir gleich zu Anfang gesagt sein: Das "Hacken", von dem hier die Rede ist, ist **nichts Verbotenes**. Dazu gleich mehr.

Um deine Neugier schon mal ein wenig zu stillen, hier eine ganz kurze Definition: **SMW-Hacking ist das Erstellen eigener Spiele auf Basis des Spiels Super Mario World.** Es geht hier also ums **Spielemachen**, und genauer auf die Sache eingehen werde ich jetzt.

Super Mario World

SMW steht für **"Super Mario World"** – ein Videospiel, das in Europa im Jahr 1992 für die damals neue Spielekonsole **SNES (Super Nintendo Entertainment System)[1] erschienen ist.** Wie die jüngeren Spiele der Mario-Reihe ist auch Super Mario World ein Jump-'n'-Run – es geht ums Springen über Plattformen und auf Gegner, ums Sammeln von Powerups und Münzen und um die Rettung der Prinzessin von Koopa-König Bowser. Für jeden, der schon mal ein Mario-Spiel gespielt hat, dürfte das nichts Neues sein.

Wenn ich ab jetzt von "SMW" rede, gehe ich davon aus, dass du weißt, was das bedeutet. Die Abkürzung ist in der Hacking-Gemeinde (und sicher auch in der Videospiel-Szene allgemein) sehr geläufig.

Bild 2: Die Spielekonsole SNES

Bild 1: Das Spiel Super Mario World

ROMs und Emulatoren

Wenn man SMW spielen wollte – damals, in den frühen neunziger Jahren – musste man die Spielekonsole SNES besitzen und sich das Spiel, wenn man es noch nicht hatte, im Laden kaufen. Die Konsole hat man zu Hause an den Fernseher angeschlossen, das **Spielmodul** ("Cartridge") in die Konsole gesteckt und alles angeschaltet – und dann konnte das Spiel beginnen.

[1] Grafik von JCD1981NL und Evan-Amos bei Wikipedia - http://de.wikipedia.org/w/index.php?title=Datei:SNES-PAL-Console-Set.png&filetimestamp=20110218025939

Heute sind SNES und Spielmodule veraltet, und es gibt einfachere Möglichkeiten, Spiele wie SMW zu spielen (und vor allem billigere). Seit Jahren schon ist es nämlich möglich, Spiele für die SNES-Konsole auch **auf dem Computer zum Laufen zu bringen**.

Wie das möglich ist? Mit einem Computerprogramm, das man **Emulator** nennt. Die Aufgabe eines Emulators ist es, **andere Systeme nachzubilden** (zu simulieren) – so zum Beispiel Spielekonsolen. Das bedeutet, wenn man den Emulator auf dem Computer ausführt, **verhält er sich wie eine Konsole**, und man kann auf ihm Spiele laufen lassen, als besäße man das echte Gerät. Spielekonsolen, die schneller sind als heutige PCs (etwa die Wii von Nintendo) kann der Emulator für den Computer natürlich nicht ganz so gut nachbilden, aber bei älteren Konsolen läuft in der Regel alles rund.

Für eine ganze Reihe von Geräten existieren Emulatoren – allein für die SNES gibt es mindestens drei (und sicher noch einige mehr, die weniger bekannt sind).

Wie kann man nun ein altes Konsolenspiel auf dem Emulator ausführen? Das alte Spielmodul kann man schließlich nicht in den Computer schieben. Nein, fast alle alten Spiele liegen mittlerweile auch **in Form von Computerdateien vor**. Diese Dateien werden **ROM**s genannt ("ROM" steht für "Read-Only Memory", wie auch in "CD-ROM"). ROMs enthalten **alle Daten des echten Spielmoduls** (den Programmcode, die Grafiken und all das), nur eben nicht als greifbarer Gegenstand, sondern als Datei wie jede andere auch. Wie ein Bild oder Dokument auf dem Computer eben.

Solche ROMs kann man mit dem Emulator öffnen, so als würde man das echte Spielmodul in die Konsole schieben. Dann kann man das Spiel **auf dem Computer spielen**, wobei einem die Tastatur als Controller und der Bildschirm als… Bildschirm dient. Es gibt einige Funktionen, die nur der Emulator hat, zum Beispiel das Speichern des momentanen Zustandes zu jedem beliebigen Zeitpunkt ("Save States"). Darum wollen wir uns allerdings nicht kümmern, schließlich soll das hier kein Buch über Emulatoren werden.

Der Emulatoren-"Krieg"

Eine Sache zum Thema Emulatoren muss ich allerdings noch erwähnen, bevor wir vollends zum SMW-spezifischen Teil kommen. (Ich sagte ja, das Lesen wird dir Mühe machen…)

Wie bereits erwähnt, es gibt eine Menge verschiedener SNES-Emulatoren. Die bekanntesten und meistverwendeten dürften die folgenden drei sein: **ZSNES, snes9x und bsnes**.

Nicht alle Emulatoren sind allerdings gleich gut. Ein Emulator soll ein vorhandenes System möglichst genau nachahmen können – ein wichtiges Kriterium für einen Emulator ist also seine **Genauigkeit**. Wenn du einen weniger guten Emulator verwendest, der sich bei der Simulation der SNES einige Schlampigkeiten erlaubt, musst du damit rechnen, dass einige Spiele fehlerhaft oder gar überhaupt nicht ausgeführt werden. Ein guter Emulator dagegen verhält sich ganz genau wie die Spielekonsole und macht auf Bruchteile einer Sekunde

genau alles exakt gleich, so dass man keinen Unterschied zwischen Emulator und Konsole feststellen kann. Merke: **Nur auf einem guten Emulator läuft jedes Spiel fehlerfrei.**

ZSNES[2] ist ein Beispiel für einen nicht so genauen Emulator – es ist eine ganze Reihe von Fehlern bekannt, und viele SNES-Spiele sind mit ZSNES eingeschränkt oder gar nicht spielbar.

bsnes[3] ist dagegen ein Musterbeispiel für Genauigkeit. Er wurde extra mit dem Ziel entwickelt, so genau wie möglich die SNES zu simulieren – und das mit Erfolg, denn momentan ist kein einziger Fehler bekannt. Zudem wird an bsnes immer noch aktiv gearbeitet – im Gegensatz zu ZSNES, dessen letzte Version im Jahr 2007 erschienen ist. Ein "Genauigkeits-Manifest" mit einer Auswahl an Fehlern, die in ungenaueren Emulatoren auftreten, findest du auf der Seite von byuu, dem Entwickler von bsnes[4] (Englischkenntnisse sind hilfreich).

snes9x[5] ist so eine Art Mittelding – mit der neuesten Version wurden viele Fehler behoben, aber einige Ungenauigkeiten gibt es dennoch.

Wir halten fest: In ungenauen Emulatoren können Fehler beim Ausführen von Spielen auftreten. Aber auch der umgekehrte Fall ist möglich: Wenn man ein Spiel erstellt und es nur in ZSNES testet, besteht die Gefahr, dass es auf einer echten Konsole nicht richtig funktioniert. Das erkennt ein ungenauer Emulator wie ZSNES nicht, und so merkt man vielleicht gar nicht, dass mit dem Spiel was nicht stimmt. Die Folge: Das Spiel ist nur in ZSNES spielbar, und wenn irgendwann in der Zukunft sogar ZSNES seine Fehler behebt, dann kann man es **gar nicht mehr spielen**. Du solltest also immer darauf achten, dass dein Spiel auf einer echten SNES funktioniert, und das tut man am besten indem man es **auch in bsnes testet** – keine unnötige Arbeit, sondern **eine Investition in die Zukunft**.

(Es herrscht vielleicht kein "Krieg" zwischen den verschiedenen Emulatoren, wie es die Überschrift andeutet, aber mindestens ein Wettbewerb, in dem man als Endnutzer sogar mitentscheiden kann.)

Welchen Emulator solltest du also nehmen? Nach den letzten Absätzen sollte klar sein, wie meine Empfehlung ausfällt: **bsnes, denn nur das ist zukunftssicher**. Ich will aber nicht verschweigen, dass auch ZSNES seine Vorteile hat: Meiner Meinung nach ist es einfacher zu bedienen, denn ich habe mit ZSNES angefangen und mich einfach an seine Benutzeroberfläche gewöhnt (jemandem, der sich erst orientiert, mag es da anders gehen). Außerdem ist es aufgrund seiner Schlampigkeit schneller als bsnes, mit dem ältere PCs möglicherweise Schwierigkeiten haben. Zum Spielen der meisten Spiele reicht mir ZSNES, und zugegeben ist es immer noch mein Stamm-Emulator. Zumindest zum Testen seiner Spiele sollte man aber auf jeden Fall bsnes verwenden.

---

[2] http://www.zsnes.com
[3] http://www.byuu.org/bsnes
[4] http://www.byuu.org/bsnes/accuracy
[5] http://www.snes9x.com

Praxiserfahrung – die Suche nach ROMs

Bevor ich aufs SMW-Hacken gestoßen bin, hatte ich von Konsolen und deren Spielen äußerst wenig Ahnung, und besessen oder gespielt hatte ich sie erst recht nicht. Das erste, was ich getan habe, als ich von Emulatoren gehört und eine ROM gefunden hatte, war, **SMW komplett durchzuspielen** – ich wollte ja erst mal wissen, worauf ich mich da einlasse, schließlich war es mein erstes offizielles Mario-Spiel überhaupt.

Wenn es dir so geht wie mir damals und du, aus welchen Grund auch immer, noch keine Gelegenheit hattest, SMW zu spielen, kann ich dir an dieser Stelle nur sehr empfehlen, das jetzt nachzuholen! Wenn du nicht weißt, wie man SMW spielt, wirst du auch beim Hacken keine Freude haben, und Schwierigkeiten sowieso. Und wenn du das Spiel schon kennst und zu Hause auf der Konsole gespielt hast, dann probier doch jetzt mal aus, wie es sich auf einem Emulator anfühlt.

Also dann – lade dir einen **Emulator** deiner Wahl herunter, besorge dir eine **ROM** von SMW, **öffne sie** im Emulator und **lege los!** Wie man spielt, soll nicht Thema dieses Tutorials sein, und wie man Sachen herunterlädt, muss ich dir hoffentlich auch nicht erklären.

Das einzige Problem, auf das du dabei eventuell stößt, ist die Frage **"Wo bekomme ich denn eine ROM her?"** Im Laden kaufen kannst und musst du sie natürlich nicht – wie gesagt, es ist eine ganz normale Datei für den Computer. Im Internet gibt es eine ganze Menge Seiten, die ROMs von allen möglichen Spielen für alle möglichen Konsolen kostenlos zum Download anbieten. Welche Seiten das sind, verrate ich dir hier mal lieber nicht – nur, um ganz sicher zu gehen.

Die Legalität des Ganzen ist ein Thema, auf das ich gleich eingehen werde - aber ich kann dir schon mal sagen, es ist ein kompliziertes, und ich habe selber ziemlich wenig Ahnung davon. Für den Anfang gilt einfach: **Mach dir keine Sorgen**.

Und was die ROM-Seiten angeht, die kannst du auch problemlos selber finden. Suche einfach bei Google (oder einer anderen Suchmaschine deiner Wahl) nach Begriffen wie "snes roms", "super mario world rom download" oder so etwas. Wenn du dann auf eine Datei stößt, die "**Super Mario World (U) [!].smc** heißt", dann kannst du dir so gut wie sicher sein, dass das die SMW-ROM ist, die du suchst. Das **U** bedeutet, dass es sich um die US-amerikanische Version des Spiels handelt (das ist beim Hacken wichtig), und das **[!]** bedeutet – so viel ich weiß – dass die ROM in Ordnung und spielbar ist. **.smc** ist eine geläufige Dateiendung für SNES-ROMs. Wenn sich die ROM doch nicht fehlerfrei spielen lässt, dann lade von einer anderen ROM-Seite eine herunter – wie gesagt, es gibt mehr als genug davon.

Also dann, hast du ROM und Emulator beisammen? Worauf wartest du dann noch? Spiel dich erst mal ein bisschen ein, damit du ein Gefühl fürs Gameplay bekommst.

Das ist SMW-Hacking! - eine genauere Beschreibung

Jetzt hast du schon eine Menge Vorwissen gesammelt - du weißt, was ein Emulator ist, welchen du benutzen willst, was eine ROM ist, wo man sie herbekommt, und du hast dich auch schon mit dem Spielen von SMW vertraut gemacht. Jetzt wird es endlich Zeit, auf das Thema zurückzukommen, um das es in diesem Tutorial ja eigentlich geht – das **SMW-Hacken.**

Eine ROM, wie du weißt, liegt auf dem Computer als ganz normale Datei vor. Die ROM kann man herunterladen, kopieren und löschen wie jede andere Datei auch – und natürlich kann man sie wie jede andere Datei **bearbeiten**. Und darauf kommt es an! Die ROM enthält je bekanntlich die Daten des Spiels – den Programmcode, die Grafiken, die Musik und all das. Die ROM zu verändern, heißt also, **das Spiel zu verändern**. Kurz gesagt: **Man kann SMW verändern und sein eigenes Spiel daraus machen!** Und genau darum geht es beim SMW-Hacken. Indem du ein anderes Spiel als Grundlage benutzt, kannst du sehr schnell und einfach dein eigenes Spiel erstellen. Das ist doch was, oder? Die Spiele, die durch das Verändern einer ROM entstehen, werden **Hacks** genannt, und die Spiele, die aus SMW gemacht sind, dementsprechend **SMW-Hacks**.

Bild 3: Ein Level in Super Mario World

Hacken halte ich für die einfachste Methode, selbst ein Spiel zu erstellen, eben **weil das Fundament schon da ist**. Das Originalspiel stellt schon die Grundlagen wie Grafiken, Musik und Engine bereit, und das nicht alles selber zusammensuchen und programmieren zu müssen, ist eine echte Erleichterung. Du als Hacker kannst sofort kreativ werden und mit dem Bauen von Leveln loslegen.

SMW ist natürlich bei weitem **nicht das einzige Spiel, das man hacken kann** – man kann schließlich jede Datei bearbeiten, also kann man auch jedes Spiel, das als ROM vorliegt, verändern. Von allen Spielen ist SMW aber mit Abstand **am einfachsten** zu hacken! Wie du vielleicht weißt, besteht jede Datei eigentlich nur aus Folgen von Nullen und Einsen, die der Computer verstehen und interpretieren kann. Man könnte nun eine beliebige ROM nehmen und einfach von Hand die Anordnung der Nullen und Einsen verändern, um das Spiel zu modifizieren, aber das ist nicht nur eine Heidenarbeit, sondern auch extrem kompliziert. So was macht niemand, der bei klarem Verstand ist (und der eine Wahl hat).

Für viele Spiele gibt es daher spezielle **Hacking-Programme**, die nette und schlaue Leute geschrieben haben, um das Spielemachen so einfach wie möglich zu gestalten. Da gibt es zum Beispiel Tools, die das Bauen von Leveln oder das Bearbeiten von Grafiken erleichtern, oder sogar Tools, die den Programmcode des Spiels verändern, damit du beispielsweise neue Musik ganz leicht einfügen kannst. Und **das SMW-Hacken ist deshalb so einfach,**

weil für SMW eine Unmenge an Hacking-Programmen existiert. Es gibt eine große Community, die sich mit dem SMW-Hacken beschäftigt, der Aufbau des Spiels ist so gut wie vollständig erforscht, und für beinahe jeden Aspekt des Spiels wurden nützliche Tools programmiert, die das Hacken von SMW wirklich zum Kinderspiel machen.

Welche genau das sind, das erfährst du gleich – keine Bange, in diesem Tutorial werde ich auf alle nötigen Tools eingehen. Erst einmal hoffe ich, dass ich nach all dem Vorgeplänkel deine Frage, was "SMW-Hacken" denn nun eigentlich ist, beantwortet habe – und dass du an der ganzen Sache vielleicht Gefallen gefunden hast.

Hacking und Legalität

Wie ich vorhin schon kurz erwähnt habe, stellt sich vielen ROM-Hackern irgendwann die Frage der Legalität. "Hacken", das klingt schon so bedrohlich, und verboten sowieso – darf man das überhaupt? Darf ich meine Hacks denn überhaupt veröffentlichen? Wie gesagt, **die Rechtslage ist kompliziert**, zumindest für Laien wie mich, und überall hört man Widersprüchliches dazu. Ich selbst habe von Rechtsdingen gar keine Ahnung, aber das Bisschen, was ich weiß, will ich dir hier näher bringen.

(Ich bin kein Jurist und kein Anwalt, und das hier ist keine Rechtsauskunft.)

Klar, so ein Spiel war nicht immer als ROM zum kostenlosen Download verfügbar. Früher gab es sie nur als handfeste Spielmodule, und die zu erwerben kostete handfestes Geld. Nach allem, was ich weiß, ist das Copyright, das Nintendo an SMW hat, noch längst nicht abgelaufen, also kann man das Herunterladen einer ROM durchaus als **Produktpiraterie** bezeichnen. Ich glaube allerdings nicht, dass es für Nintendo ein großer Verlust ist, wenn man für ein zwanzig Jahre altes Spiel, das es sowieso nicht mehr im Laden zu kaufen gibt, nichts mehr bezahlt. (Vielleicht verdient Nintendo noch an SMW-Versionen für die Wii oder so, das wissen andere besser als ich.)

Sowieso ist **in anderen Ländern auch die Rechtslage anders**. Mancherorts ist das Besitzen von ROMs angeblich legal, wenn man auch das Originalspiel besitzt, anderswo vielleicht das Besitzen, aber nicht das Verbreiten, oder das Verbreiten nur, wenn man dem Endnutzer empfiehlt, die ROM nur zu spielen, um herauszufinden, ob man sich das echte Spiel kaufen möchte… ich weiß auch nicht. Das alles ist jedenfalls vorstellbar.

Ich hoffe, ich habe deutlich genug rübergebracht, dass das **Hacken selbst nicht illegal** ist – das Verändern einer Datei ist ja schließlich nicht strafbar. Das kann ich gar nicht genug betonen. **Fragwürdig ist nur das Besitzen und Verbreiten von ROMs.**

In all dem Trubel bleibt eine nicht unwichtige Sache anzumerken: **Mach dir keine Sorgen.** ROMs herunterzuladen ist vielleicht nicht die legalste Sache der Welt, aber mit Sicherheit auch nicht keine Straftat von höchster Priorität. Ich glaube, die Polizei hat Wichtigeres zu tun, als ROM- oder Hacking-Seiten zu durchforsten, herauszufinden, ob du ROMs besitzt und

kurze Zeit später mit Blaulicht und Handschellen vor deiner Tür stehen, um dir eine Haftstrafe aufzubrummen. Selbst Nintendo ist die Sache ziemlich egal – es kann als sicher gelten, dass Nintendo von der ROM-Hacking-Szene Wind bekommen hat, und trotzdem unternehmen sie nichts dagegen. So weit ich weiß, ist ihr Argument, dass sie begeisterte Fans ihrer Spiele nicht kriminalisieren wollen, und das leuchtet ein.

Der vorige Absatz ist nicht als Freibrief zu verstehen – ich will dir nicht explizit *raten*, das Gesetz zu brechen, aber ich bin auch davon überzeugt, dass man **nichts zu befürchten** hat, wenn man hackt. (Und ich glaube, die meisten Leute in meinem – und deinem – Alter geben eh einen Dreck auf das Gesetz. Bei Rot über die Straße gehen, mit fünfzehn Jahren Bier kaufen, nach 22 Uhr alleine in Restaurants und Kneipen sitzen – derlei Sachen eben.) Hätte es irgendwann mal einen Fall gegeben, in dem ein ROM-Hacker für den Besitz von ROMs bestraft wurde, hätte man davon gehört, da bin ich mir sicher.

Auch **der Begriff "Hacken**" mag dich erst mal einschüchtern, was die Legalität oder die Sicherheit des Ganzen angeht, vermitteln einem die Medien schließlich ein ziemlich bedrohliches Bild von Leuten, die sich "Hacker" nennen: düstere Gestalten in dunklen Räumen, die den ganzen Tag nichts anderes tun, als auf ihre Tastatur einzuhämmern, Viren zu verbreiten, das Netz lahmzulegen und die Welt ins Chaos zu stürzen. Wie dir hoffentlich schon klar geworden ist – **damit hat SMW-Hacken nichts am Hut**. Hacken muss nicht immer etwas mit Netzwerken und Verbrechern zu tun haben. Für mich bedeutet der Begriff ganz allgemein **"sich Zugriff zu etwas verschaffen, was man nicht sehen soll oder muss, und dort Dinge verändern"**. Insofern passt "Hacken" doch ganz gut - man hackt sich in den Code und die Daten von SMW rein.

Warum verlinke ich nicht auf ROM-Seiten und gebe selbst keine weiter? Ich weiß einfach nicht genau, was das Gesetz nun erlaubt und was nicht, und da gehe ich halt auf Nummer sicher. (Das sehen und machen auch die großen Hacking-Seiten im Internet nicht anders.) Sollte aber kein großes Hindernis darstellen, denn wie gesagt, es ist überhaupt kein Problem, ROM-Seiten selbst zu finden. Irgendwie wirst du garantiert an ROMs kommen.

Fazit: **Die Sache ist kompliziert, aber mach dir keine Sorgen.**

Patches – Gedichte und mehr sorglos verbreiten

Nehmen wir an, du hast einen SMW-Hack erstellt und willst ihn nun im Internet veröffentlichen. Du willst die SMW-ROM, die du bearbeitet hast, hochladen, aber da fällt dir ein, dass auf den großen Hacking-Seiten das Hochladen oder Verlinken von **ROMs gar nicht gerne gesehen** ist.

Das ist deshalb so, weil man beim Hacken ja auf einem bestehenden Spiel aufbaut. Man verändert eine Menge an dem Spiel, aber große Teile des Programmcodes, und oft auch die Grafiken und die Musik, bleiben erhalten. Diese Dinge sind allerdings **nicht von dir gemacht** worden, sondern von Nintendo, und die haben schließlich das Copyright darauf. Es

gibt einige Leute, die **komplette ROMs selber machen** (solche Spiele nennt man dann "Homebrew") – in diesen ROMs ist kein geschützter Code oder Ähnliches von anderen Leuten enthalten, daher darf man diese auch ohne Weiteres verbreiten. **Hacks allerdings enthalten ja noch Teile des geschützten Originalspiels, daher ist es nicht gern gesehen, diese ROMs weiterzugeben.**

Ja verdammt, denkst du dir, wenn man keine ROMs hochladen darf, wie kann man denn sein Spiel überhaupt verbreiten? Man muss doch die ROM weitergeben, denn das ist ja schließlich die Spieldatei! Wie blöd kann man denn sein, ROMs zu verbieten?

…oder gibt es etwa noch **eine andere Möglichkeit, Hacks der Öffentlichkeit zugänglich zu machen**? In der Tat, die gibt es – und, oh Freude, **sie ist vollkommen legal**!

Um zu verdeutlichen, wie das funktioniert, entfernen wir uns kurz von Dateien und Hacking und wählen ein anschaulicheres (wenn auch ziemlich albernes) Beispiel. Nehmen wir an, dir liegt folgendes Gedicht vor:

> *Alle meine Entchen*
> *Schwimmen auf dem See*
> *Köpfchen in das Wasser*
> *Schwänzchen in die Höh'*

(Ja, es ist eher ein Lied als ein Gedicht, und vielleicht kennst du eine Variante, in der es "unters Wasser" heißt, aber das spielt hier keine Rolle.)

Angenommen, dieses Gedicht kostet Geld, und es zu verbreiten, wenn man nicht der Urheber ist, wäre verboten. Du hast dir dieses Gedicht jetzt beschafft und es ein wenig abgewandelt, um dein eigenes daraus zu machen:

> *Alle meine **Gänse***
> *Schwimmen auf dem **Teich***
> *Köpfchen in das Wasser*
> *Schwänzchen in die Höh'*

(Jetzt reimt es sich nicht mehr, aber auch das tut nichts zur Sache.)

Dein so entstandenes neues Gedicht willst du gerne veröffentlichen, aber du stellst fest, dass du das gar nicht darfst – schließlich stammen 13 der 15 Wörter nicht von dir, sondern von dem Autor des Original-Gedichtes.

Was machst du also? Wenn du nicht das ganze Gedicht veröffentlichen darfst, dann eben **nur die Teile, die am Original verändert wurden**! Du veröffentlichst also folgenden Satz:

> *Nimm das Gedicht "Alle meine Entchen" und ersetze das dritte Wort durch "Gänse" und das siebte Wort durch "Teich".*

Das Tolle daran: Du hast das Original-Gedicht nicht verbreitet und kein Wort daraus erwähnt (nur, welches Gedicht es ist), aber trotzdem kann jeder nachvollziehen, wie dein verändertes Gedicht aussieht. Die Voraussetzung ist allerdings, dass der Empfänger das Original-Gedicht hat – aber wie er daran gekommen ist, ist seine Sache, nicht deine. Du hast **nur die Änderungen am Original verbreitet**, nicht das ganze Werk.

Dass der Empfänger das Original besitzt, ist wichtig, denn sonst kann er mit den Änderungen, die du veröffentlicht hast, auch nichts anfangen. Es muss auch genau dasselbe Original sein, denn sonst kommt am Ende etwas ganz Anderes heraus. Angenommen, der Empfänger nimmt als "Original" folgendes Gedicht:

> *Alle Jahre wieder*
> *Kommt das Christuskind*
> *Auf die Erde nieder,*
> *wo wir Menschen sind*

Wendet er die von dir hochgeladenen "Anweisungen" auf sein Gedicht an, bekommt er natürlich nicht das gewünschte Resultat:

> *Alle Jahre **Gänse***
> *Kommt das Christuskind*
> ***Teich** die Erde nieder,*
> *Wo wir Menschen sind*

Das zu schreiben war natürlich nicht deine Absicht, und sinnvoll ist das Ergebnis auch nicht gerade. Du siehst: Nur die Änderungen hochzuladen hat den Vorteil, dass das Verbreiten vollkommen unbedenklich ist, aber den Nachteil, dass man das Original haben muss, um das richtige Ergebnis zu bekommen. (Das ist dann aber nicht dein Problem!)

Ich hoffe, dieses Beispiel war nicht zu… abgedreht… ich wollte eben klar machen, worum es geht. Wichtig ist: **Genau so macht man es auch mit Hacks!** Man veröffentlicht nicht die ROM selbst, sondern eine **Datei, in der nur die Änderungen am Originalspiel notiert sind**. Diese Datei kann man völlig problemlos weitergeben, denn in ihr steht ja nicht der Code des Originalspiels drin.

Die Datei mit den Änderungen wird **Patch** genannt. Sie kann sich dann jeder ohne Weiteres herunterladen. Er kann allerdings erst etwas damit anfangen, wenn er auch das Originalspiel hat, also muss auch er eine ROM von irgendwo her bekommen. Außerdem braucht er ein **Patch-Programm ("Patcher")**, das diese Änderungen überhaupt vornimmt.

Hat er Patch, ROM und Patcher beisammen, teilt er dem Patcher mit, auf welche Datei er die im Patch notierten Änderungen anwenden soll (also die ROM), und dann wird **gepatcht**. Nach dem Anwenden des Patches auf die ROM ist diese nun **zum dem Spiel geworden**,

das man erstellt hat, und wenn der Empfänger die ROM im Emulator öffnet, kann er dein Spiel spielen!

Wenn du einen Patch erstellen willst, machst du im Grunde dasselbe, nur umgekehrt: Du öffnest den Patcher, teilst ihm mit, welche Datei du als "Original" verwenden willst (die Original-ROM von SMW) und aus welcher Datei die Änderungen kommen sollen (deine neue ROM mit deinem eigenen Spiel). Dann sucht der Patcher nach den Unterschieden zwischen dem Original und deiner neuen ROM und schreibt diese Unterschiede in eine neue Datei – den Patch.

Patches sind nicht nur fürs ROM-Hacken gut - als "Original" kannst du jede Datei nehmen, die du willst. Du kannst auch tatsächlich Textdateien verwenden und, genau wie im Beispiel, aus einem Gedicht mit einem Patch ein anderes machen. Die beiden Dateien müssen nicht einmal vom selben Typ sein – mit einem Patch kannst du auch aus einem Bild ein Musikstück machen, oder aus einer Textdatei eine ROM. **Alles ist möglich!**

Wichtig ist aber, dass **im Zusammenhang mit Hacking** Patches **unbedingt eine ROM als Ausgangsdatei** brauchen. Warum das? Nun, vor einigen Jahren hatte ich selbst mal folgende Idee: *"Warum nimmt man aus Ausgangsdatei nicht einfach eine leere Textdatei? Man kann ja problemlos einen Patch erstellen, der aus dieser leeren Datei eine ROM macht, und das hat den Vorteil, dass jeder sich so eine leere Datei machen kann. Dann braucht man als Empfänger keine ROM mehr, um das Spiel zu spielen!"*

Dabei hatte ich allerdings eine Sache nicht bedacht, die dir möglicherweise schon aufgefallen ist: Wenn man eine leere Datei aus Ausgangspunkt verwendet, muss der Patch ja die Daten der gesamten ROM enthalten, denn die Unterschiede zwischen einer leeren Datei und einer ROM sind… alles.

Um noch mal auf das Beispiel mit den Gedichten zurückzukommen - Wenn du als Original ein Gedicht ohne Worte verwendest, sehen die Änderungen, die du veröffentlichst, so aus:

> *Nimm ein leeres Gedicht und ersetze seinen Inhalt durch "Alle meine Gänse schwimmen auf dem Teich, Köpfchen in das Wasser, Schwänzchen in die Höh'".*

In dieser Anweisung sind nicht mehr nur die beiden Wörter enthalten, die du am Original verändert hast, sondern auch die 13 Wörter, die du aus dem Original übernommen hast. Mit anderen Worten, du verbreitest Teile des Gedichts, die nicht von dir stammen – das ist genau so, als würdest du die ganze ROM verbreiten. Daher sollte man bei Hacks immer darauf achten, als Ausgangspunkt eine ROM desselben Spiels zu verwenden, denn die hat mit deinem neuen Spiel einiges gemeinsam.