# Transparenz von Smart Contracts

Beurteilung der Transparenz von Blockchain-basierten Smart Contracts

## A. Einleitung

Smart Contracts bieten eine neue Qualität der Transparenz der automatischen Vertragsausführung. Während klassische Vertragsautomatisierung eine Machtverschiebung zu Gunsten der automatisierenden Vertragspartei bedeutet, liegen Smart Contracts auf einer Blockchain auf neutralem Grund zwischen den Parteien. Eine spätere Manipulation, wie sie es z.B. DRM-Systeme ermöglichen, ist dort ausgeschlossen. Doch ist diese theoretisch bessere Transparenz effektiv vorhanden? Lässt sich der Quellcode eines Smart Contracts auf einer Blockchain sicher verifizieren und garantiert dies einen Schutz vor unerwarteten Ergebnissen?

## B. Smart Contracts

Der Begriff *Smart Contracts* wird auf *Nick Szabo* zurückgeführt. Er definierte 1993/94 Smart Contracts als ein *computerized transaction protocol that executes the terms of a contract*.[1] Damit wäre bereits jede automatisierte Vertragsausführung ein *Smart Contract*. Jeder Verkaufsautomat, Geldautomat, der Appstore, jedes DRM-System oder selbst eine Schranke im Parkhaus wären Smart Contracts. In den letzten drei Jahren prägten dann vor allem *Vitalik Buterin* und die Ethereum-Blockchain den Begriff. Im Folgenden beschränkt sich die Betrachtung auf Smart Contracts auf dieser Blockchain.

Ethereum bietet die Möglichkeit, die Ausführung von Transaktionen auf der Blockchain über kleine Computerprogramme zu steuern.[2] Während *Buterin* zunächst nur davon sprach, dass mit Ethereum-Skripten Smart Contracts realisiert werden können, wird inzwischen jedes auf der Ethereum-Blockchain ausgeführte Skript als *Smart Contract* bezeichnet.[3] Diese kleinen Computerprogramme sind bereits auf Grund ihrer begrenzten Größe nicht besonders intelligent und haben auch nicht zwangsläufig einen Bezug zu juristischen Verträgen. Sie bieten jedoch im Vergleich zur obigen Definition zwei neue Eigenschaften: Sie liegen als Code offen einsehbar auf einer Blockchain. Zudem gibt es eine technische Sicherheit, dass der Code so, wie er einmal abgespeichert wurde, auch ausgeführt wird. Damit wird aus der einseitig automatisierten Vertragsausführung ein neutraler Mittler, der gleich einem Treuhänder die korrekte Vertragsausführung sicherstellt.

Die Vertragsfreiheit erlaubt generell auch die Vereinbarung einer Programmiersprache als Vertragssprache (§ 311 Abs. 1 BGB)[4]. Smart Contracts können jedoch als Allgemeine Geschäftsbedingungen Restriktionen unterworfen sein. Dazu müssen sie von der einen Vertragspartei der anderen Vertragspartei vorgegeben sein (§ 305 Abs. 1 S. 1 BGB). Bei von

---

[1] *Nick Szabo*, im Internet publiziert unter http://szabo.best.vwh.net/smart.contracts.html, seit Mitte 2016 nicht mehr online. Über das Wayback-Archiv: https://web.archive.org/web/20160620185026/http://szabo.best.vwh.net/smart.contracts.html, alle Links zuletzt aufgerufen am 23. Juli 2018.

[2] *Vitalik Buterin*, Ethereum White Paper, https://www.weusecoins.com/assets/pdf/library/Ethereum_white_paper-a_next_generation_smart_contract_and_decentralized_application_platform-vitalik-buterin.pdf.

[3] So etwa in der Einführung in Ethereum-Skripts https://ethereum.org/greeter.

[4] So etwa *Djazayeri*, jurisPR-BKR 12/2016 Anm. 1 E II; Kaulartz InTer 2016, 201ff., 204.

Dritten zur allgemeinen Verwendung bereitgestellten Smart Contracts wird dies möglicherweise nicht der Fall sein[5]. Ist ein Smart Contract als Allgemeine Geschäftsbedingung zu betrachten, gelten die Regeln der §§ 305ff. BGB, die insbesondere in § 307 Abs. 1 S. 2 BGB eine gesteigerte Transparenzpflicht beinhalten. Doch auch wenn die Hürde zur Annahme von Allgemeinen Geschäftsbedingungen nicht erreicht sein sollte, können Vertragsbedingungen, die nicht zur Kenntnis genommen werden können, kein Gegenstand eines Vertragsschlusses sein.

## C. Transparenz

Wikipedia definiert *Transparenz* fachspezifisch: Für die Physik, Computergrafik, Akustik, Computersysteme, Signalverarbeitung, Politik und Wirtschaftswissenschaften werden eigene Definitionen angeboten[6]. Im Recht scheint der Begriff „Transparenz" zum Modewort geworden zu sein. Die DSGVO verwendet ihn häufig und definiert ihn im Erwägungsgrund 58. Dabei geht es darum, dass Informationen präzise, leicht zugänglich und verständlich sein sollen.

Das „Transparenzgebot" in § 307 Abs. 1 S. 2 BGB ist Umsetzung von Art. 4 Abs. 2 RL 93/13/EWG. Es sieht vor, dass Klauseln in Allgemeinen Geschäftsbedingungen „klar und verständlich" formuliert sein müssen. Das seien sie – so der EuGH – wenn die betroffenen Verbraucher in der Lage sind, die sich für sie aus einer Klausel ergebenden wirtschaftlichen Folgen auf der Grundlage genauer und nachvollziehbarer Kriterien einzuschätzen[7].

In Teilbereichen kann es zwischen präzisen, vollständigen aber schwer verständlichen Informationen auf der einen Seite und einfach verständlichen aber inhaltlich unvollständigen Informationen einen Zielkonflikt geben. Als Folge daraus aber nur eine Ebene zu bedienen und z.B. aus Gründen der Verständlichkeit nicht alle Informationen offen zu legen, wird dem Transparenzgebot jedoch nicht gerecht[8]. Vielmehr sollte in solchen Fällen eine einfache Darstellung der Grundprinzipien neben die detaillierte Darstellung gestellt werden. Schließlich verzichten wir ja auch in juristischen Begründungen nicht auf komplexe juristische Darstellungen, nur weil die Betroffenen diese ggf. nicht alleine verstehen könnten. Die Details sind dann wichtig, wenn sich die Betroffenen z. B. mit professioneller Unterstützung effektiv gegen sie belastende Maßnahmen wehren wollen.

Die Definition von „Transparenz" in der Informatik ist konträr zur juristischen Verwendung des Begriffs. Unsichtbare Systemkomponenten werden dort als transparent bezeichnet[9]. Wenn im Folgenden von der Transparenz von Smart Contracts gesprochen wird, ist die juristische Bedeutung gemeint. Als Kriterien für die Transparenz werden im Folgenden daher sowohl die Zugänglichkeit von Informationen als auch deren Verständlichkeit gesehen.

---

[5] Vgl. dazu die Ausführungen zum Reisefrachtvertrag: Münchener Vertragshandbuch Bd. 4 WirtschaftsR III, VIII. See- und Luftfrachtrecht 3. Reisefrachtvertrag (Gencon-Charter).

[6] Wikipedia, Transparenz, https://de.wikipedia.org/wiki/Transparenz.

[7] EuGH, Urt. v. 30.4.2014 – C-26/13.

[8] a. A. wenig nachvollziehbar Häuser, LTO 12.7.2018, Müssen sich Unternehmen in ihre Algorithmen schauen lassen?, https://www.lto.de/recht/kanzleien-unternehmen/k/dsgvo-ki-geschaeftsgeheimnisse-unternehmen-schutz-offenlegung/.

[9] Wikipedia, Transparenz (Computersystem), https://de.wikipedia.org/wiki/Transparenz_(Computersystem).

# D. Transparenz von Smart Contracts

Ein Snack-Automat, bei dem die einzelnen Produkte klar gekennzeichnet und mit Preisen versehen sind, könnte als „transparent“ angesehen werden. Allerdings gibt es keine Gewähr, dass die gewünschte Ware nach Einwurf der passenden Geldmenge tatsächlich ausgegeben wird. Die transparente Darstellung stimmt mit der automatisierten Vertragsdurchführung nicht automatisch überein. Bei Smart Contracts auf der Blockchain ist dies anders. Dort ist technisch sichergestellt, dass der offengelegte Code genauso ausgeführt wird. Allerdings ist die Darstellung des Codes komplexer als die Präsentation von Ware und Preis beim Snackautomaten.

Man könnte versucht sein, die einfache Darstellung als die alleinige rechtlich relevante Darstellung zu sehen. Dies ist jedoch meistens nicht der Fall. Zum einen gelten häufig AGB, die in der Regel nicht mehr so einfach verständlich sind. Zum anderen kann auch ein einfacher Vertrag mit all seinen Folgen nur mit entsprechenden juristischen Hintergrundwissen erfasst werden. Ein programmierter Vertrag muss daher eine gewisse Komplexität aufweisen um z.B. überhaupt mit zwingendem Vertragsrecht vereinbar zu sein. Die notwendige Komplexität sollte daher der Gültigkeit eines Smart Contracts nicht entgegengehalten werden können – zumindest soweit diese Komplexität Folge des unabdingbaren Rechts ist, welches ein Smart Contract auch implementieren muss.

Smart Contracts bieten eine neue Form von Transparenz, die auf klassischem Weg geschlossene juristische Verträge meistens nicht bieten:

- Der Vertragsabschluss ist im Detail nachvollziehbar und unveränderlich dokumentiert.
- Der Vertragsinhalt ist unveränderbar festgehalten.
- Leistungen können bei Vorliegen der entsprechenden Bedingungen automatisiert erfolgen.
- Die automatisierten Leistungen können von einer Vertragspartei nicht verhindert oder manipuliert werden.
- Die Vertragsausführung wird ebenfalls Schritt für Schritt manipulationssicher protokolliert.

Die Transparenz von Smart Contracts begegnet jedoch spezifische Herausforderungen:

- Auf einer Blockchain steht nicht der relativ einfach lesbare Quellcode, sondern der schwer zu verstehende Objektcode oder Bytecode.
- Softwarefehler lassen sich nicht vollständig vermeiden. Dies wird als spezifisches Problem komplexer Softwareentwicklung gesehen. Allerdings gibt es auch viele klassische juristische Verträge, in denen die gewählten Formulierungen das von den Parteien tatsächlich gewollte nicht richtig wiedergeben. Wenn bei Softwarefehlern Intention und Code auseinanderfallen, so ist dies auch ein Problem der Transparenz.
- Der Vertragsschluss erfolgt auf Basis von privaten Keys. Wer sich hinter den Keys verbirgt, kann ggf. nicht ersichtlich sein.

Im Folgenden soll auf diese Herausforderungen eingegangen und Lösungsmöglichkeiten bewertet werden.

## I. Quellcode, Objectcode und Bytecode

Vernünftig lesbar ist nur der Quellcode. Ausgeführt werden jedoch Object- oder Bytecode. Die Übersetzung von Quellcode zu Object- oder Bytecode machen Compilerprogramme. Der

umgekehrte Weg ist aufwendiger. Um vom Object- oder Bytecode auf den Quellcode zu gelangen, gibt es folgende Alternativen:

### 1. Das Compilieren nachvollziehen

Steht eine Version des Quellcodes außerhalb der Blockchain zur Verfügung, so kann dieser compiliert werden und das Ergebnis mit dem auf der Blockchain stehenden Object- oder Bytecode verglichen werden. Diese Methode setzt voraus, dass man nicht nur den Quellcode außerhalb der Blockchain hat, sondern auch die genau gleiche Compilerversion verwendet. Steht beides zur Verfügung, ist dieses Verfahren einfach und sicher. Blockchain-Explorer unterstützen diese Art der Verifizierung von Smart Contracts[10].

### 2. Decompiler

Spezielle Software kehrt den Vorgang des Compilierens um und erstellt aus Object- oder Bytecode wieder Quellcode. Der Vorteil dieser Decompiler besteht darin, dass neben der Blockchain keine weiteren Informationen benötigt werden. Allerdings gehen dabei Informationen verloren. Dem vom Decompiler generiert Quellcode fehlen z. B. bestimmte Bezeichnungen sowie Kommentare, so dass die Verständlichkeit hinter der des ursprünglichen Quellcodes zurückbleibt. Inzwischen gibt es für Ethereum mehrere solche Decompiler[11].

### 3. Ergebnis

Zu Smart Contract auf der Ethereum-Blockchain existieren inzwischen leistungsfähige Tools, mit denen verifiziert werden kann, welchen Quellcode ein Smart Contract hat.

## II. Softwarefehler

Softwarefehler in Smart Contracts können drastische Auswirkungen haben. Kryptogeldbeträge können für immer eingefroren sein wie z.B. beim Parity-Bug.[12] Auf der anderen Seite können Schwachstellen Unberechtigten Zugriff auf das von einem Smart Contract gehaltene Kryptogeld geben. Dies war z.B. beim DAO-Bug der Fall.[13] Um Bugs vorzubeugen oder sie zumindest rechtzeitig zu erkennen, gibt es neben den üblichen Qualitätssicherungstechniken für Software folgende mögliche Gegenmaßnahmen:

### 1. Formale Verifizierung von Software

Die Funktion von Software lässt sich mathematisch beweisen. Dieses Verfahren ist für Software üblichen Umfangs viel zu aufwendig. Für die sehr kurzen Smart Contracts auf einer Blockchain kann der formale Beweis jedoch möglich sein[14]. Allerdings wird selbst eine formale Verifizierung nicht alle Fehler finden. Fehler, die bereits im Vertragsdesign enthalten sind, werden möglicherweise nicht entdeckt.

---

[10] So z.B. Etherscan (https://etherscan.io) oder Etherchain (https://etherchain.org).
[11] So etwa Online Solidity Decompiler (https://ethervm.io/decompile).
[12] *Axel Kannenberg*, Kryptogeld-Wallet Parity: Bug friert Ether-Einheiten im Wert von Millionen ein, heise online, 7. November 2017, https://www.heise.de/newsticker/meldung/Kryptogeld-Wallet-Parity-Bug-friert-Ether-Einheiten-im-Wert-von-Millionen-ein-3882902.html.
[13] *Christoph Bergmann*, Die Abwicklung der DAO, BitcoinBlog.de, 30. Juni 2016, https://bitcoinblog.de/2016/06/30/die-abwicklung-der-dao/.
[14] *Bernhard Mueller*, How Formal Verification Can Ensure Flawless Smart Contracts, consensys, 29. Januar 2018, https://media.consensys.net/how-formal-verification-can-ensure-flawless-smart-contracts-cbda8ad99bd1.

### 2. Automatisch Analyse auf typische Fehler

Die Programmiersprache Solidity verhindert riskanten Code nicht automatisch, so dass Entwickler*innen selber kritische Konstrukte vermeiden müssen. Automatische Analysetools können solche Schwachstellen inzwischen erkennen und Alarm schlagen[15]. Aber auch diese Tools finden nur einen Teil der Fehler.

### 3. Weniger fehleranfällige Programmiersprachen

Solidity ist Turing-vollständig. Das bedeutet, dass sich theoretisch in Solidity jedes beliebige Programm schreiben lässt[16]. Theoretisch deshalb, da die Turing-Vollständigkeit nichts über Laufzeit und Speicherplatzbedarf aussagt. Die Turing-Vollständigkeit bedingt jedoch auch eine begrenzte Beherrschbarkeit. Daher gibt es Ansätze, Programmiersprachen für Smart Contracts zu entwickeln, die einen beschränkten Funktionsumfang haben und damit nicht nur weniger Fehlermöglichkeiten bieten, sondern auch einfacher verifizierbar[17] und für Menschen einfacher verständlich sind.

### 4. Ergebnis

Auch wenn es erfolgversprechende Ansätze gibt, die Fehlerwahrscheinlichkeit deutlich zu reduzieren, existiert bisher kein Verfahren, welches eine Fehlerfreiheit insgesamt garantiert. Software-Bugs sind daher Eventualitäten, mit denen auch zukünftig bei Smart Contracts umgegangen werden muss.

## III. Identifikation der Vertragspartner

Nicht immer ist eine Identifikation der Vertragspartner erforderlich. Im Gegenteil, die Absicherung der Transaktion durch die Blockchain hat häufig zur Folge, dass eine eindeutige Identifikation entbehrlich ist. Da sind es häufig eher gesetzliche Pflichten z.B. aus § 11 Abs. 1 Geldwäschegesetz, die eine Identifikation der Vertragspartner erzwingen.

Während es vielfältige Spuren gibt, um Vertragspartner mit einer gewissen Wahrscheinlichkeit zu identifizieren, wird zur eindeutigen Identifikation einer Person stets ein Intermediär benötigt. Dieser Intermediär muss die Identifizierung zudem Blockchain-kompatibel durchführen. Staatliche Stellen leisten dies bislang noch nicht, jedoch bieten verschiedene Startups Brücken zwischen staatlichen Identitäten und Smart Contracts an[18].

# E. Fazit und Ausblick

Smart Contracts auf der Blockchain haben gegenüber einseitiger automatisierter Vertragsausführung große Vorteile in der Transparenz. Damit diese Vorteile auch zum Tragen kommen, ist die Verifikation des Objekt- oder Bytecodes auf der Blockchain mit dem Quellcode, eine hohe Fehlerfreiheit sowie ggf. die sichere Identifikation der Vertragspartner erforderlich. Zu diesen Herausforderungen gibt es Lösungsansätze.

Dadurch, dass immer häufiger Services und nicht Eigentum Vertragsinhalt sind, sind Konsumenten gegen Eigenmacht des Vertragspartners nicht mehr ausreichend geschützt. Da diese Eingriffe häufig jedoch ähnlich schwerwiegen wie Eingriffe in das Eigentum, sollten diese

---

[15] *Karla/Goel/Dhawan/Sharma*, ZEUS: Analyzing Safety of Smart Contracts, 10.14722/ndss.2018.23092.
[16] Turing-Vollständigkeit, Wikipedia, https://de.wikipedia.org/wiki/Turing-Vollst%C3%A4ndigkeit.
[17] So etwa Seneca (https://github.com/Lamden/seneca) oder Vyper (https://github.com/ethereum/vyper).
[18] So etwa Jolocom (https://jolocom.com).

Eingriffe bei automatische Vertragsausführung zumindest an ein hohes Maß an Transparenz gekoppelt werden. Automatische Vertragsausführungen, die z.B. Fahrzeuge bei Zahlungsverzug stilllegen, sollten zukünftig nur noch zulässig sein, wenn ein Mindestmaß an Transparenz auch im Sinne von Überprüfbarkeit und Manipulationssicherheit vorhanden ist. Smart Contracts auf der Blockchain bieten damit das Potential, den durch die Überführung von Eigentum in Services verlorengegangenen rechtlichen Schutz durch technische Innovation zu kompensieren.

Doch auch ein optimal transparenter Smart Contract auf einer Blockchain muss damit umgehen können, dass die Vertragsparteien etwas Anderes meinten als tatsächlich im Vertragscode steht. Im Sinne von falsa demonstratio non nocet muss dann die Vertragsausführung angepasst werden. Ähnliches gilt, wenn der Vertragscode gegen zwingendes Recht verstößt. Beides erfordert transparent einzubindende Dispute Resolution Mechanismen, um diese Konflikte zu lösen.