media/rep/

Repositorium für die Medienwissenschaft

Oliver Leistert

Hashing in digitalen Kulturen: wie Kryptographie den apparatus of capture absichert.

2024

https://doi.org/10.25969/mediarep/22944

Veröffentlichungsversion / published version
Sammelbandbeitrag / collection article

Empfohlene Zitierung / Suggested Citation:
Leistert, Oliver: Hashing in digitalen Kulturen: wie Kryptographie den apparatus of capture absichert.. In: Ralf Adelmann, Tobias Matzner (Hg.): *Filter.* Paderborn: Universität Paderborn 2024 (Medienwissenschaftliches Symposium der DFG 4), S. 1–9. DOI: https://doi.org/10.25969/mediarep/22944.

Nutzungsbedingungen:
Dieser Text wird unter einer Creative Commons - Namensnennung 4.0/deed.de Lizenz zur Verfügung gestellt. Nähere Auskünfte zu dieser Lizenz finden Sie hier:
https://creativecommons.org/licenses/by/4.0/deed.de

Terms of use:
This document is made available under a creative commons - Attribution 4.0/deed.de License. For more information see:
https://creativecommons.org/licenses/by/4.0/deed.de

DFG Deutsche Forschungsgemeinschaft

Philipps Universität Marburg

Oliver Leistert

Hashing in digitalen Kulturen: wie Kryptographie den *apparatus of capture* absichert.

Abstract über Hashing von Oliver Leistert. Hier wird kurz zusammengefasst, worum es im folgenden Artikel geht.

Schlagworte: Hashing, Digitale Kulturen, Kryptographie, Algorithmus

In den Medienwissenschaften gibt es keine Bestandsaufnahme der Anwendung kryptographischer Hashes in digitalen Kulturen. Dieses Forschungsdesiderat ist erstaunlich, weil sie es sind, die inzwischen ganz wesentlich zum stabilen Betrieb des Internets beitragen bzw. es in seiner heutigen Gestalt mitprägen. Was wenig bekannt ist: kryptographische Hashfunktionen sind Kontrollbausteine digitaler Kulturen in gewissermaßen schlankster möglicher Form. Ihre Filterfunktion ist mathematischer Natur, die Wirkung jedoch dient ganz wesentlich der Zugangsmodulation und der Sicherung von Eigentumsregimen. Auf Hashfunktionen und auf ihre kryptographische Belastbarkeit setzen heute diverse Aspekte der Ökonomie und der Sicherheit des Internets auf, da Daten schnell und sicher validiert und authentifiziert werden müssen. Die Erzeugung eindeutiger Identitätsnachweise oder Authentifizierungen, gesichert durch Kryptographie, mit denen sehr schnell gerechnet werden kann, ermöglicht erst die Prozessierbarkeit des endlosen Stroms der unzählbaren digitalen Objekte, die durchs Netz geschickt werden. Es handelt sich aber nicht nur um technische Gründe, die Hashes ins Zentrum des Internets haben rücken lassen, vielmehr werden sie zunehmend zu Mitteln der politischen Ökonomie, des Rechts und der Kontrolle.

Identifizierungs- und Verifikationstechniken sind *machtanalytisch* stets interessant, etablieren sie doch Asymmetrien des Zugangs, der Nutzung und der Kontrolle. Sie stehen inzwischen im Zentrum eines *apparatus of capture*, der mit dieser Filterfunktion den Transport der Daten algorithmisch kontrolliert.[1] Kryptographische Hashingfunktionen sind zunehmend die fundamentale, basale Armierung dieses Apparates geworden.[2]

Wofür (kryptographische) Hashfunktionen einst erfunden wurden, welche eigene Geschichtlichkeit sie ins Internet einbringen, und worin die Kontrolle besteht, für die sie heute ganz wesentlich in Gebrauch sind, wird in diesem Beitrag kursorisch beschrieben.

Was sind (kryptographische) Hashfunktionen?

Hashing heißt so viel wie zerhacken, zerstückeln oder auch verwürfeln: „chop and mix“. Hashfunktionen sind Einweg-Funktionen, die im informatischen Kontext aus einem beliebigen digitalen Objekt als Eingabe ein stets gleichlanges digitales Objekt („fixed-sized output“) als Ausgabe errechnen. Ferner soll dieselbe Eingabe stets dieselbe Ausgabe erzeugen.

Hashing formiert somit alle Objekte auf dieselbe Größe. Dabei ist das (utopische) Ziel, dass die Ausgabe der Rechenoperation (der Hash) von nur einem einzigen Eingabeobjekt erzeugt werden kann.

1 Vgl. Philip E. Agre: Surveillance and Capture: Two Models of Privacy, in: *The Information Society*, Bd. 10, Nr. 2, 1994, 101–127; sowie Gilles Deleuze, Félix Guattari: *Tausend Plateaus: Kapitalismus und Schizophrenie*, Berlin 1992 [1980], Kapitel 13 („7000 v. Chr.: Vereinahmungsapparat“), 588-656.

2 Dies wäre noch weiter auszuführen. Insbesondere inwiefern Giganten wie Google, AWS oder auch Cloudflare auf das Hashen angewiesen sind, und welche funktionalen Rollen das Hashen übernimmt.

Auch soll für jede Eingabe jede Ausgabe gleich verteilt sein, weshalb diese Funktionen auch Streuwertfunktionen genannt werden.

Die Eingabe wird gelegentlich auch Schlüssel genannt, da allein mit ihr die Ausgabe generiert werden kann (kryptographische Schlüssel sind hiervon jedoch zu unterscheiden). Manchmal wird auch von einer Nachricht gesprochen, dann heißt die Ausgabe *Message Digest*. *Kryptographische* Hashfunktionen, wie SHA256, müssen sich daran messen lassen, von der Ausgabe keinerlei Rückschlüsse auf die Eingabe zuzulassen. Die Mächtigkeit dieser Funktionen besteht in dieser Gerichtetheit, in ihrer Einweg-Funktionalität.[3] Ein „Zurückrechnen" von Ausgabe zu Eingabe soll nicht möglich sein. Und genau dieser Aspekt eröffnet die Möglichkeit der Etablierung asymmetrischer Zugänge und von Kontrollfiltern.

Das weite Feld von Anwendungen solcher Funktionen sorgt auch für unterschiedliche Begriffe: Prüfsumme oder Fingerprint wird die Ausgabe genannt, um zu überprüfen, ob etwa Software beim Download Schaden genommen hat, oder jemand Drittes die Datei verändert hat. Hiermit wird ausschließlich die Integrität der Daten überprüft. Bereits die Differenz eines *bits* würde dann zu einer gänzlich anderen Ausgabe der Hashfunktion führen. Diese Funktionalität heißt *Avalanche Effect* und sie verwischt bzw. dekorelliert Eingabe von Ausgabe.[4]

Hashfunktionen sind für die Kryptographie fundamental. Um nur zwei Anwendungen zu nennen: Messages Authentication Codes, mit denen nicht nur die Integrität, sondern auch die Authentizität von Daten überprüft werden,[5] und Zero-Knowledge-Proofs, mit denen zwei Parteien beweisen können, das gleiche Geheimnis zu teilen, ohne es zeigen zu müssen.[6]

Ein kleines Beispiel soll die Grundfunktion veranschaulichen: gegeben sei eine große Filmdatei. Wird die kryptographische Hashfunktion SHA256 auf diese Datei angewendet, lautet das Ergebnis:

6eec689081cf1d7e32edb2d691ca49ac8f0892ec7c36bd5892703a96294ffe05.

Gegeben sei eine kleine Textdatei. Wird dieselbe Hashfunktion SHA256 auf diese Datei angewendet, ist das Ergebnis gleich lang und lautet dennoch vollkommen anders:

4355a46b19d348dc2f57c046f8ef63d4538ebb936000f3c9ee954a27460dd865.

Deutlich wird, dass weder Größe noch irgendein anderer Aspekt, z.B. der Dateityp der Eingabe, auf nachvollziehbare Weise mit dem Output korreliert. Auch würde die Filmdatei bei nur minimaler Veränderung von einem *bit* erneut einen gänzlich anderen String als Ausgabe der SHA256-Funktion generieren.

Die standardisierte Ausgabe, im Falle dieser Funktion ein hexadezimaler String bestehend aus 64 Symbolen, macht deutlich, dass beliebige Eingabe-Objekte in eine Menge aus einmaligen, sehr einfach unterscheidbaren, wohlgeformten Elementen überführt werden, die sich sehr effektiv und sparsam übertragen und speichern lassen, auch weil deren Länge im Vorhinein bekannt ist.

---

3 Kryptographisch gesehen handelt es sich um eine Trinität: Collision Resistance, Preimage Resistance, Second Preimage Resistance. Zu den beiden letzteren, siehe *https://en.wikipedia.org/wiki/Preimage_attack* (18.09.2023).

4 Der Begriff und die Mathematik hierzu gehen auf Horst Feistel zurück, dem Erfinder des Lucifer-Algorithmus. Dieser wurde in abgewandelter und in durch die NSA veränderter Form 1977 zum behördlichen Chiffrierstandard DES (Data Encryption Standard) der USA.

5 Indem zusätzlich noch kryptographisch die Authentizität validiert wird.

6 Anwendungsfälle umfassen Situationen, in denen Wissen nachgewiesen werden muss, ohne es offenbaren zu wollen.

Kurze Geschichte des Hashens in der Informatik

Das Hashen in der Informatik entwickelt sich aus der Notwendigkeit von schnellem Speichern und Wiederfinden als grundlegende Operationen der Komputation. Die Suche nach effektiven Algorithmen zur Speicherung und zum Wiederfinden („Search & Retrieval") von Informationen ist bereits 1945 in Vannevar Bushs „As We May Think" als zu lösende Aufgabe beschrieben[7] und wird im Zuge der „Control Revolution", die James Beniger als Effekt einer überbordenden industriellen und bürokratischen Komplexität nach dem Ende des zweiten Weltkrieges rekonstruiert hat,[8] immer wichtiger.

In der Informatik werden Hashfunktionen deshalb im Kontext des Problems der *Effektivität* von Speicherung und dem Wiederfinden von Daten eingeführt. Die Frage, die zu lösen ist, klingt einfach: Wie finde ich möglichst schnell etwas wieder, das ich irgendwo gespeichert habe? Die bis dato zum Speichern verwandte Datenstruktur, die Liste, skaliert nicht und entwickelt sich trotz binärer Suche zum Bottleneck der Datenverarbeitung.[9] Lineare Suche dauert nun mal lineare Zeit.

Ein Hashtable war die Lösung, sprich eine Tabelle oder Array,[10] das als Adresse die Ausgabe von Hashfunktionen aufnimmt.

Das Problem der Kollision steht damit bereits früh im Zentrum der Forschung. Eine Kollision findet statt, wenn mehr als ein Schlüssel auf eine Adresse zeigt. Um Kollisionen zu verhindern, also die Zuweisung zweier oder mehrerer Schlüssel auf eine Adresse, werden zwei Wege eingeschlagen.

Ein einfacher Trick ermöglicht die erste Methode, die der verketteten Listen. Die Adresse bildet eine Liste, an die Objekte angehängt werden können. Trifft ein Schlüssel auf eine Adresse, die schon belegt ist, wird einfach an Platz zwei der Liste angehängt. Dieser Hashtable ist in sich dynamisch erweiterbar, da er durch Verlistung eine weitere strukturelle Dimension aufschließt.

Diese Kollisionsverhinderung durch innere, verkettete Listen wird zuerst in einem internen IBM Memo im Jahre 1953 von Hans Peter Luhn beschrieben.[11] Insofern lässt sich der IBM-Angestellte als eine Gründungsfigur des Hashens in der Computerwelt sehen, auch wenn der Begriff selbst erst rund zehn Jahre später im Umlauf ist.[12]

Das zweite Verfahren arbeitet durch offene Adressierung. Dies ist ein Verfahren aus dem Jahr 1954, das u.a. Elaine M. McGraw und Gene Amdahl bei IBM entwickelten.[13] Dieses Verfahren eignet sich, wenn im Vorhinein der erforderliche Speicherbedarf bekannt ist und eine größerer zusammenhängender Adressraum geschützt und reserviert werden kann.

---

7 Vannevar Bush: As We May Think, in: *The Atlantic*, 1945, 101–108; sowie erläuternd: Hartmut Winkler: Vannevar Bush: As We May Think, in: *FormDiskurs* 1, Nr. 2, 1997, 136–47.

8 James R. Beniger: *The Control Revolution: Technological and Economic Origins of the Information Society,* Cambridge 1986.

9 Die binäre Suche fängt in der Mitte an und bewegt sich je nach Ziel als nächstes in die Mitte der linken Hälfte oder der rechten Hälfte und immer weiter, bis die Adresse gefunden ist.

10 Arrays sind Datenstrukturen. Tabelle und Array werden mitunter durcheinander benutzt.

11 Alan G. Konheim: *Hashing in Computer Science: Fifty Years of Slicing and Dicing*, Hoboken 2010, 125–6.

12 Luhn gab auch dem Luhn-Algorithmus seinen Namen. Dies ist eine einfache Prüfsummenfunktion auf modulo-10 Basis, die heute noch breite Verwendung findet, z.B. zur Berechnung von Kreditkartennummern. Ferner erfand er eine Vielzahl von Indexierungsfunktionen zum Wiederfinden von Informationen, von denen insbesondere der ‚Key Word in Context' oder kurz KWIC-Algorithmus breite Anwendung fand und als ein Vorläufer von Suchmaschinen gelten kann. Siehe Hallam Stevens: The IBM Engineer Gave Computers a Way to Quickly Search Documents, DNA, and Databases, in: *IEEE Spectrum*, Bd. 55, Nr. 2, 2018, 44–49. Luhn hat ausserdem ganz wesentlich die statistische Auswertung von Texten zur Herstellung von Indizes vorangetrieben. Siehe Hans Peter Luhn: A Statistical Approach to Mechanized Encoding and Searching of Literary Information, in: *IBM Journal of Research and Development* 1, Nr. 4, 1957, 309–17. Siehe hierzu: Theo Röhle: *Der Google-Komplex. Über Macht im Zeitalter des Internets*. Bielefeld 2010, 114ff.

13 Alan G. Konheim: *Hashing in Computer Science: Fifty Years of Slicing and Dicing*, Hoboken 2010, 125.

Im von G.D. Knot vielleicht ersten vorgelegten historischen Rückblick auf das informatische Hashen aus dem Jahr 1975 wird hervorgehoben, dass Hashing noch aus der Ära der Lochkarten stammt. Schon damals ging es darum, mittels des errechneten Hashes die richtigen Speicherorte zu finden. Insbesondere einen Aspekt des Hashens benennt Knott, dessen Bedeutung heute kaum überschätzt werden kann: „[W]e may consider hashing as a *compression operation* where we convert key-values into shorter values, and then file the items involved according to the shortened values, somehow resolving the problem of collision".[14] Die Bedeutung von Datenkomprimierung hat im Laufe der Ausweitung der berechneten Welt stets zugenommen. Gleichzeitig erzeugt die Anforderung einer Komprimierung eine utopische Dimension für die Anwendung von Hashfunktionen, denn die Kompression produziert ein mathematisch unlösbares Problem: es kann keine kollisionsfreie Abbildung einer größeren auf eine kleinere Menge geben. Luhns internes IBM Memo ist laut Knott der erste bekannte Vorschlag, wie das Problem der Kollisionen elegant gelöst wird: „In early 1953, Luhn had written an internal IBM memo in which he suggested putting information into "buckets" in order to speed up a search."[15] Die eigentlich zeitlich aufwendige und zu überwindende sequentielle Suche in den einzelnen kleinen Eimern fällt immer noch viel schneller aus, als die bisher übliche Suche in einer großen Liste.

Das Adressierungsproblem bestand grundsätzlich fort, wandelte sich jedoch durch die Entwicklung kryptographischer Streuwertfunktionen und generell durch die Weiterentwicklung des gesamten IT-Sektors.

Kryptographisches Hashen

Während das Problem der effektiven Speicherung im Laufe der Jahrzehnte eine eigene Dynamik entwickelt,[16] wächst die Bedeutung kryptographischer Hashfunktionen für die IT ebenfalls, da sie jenseits der Adressierungsoptimierung noch etwas anderes leisten: sie können einen mathematisch und juristisch belastbaren Nachweis über ein Objekt liefern, ohne dabei das nachgewiesene Objekt offenlegen zu müssen.

Ein intuitiv gut verständliches Problem ist das der Zugangspasswörter zu Computern. Diese werden heute nur noch als kryptographische Hashes in den Rechnern abgelegt. Bei der Passworteingabe wird das eingegebene Passwort durch eine Hashfunktion gereicht und der Output, der Hash, wird mit dem Eintrag in der Password-Hashes-Datei verglichen. In der Passwort-Hashes-Datei steht nur der Hash des Passworts. Erlangt jemand unbefugt Zugriff auf die Password-Hashes-Datei, kann kein Passwort ausgelesen werden und auch die illegitime Kenntnis der Hashes führt zu nichts, denn diese Hashes können die Kenntnis eines Passwortes nur bestätigen, aber mit ihnen lässt es sich nicht rekonstruieren. Dies ist das Wunderbare der Einwegfunktionen: mit ihnen lässt sich nur in eine Richtung rechnen. Für den Fall der Passworteingabe ist der Gewinn an Sicherheit durch das kryptographische Hashen offensichtlich, da die sensiblen Passwortdaten gar nicht mehr im Rechner abgelegt werden müssen. „Einwegfunktionen […] schützen Algorithmen vor ihrem eigenen Ergebnis",[17] schrieb Friedrich Kittler, bereits ahnend, dass diese ‚Schutzfunktion' kaum überbewertet werden kann. Heutige, zusehends auch von Algorithmen kontrollierte Gesellschaften lassen sich permanent und meist ohne davon zu wissen von Hashes schützen. Doch was in manchen Kontexten Schutz und Sicherheit sind, kann in anderen Kontexten Ausschluss und Risiko bedeuten. Die Einwegfunktionalität steht in jedem Fall im Zentrum einer

14 Gary D. Knott: Hashing Functions, in: *The Computer Journal*, Bd. 18, Nr. 3, 1975, 265–278, hier 273.
15 Ebd.
16 Vgl. Marcus Burkhardt: *Digitale Datenbanken: eine Medientheorie im Zeitalter von Big Data*, Bielefeld 2015, 149-218.
17 Friedrich Kittler: *Draculas Vermächtnis. Technische Schriften*, Leipzig 1993, 235.

Operation, die einen Unterschied macht, indem sie eine Sperre errichtet, die nur aus einer Richtung passierbar ist.

Dass das kryptographische Hashen aber nicht nur Geschichte macht, sondern selbst Geschichte hat, soll im folgenden Abschnitt dargestellt werden.

Ein mehrfach gebrochener Filter: das langsame Ausschleichen von MD5

Ein Blick in die Geschichte der kryptographischen Hashfunktionen zeigt, dass diese Elemente des Filterns und der Kontrolle selbst ihre Geschichte haben und darin der Gefahr ausgesetzt sind, ihre Filter- und Kontrollfunktion mehr oder weniger mit einem Schlag zu verlieren, indem doch wieder in beide Richtungen gerechnet werden kann.

Die Gründe, warum dies möglich wird, obwohl sie „Einwegfunktionen" heißen, sind vielfältig. Neben der rasant wachsenden Rechenleistung, mit der bereits durch einfache Brute-Force-Attacken das Knacken der Funktion, also das Aufheben der Mappings in nur eine Richtung, in immer kürzerer Zeit möglich wird, tauchen (neben den vielfältigen fehlerhaften Implementationen) auch immer wieder Schwächen in den Algorithmen selbst auf, die den Gang vom Hashwert zum Objekt nicht mehr ausreichend rechenintensiv gestalten.

Exemplarisch wird hier die Geschichte des MD5 Message-Digest-Algorithmus referiert. Dieser Algorithmus wird 1991 von Ronald Rivest[18] entwickelt und 1992 im RFC[19] 1321 beschrieben.[20]

2011, rund zwanzig Jahre später, erscheint das RFC 6151,[21] das MD5 für unsicher erklärte: „The published attacks against MD5 show that it is not prudent to use MD5 when collision resistance is required",[22] d.h. für alle sicherheitsrelevanten Vorgänge.[23] Bereits 1993 werden die ersten erfolgreichen Kollisionen, d.h. das Brechen der eindeutigen Abbildung, gemeldet. 2005 wird auf der EuroCrypt-Konferenz schließlich ein Paper vorgestellt, dass das Verfahren vollständig beschreibt.[24] 2006 können Kollisionen bereits mit einem Standard-Notebook innerhalb einer Minute errechnet werden.[25] Kaum ein Jahr später können falsche X.509 Zertifikate, die mit MD5 erstellt wurden, Webbrowsern erfolgreich als korrekte Zertifikate untergeschoben werden.[26]

---

18 Das ‚R' in Rivest-Shamir-Adleman, kurz RSA. Dies ist eine Sicherheitsfirma, die 1977 ganz wesentlich zum asymmetrischen Public-Private-Key Verfahren beigetragen hat.

19 RFCs (Request for Comments) sind das technische Internet als Textsammlung, gehostet von der Internet Engineering Task Force (IETF). Hier finden sich praktisch sämtliche Standards, Protokolle und andere technische Elemente des Internets natürlichsprachlich beschrieben, beginnend mit dem RFC0001 („Host Computer") von 1969.

20 Ronald Rivest, The MD5 Message-Digest Algorithm, *RFC 1321*, April 1992, https://doi.org/10.17487/RFC1321.

21 Sean Turner und Lily Chen, Updated Security Considerations for the MD5 Message-Digest and the HMAC-MD5 Algorithms, *RFC 6151*, März 2011, https://doi.org/10.17487/RFC6151.

22 Ebd. Unter ‚2.1 Collision Resistance' ist die Geschichte der Angriffe auf diesen Algorithmus knapp referiert. 2008 bereits erklärte das CERT Coordination Center MD5 für unsicher, nachdem erfolgreich und schnell falsche SSL-Zertifikate erzeugt wurden. Siehe www.kb.cert.org/vuls/id/836068/ (12.2.2022).

23 Ein illustrer Effekt der gefundenen Schwächen des MD5 ist, dass 2010 das US-Cyber-Command Logo dechiffriert werden konnte, das den MD5-Hash des Mission-Statement der Einheit im Emblem trägt: 9ec4c12949a4f31474f299058ce2b22a. Siehe www.wired.com/2010/07/solve-the-mystery-code-in-cyber-commands-logo/ (12.2.2022).

24 Xiaoyun Wang und Hongbo Yu: How to Break MD5 and other Hash Functions, in: Ronald Cramer (Hg.): *Advances in Cryptology – EUROCRYPT 2005. Lecture Notes in Computer Science*. Berlin, Heidelberg 2005, https://doi.org/10.1007/11426639_2.

25 Vlastimil Klima: Tunnels in Hash Functions: MD5 Collisions within a Minute, in: *Cryptology ePrint Archive*, Report 2006/105, 2006, https://eprint.iacr.org/2006/105 (12.02.2022).

26 Marc Stevens, Arjen Lenstra, Benne de Weger: Target Collisions for MD5 and Colliding X.509 Certificates for Different Identities, in: *Cryptology ePrint Archive*, Paper 2006/360, (2006), https://eprint.iacr.org/2006/360 (12.02.2022).

X.509-Zertifikate sind enorm kritische Elemente der Internetsicherheit. Jede mit *https* gesicherte Verbindung setzt hierauf auf. Darüber hinaus wird im bereits genannten RFC 6151 festgestellt: „In fact, more delicate attacks on MD5 to improve the speed of finding collisions have been published recently. However, the aforementioned results have provided sufficient reason to eliminate MD5 usage in applications where collision resistance is required such as digital signatures.“ Im Klartext heißt dies, dass MD5 nicht mehr verwendet werden soll. Doch wie eine unsicher gewordene kryptographische Funktion wieder einfangen, die praktisch überall im Internet Verwendung gefunden hat?

Dieses Problem verweist auf die schwierigen und vielfältigen Zeitlichkeiten, denen die Sicherheit des Internets unterliegt. Zwischen 1992, der Erfindung von MD5, und 2011, dem Erscheinen der Sicherheits-Warnung der IETF zu MD5, ist das Internet für die Massen entstanden und mit ihm ein ganzes Bündel an Sicherheitsanforderungen. 1992 und 2011 sind vollkommen unterschiedliche Epochen des Internets. Message Digest 5 ist insbesondere aufgrund seiner hohen Geschwindigkeit und eines verhältnismäßig geringen Rechenaufwands sehr beliebt bei der Validierung von Daten. Zertifikate werden standardmäßig von ihren Ausgabestellen mittels MD5 authentifiziert und signiert.

Das Versagen von MD5 ist ein wirklich dramatischer Moment in der Geschichte des kryptographischen Hashens, da weite Teile der Verifizierung geschützter Verbindungen darauf aufsetzt. Zertifizierungsinstanzen können nicht mehr länger authentifiziert werden, da die Methode der Authentifizierung versagt. Die top-down organisierte Zertifikatsstruktur des Internets fällt in weiten Teilen in sich zusammen, da die oberste Instanz gebrochen ist. Die gesamte darauf aufbauende Kette mathematischer Beweise bricht weg.[27]

Die Geschichten von erfolgreichem Datendiebstahl durch Attacken auf das MD5-Verfahren sind zahlreich und vermutlich sind viele unbekannt geblieben. Als erstes ist an Man-in-the-Middle-Attacken zu denken, denn mittels eines gefälschten X.509-Zertifikats können Angreifer den eigentlich verschlüsselten Internet-Traffic mitlesen und manipulieren, ohne dass dies auffallen muss.

Ein prägnantes Beispiel liefert die Malware *Flame*, die insbesondere Rechner im Iran infiziert. Sie nutzt aus, dass eine Microsoft-Zertifizierungsinstanz zur Authentifizierung von Software auch im Jahre 2012 noch MD5-Hashes benutzt. Die Autor*innen von *Flame*, wahrscheinlich Angestellte von NSA, CIA und dem israelischen Militär,[28] sind in der Lage, ein Zertifikat zu erstellen, das Windows-Rechnern die extrem komplexe Malware *Flame* als eine von Microsoft geprüfte und akzeptierte Software darstellt.

Das Beispiel zeigt, dass eine top-down organisierte Zertifikatsstruktur besonders anfällig ist, wenn das Wurzelzertifikat kompromittiert wurde. Deutlich wird auch, wie folgenreich das Versagen von MD5 *und* dessen fortdauernde Nutzung ist. Wenn MD5 ein Internet-Filter ist, der das *Echte* vom *Falschen* trennen soll, ist, seit er gebrochen wurde, das Echte vom Falschen rechnerisch nicht mehr zu unterscheiden. Doch diese Unterscheidbarkeit ist systemische Bedingung des allgemeinen Internetbetriebs geworden.

---

27 Unter *www.phreedom.org/research/rogue-ca/* ist der Angriff auf das Wurzel-Zertifikat beschrieben: „we executed a practical attack scenario and successfully created a rogue Certification Authority (CA) certificate trusted by all common web browsers. This certificate allows us to impersonate any website on the Internet, including banking and e-commerce sites secured using the HTTPS protocol.“ 2013 wurde ein Angriff auf MD5 beschrieben, für den ein durchschnittlicher Rechner dieser Zeit nur noch wenige Sekunden brauchte: Tao Xie, Fanbao Liu und Dengguo Feng: Fast Collision Attack on MD5, *Cryptology ePrint Archive*, Paper 2013/170, 2013, *https://eprint.iacr.org/2013/170* (12.02.2022).

28 Siehe *https://en.wikipedia.org/wiki/Flame_(malware)* (12.12.2022).

Zu kryptographischen Zeitlichkeiten digitaler Kulturen

Solch ein zeitlicher *Lag* zwischen der Publikation von Schwächen in kryptographischen Verfahren und dem nur allmählichen Ausschleichen der unsicheren Funktionen aus dem Rechnerverbund Internet[29] deutet auch auf eine historische Problematik hin, die grundsätzlich zwischen akademischer Kryptologie und der kryptographischen IT-Praxis besteht. Das Katz-und-Maus-Spiel beider Seiten lässt sich insofern als Motor der Weiterentwicklung solcher Verfahren deuten. Historisch betrachtet, ließe sich der Erfolg des Internets (etwas überspitzt) auch als Effekt solcher kryptographischen Entwicklungen lesen, die jeweils bestimmte Anwendungen und deren Vermassungen überhaupt erst ermöglichen. Stärke und zugleich Schwäche der MD5-Funktion ist es gewesen, besonders schnell Hashes berechnen zu können.[30] Dies spielt durchaus eine Rolle für die Skalierbarkeit kryptographischer Verfahren. Die historische Entwicklung zeichnet sich aus durch die Stärke des Schutzes, den die implementierten Algorithmen errechnen, der Geschwindigkeit, in der sie verfahren, den Schutz zu errechnen und dem Entwicklungsstand der Hardware.

Ein kurzer Zwischenstand: Hashfunktionen werden u.a. dafür gebraucht, das Echte vom Falschen, das Korrekte vom Unkorrekten zu unterscheiden bzw. zu etablieren, was richtig und was falsch ist. Hierfür filtern sie alles Falsche heraus bis auf das *eine Richtige*.

Hashfunktionen können somit nicht nur eine epistemische, sondern auch ontologische Qualitäten zugeschrieben werden, wenn sie digitale Objekte für gültig oder ungültig erklären und damit zur weiteren Verrechnung weitergeben oder verwerfen, denn sie bestimmen, was der Fall ist. Gleichzeitig stehen die Methoden dieses Unterscheidungsvermögens in einem mathematischen Kontext, der selbst zutiefst historisch ist, denn stets wachsen die Fähigkeiten derer, die die Funktion brechen wollen, angetrieben insbesondere durch die Weiterentwicklung der Hardware und der Erhöhung von CPU- bzw. GPU-Zyklen.

Ist der Filter, die Hashfunktion, selbst historisch, erweist sich die Unterscheidung von falsch und richtig, oder wahr und falsch, die er betreibt, auch als historisch: Was falsch ist, kann wahr werden und andersherum? Im Internet auf jeden Fall! Zumindest, solange nur Maschinen für diese Werte zuständig sind. Hashes, wie sie z.B. MD5 erzeugt, haben einen zeitlich begrenzten Wahrheitswert. In der IT-Sicherheit wird dies *Algorithm Security Lifetime* genannt: die Dauer, die ein kryptographischer Algorithmus Daten absichert, bevor er unsicher wird. Diese *Algorithm Security Lifetime* könnte in Anbetracht der steigenden gesellschaftlichen Abhängigkeit von sicheren Hashfunktionen als eine historische Zeiteinheit digitaler Kulturen gelten. Durch die Geschichte der IT und des Internets wandert insofern ein *Filterideal*, das sich in Schüben materiell stets neu realisieren muss (z.B. von MD5 zu SHA256). Nur aufgrund seiner Erneuerungen kann dieser Filter die Unterscheidung von wahr und falsch durch die Zeit hindurch halten.

Kontrolle durch Hashing: cui bono?

Das Errechnen von Hashes und die damit verbundene Kontrollierbarkeit digitaler Objekte ist aus heutigen digitalen Kulturen nicht mehr wegzudenken. Es ließe sich sogar die These aufstellen, dass Hashes erst zur Bewältigung heutiger Datenmengen führen. Zur Erinnerung: Einen Hash als Surrogat für das

29 Zur lokalen Prüfung von digitalen Objekten wird MD5 weiterhin gern benutzt.

30 Siehe etwas ausführlicher zum kryptographischen Hashing aus angewandter Perspektive: Bruce Schneier: *Applied cryptography*, Indianapolis 2015, 429–460, sowie Niels Ferguson und Bruce Schneier: *Practical cryptography,* New York 2003, 83–96.

eigentliche digitale Objekt zu verwenden, reduziert den Rechenaufwand drastisch und kann insbesondere für Anwendungen, bei denen es um den Nachweis der korrekten Existenz des digitalen Objekts geht, eingesetzt werden. Der Nachweis der korrekten Existenz spielt eine wichtige Rolle für die Sicherheit aller, die das Internet benutzen. Dies kommt aber auch dem Zwang zur Verwertung im Überwachungskapitalismus sehr entgegen. Der korrekte Nachweis eines digitalen Objekts mutiert hier zum Nachweis der Verwertbarkeit eines zu verwertenden Objekts, sei es direkt monetär, oder indirekt über Tracking und Überwachung. Ein Objekt kann hier eine Browseranfrage, ein Cookie, eine Musikstream oder eben ein ganzer Film sein. Entscheidend ist, dass es in einem System von Überwachung und Kontrolle eingebunden und nur dort legal ist. Durch das Hashing ist der Nachweise der legalen Verwendung des Objekts in vielerlei Kontexten mit wenig Rechenaufwand möglich.

Symptomatisch für den systematischen Zusammenhang von Hashes und Eigentumsregimen erscheint der Umstand, dass Hashes integraler und vielleicht wichtigster Bestandteil von sog. Kryptowährungen und NFTs bzw. Blockchains sind. In diesen Feldern wird mit Marketing-Wörtern wie *web3.0* versucht, soziale Interaktionen im Netz auf monetäre Transaktionen zu reduzieren bzw. beides miteinander zu amalgamisieren.[31] Kryptographische Hashes dienen hier zuallererst als Eigentumsnachweis digitaler Unikate.

Doch auch jenseits der rechtslibertären Cryptocoins werden Hashes zur Produktion digitaler Unikate angewendet und überführen damit digitale Objekte in ein Regime von Kontrollfunktionen, das Zugang, Berechtigungen, Benutzung und Überwachung über digitale Objekte legen kann. Kryptographische Funktionen mutieren damit zu Sicherungsfunktionen von Eigentumsregimen in digitalen Kulturen. Sie ermöglichen ohne externe Tools die Errichtung von Sperren und Asymmetrien, die lange Zeit für das bürgerliche Eigentumsregime im Digitalen nur in Teilen und nur mit höheren Kosten operationalisierbar waren. Aus dieser Perspektive betrachtet, werden Hashfunktionen jenseits ihrer mathematischen Filterfunktion zu politischen Filtern des Zugangs und der Verwertung.

Diese Filter gestalten letztlich die digitalen Milieus, die die Einzelnen je nach Positionalität ihres dividuellen Phantoms errechnet werden. In solch dividuell produzierten digitalen Milieus werden scheinbar individualisierte Angebote sowie Belohnungen u.a. zur längeren Verweildauer auf Plattformen bereitgestellt. Diese „environmentalitäre Situation“[32] könnte ohne Hashes nicht auskommen.

Ausblick für ein Forschungsprogramm

Zur Kritik der politischen Ökonomie und der sozialen wie gesellschaftlichen Folgen von Plattformen des Überwachungskapitalismus oder allgemeiner dem Dispositiv Internet gibt es zahlreiche wichtige grundlegende Publikationen, die hier nicht extra referiert werden sollen. Die Ergebnisse der kritischen Arbeiten sind oft ähnlich: Diese Technologien tragen zur Verschärfung gesamtgesellschaftlicher Probleme bei, sei es u.a. durch vermehrte Diskriminierung oder durch die Ausweitung des Marktdiktats in bisher davon verschont gebliebenen sozialen Sphären. Generell lässt sich sagen, dass durch das Internetdispositiv bestehende soziale und ökonomische Asymmetrien eher verhärtet als aufgelöst werden. Doch wenig ist in der Medienwissenschaft zu hören, wie dies operationalisiert wird.

---

31 Zur Übersicht, siehe Quinn DuPont. *Cryptocurrencies and blockchains*. Medford/MA 2019, sowie Finn Brunton: *Digital cash: the unknown history of the anarchists, utopians, and technologists who created cryptocurrency*, Princeton/NJ 2019.

32 Erich Hörl: Die environmentalitäre Situation, in: *Internationales Jahrbuch für Medienphilosophie,* Bd. 4, Nr. 1, 2018, 221–50, https://doi.org/10.1515/jbmp-2018-0012.

Meines Erachtens sollte eine Kritik all dieser Phänomene auch eine Erschließung und Beschreibung der funktionalen Elemente dieser Entwicklungen umfassen. Durch das Freilegen solcher *building blocks* überwachungskapitalistischer Grausamkeiten, durch deren Sortierung und Darstellung, wäre viel für die kritische medienwissenschaftliche Arbeit, aber auch darüber hinaus für das Verständnis des Technikverbundes Internet gewonnen.

Im Falle der Hashfunktionen, die ich zu solchen *building blocks* zähle, ist zunächst die Einsicht zu nennen, dass Funktionen der Internetsicherheit mehr und mehr vom *apparatus of capture* geschluckt und umgewidmet werden, um Kontrollregime zu errichten, die Asymmetrien stabilisieren oder verstärken. Es wäre folglich sinnvoll, einen genaueren Einblick in den Umfang der Wirkungen und Effekte von kryptographischen Hashfunktionen zu erlangen. Vielleicht stellt sich dabei heraus, dass Kryptographie der eigentliche Garant des Überwachungskapitalismus und seiner extraktivistischen Praxen geworden ist.

Mit der Analyse dieser und weiterer *building blocks* digitaler Kulturen entstünde ein Katalog dessen, was den Überwachungskapitalismus *in seinem Innersten zusammenhält* und entsprechend auch eine Perspektive darauf, wie er zu überwinden wäre. In dieser Hinsicht wäre z.B. zu fragen, welchen Beitrag Kryptographie zu post-kapitalistischen Projekten wie dem *commoning* leisten kann. In jedem Fall ist es dringend notwendig, kryptographische Funktionen wie das Hashing umfänglicher in soziale, ökonomische, aber auch historische Diskussionen digitaler Kulturen einzubeziehen.