The Legal Status of Binary Code: Debunking the Myth of Illegality

Executive Summary

This report addresses the persistent misconception that binary code is illegal. Binary code, the fundamental system of 1s and 0s underlying all digital computing, is merely a method of representing information and is not, in itself, illegal under any recognized legal framework. The misconception arises from the conflation of this neutral representational system with the *content* it represents, which can indeed be illegal (e.g., malware, pirated software, classified data). This report will define binary code, clarify its legal status, analyze its association with illicit activities, examine pivotal legal cases involving specific binary sequences (such as "illegal numbers"), and distinguish the code itself from the data it encodes, thereby providing a comprehensive legal and technical clarification.

related posts : Best Binary OptionS Brokers (in 2025)

I. Understanding Binary Code: The Foundation of Digital Information

Binary code serves as the bedrock upon which the entire digital world is built. Understanding its nature and function is essential before addressing its legal status. It is not a complex, esoteric entity reserved for programmers, but rather a fundamental system of representation with historical roots and ubiquitous modern applications.

A. Defining Binary Code: The Base-2 System

At its core, binary code is a system used to represent text, computer processor instructions, or any other form of data using only two symbols.[1] These symbols are universally recognized as '0' and '1', derived from the binary (base-2) number system.[1] This system stands in contrast to the decimal (base-10) system commonly used in everyday life, which employs ten digits (0 through 9).[4] In the binary system, each '0' or '1' is termed a "bit," an abbreviation for "binary digit," representing the smallest possible unit of data in computing.[5] Because a single bit conveys very limited information, bits are typically grouped together. A common grouping is a sequence of eight bits, known as a "byte," which can represent 28, or 256, distinct values.[4] These bytes, and larger groupings, allow for the representation of numbers, characters, and complex data structures.[3]

Like the decimal system, binary code relies on positional notation. The value of each digit depends on its position within the sequence. However, instead of representing powers of 10 (units, tens, hundreds, etc.), each position in a binary number represents a successive power of 2 (20=1, 21=2, 22=4, 23=8, and so on).[5] For example, the binary number 101101 translates to the decimal number 45 as follows: (1×25)+(0×24)+(1×23)+(1×22)+(0×21)+(1×20)=32+0+8+4+0+1=45.[7] This system allows any number, and by extension any information that can be numerically encoded, to be represented using just the two binary digits.[5]

The power and prevalence of binary code in technology stem from its direct correspondence to physical states in electronic and other systems. The '0' and '1' can represent two mutually exclusive states: 'off' or 'on' for a switch [8], low or high voltage in a circuit [5], the absence or presence of an electrical pulse [2], the north or south polarization of a magnetic spot on a disk, or even the flat spaces versus raised dots in the Braille writing system.[5] This inherent simplicity makes binary the most practical and efficient language for designing and operating digital electronic devices.[4]

B. The Universal Language of Computers: Representing Data and Instructions

Binary code is not merely one way computers handle data; it is the most fundamental layer of all computing systems.[6] Every piece of digital information, regardless of its complexity or perceived form, is ultimately stored, processed, and transmitted as sequences of binary digits.[1] This includes numerical data, textual information (encoded using standards like ASCII or UTF-8, where each character corresponds to a specific binary pattern) [4], visual images (where pixels' color and brightness are encoded) [4], audio recordings (digitized sound waves) [4], video streams [6], and, crucially, the very instructions that tell a computer what to do.[1]

The Central Processing Unit (CPU), the "brain" of a computer, operates directly on binary data. It performs arithmetic calculations (like addition, where 1+1=10 in binary [4]) and logical operations (like AND, OR, NOT) using binary representations of numbers and instructions.[2] These instructions, collectively known as machine code, are themselves sequences of binary digits that the CPU is specifically designed to interpret and execute.[7] While humans typically write software using higher-level programming languages (such as C++, Java, Python, or Rust) that use English-like commands and structures [8], these languages must ultimately be translated into binary machine code through processes called compilation or interpretation before the computer hardware can understand and act upon them.[10]

The applications of binary code permeate nearly every aspect of digital technology. Calculators convert decimal inputs into binary for computation and then convert the binary result back to decimal for display.[6] File compression algorithms use binary representations to reduce data size for efficient storage and transmission.[6] Cryptographic systems rely heavily on binary operations for encryption and decryption to secure data.[6] Digital clocks use binary counters to track time.[6] Multimedia processing involves decoding streams of binary data that make up audio and video files for playback.[6] Furthermore, all digital communication, from internet traffic to mobile phone calls, involves encoding information into binary signals for transmission and decoding it at the receiving end.[6]

C. A Brief Historical Perspective

The journey of binary code from abstract concept to the cornerstone of modern technology spans centuries. While systems employing binary principles appeared in various ancient cultures—such as the 64 hexagrams of the Chinese *I Ching*, based on combinations of broken (yin) and solid (yang) lines, dating back possibly to the 9th century BC [3], or the work of the Indian scholar Pingala around the 5th-2nd centuries BC describing prosody using binary patterns [3]—the formalization of the modern binary number system is credited to the German polymath Gottfried Wilhelm Leibniz.[3] In works like his 1689 manuscript "De Progressione Dyadica" and his 1703 publication "Explication de l'Arithmétique Binaire," Leibniz systematically explored the base-2 system using only 0 and 1, detailing methods for conversion and arithmetic operations.[3] Leibniz was fascinated by the *I Ching*, seeing its hexagrams as corresponding to binary numbers and affirming his own philosophical and theological ideas about creation from nothing (*creatio ex nihilo*), viewing binary as a fundamental, simple representation of existence.[3] He also envisioned binary as a potential pathway towards a universal logical language.[3]

Leibniz's work on binary, though groundbreaking, did not find immediate practical application. The next major conceptual leap occurred in the mid-19th century with the English mathematician George Boole.[16] In his 1847 work "The Mathematical Analysis of Logic," Boole developed an algebraic system for representing logical propositions.[3] This system, now known as Boolean algebra, operates on binary principles—variables can be either true or false (represented as 1 or 0)—and uses logical operators like AND, OR, and NOT.[3] Boole's work provided a formal mathematical framework for logical reasoning based on two states.[19]

The crucial link between Boole's abstract logic and practical computation was forged by Claude Shannon, an American mathematician and electrical engineer, often called the "father of information theory".[16] In his highly influential 1937 master's thesis at MIT, titled "A Symbolic Analysis of Relay and Switching Circuits," Shannon demonstrated that the principles of Boolean algebra could be directly implemented using electrical circuits with switches (which, like binary digits, have two states: on or off).[3] Shannon showed how networks of relays could be designed to perform complex logical and arithmetic operations, effectively proving that machines could process information encoded in binary.[3] This insight laid the essential theoretical groundwork for the design of all subsequent digital computers.[15]

This historical trajectory—from Leibniz's mathematical formalization, through Boole's logical system, to Shannon's realization of its electronic implementability—is key. It reveals that binary code wasn't arbitrarily chosen for computers. Its fundamental simplicity, requiring only two states, perfectly matched the physical capabilities of early electronic components like relays and later transistors, which could reliably represent 'on' and 'off' states.[9] This inherent suitability, rooted in mathematics and logic, explains why binary became, and remains, the foundational language of digital computation.

Furthermore, the fact that *all* forms of digital information—text, images, sound, video, complex software logic—are ultimately represented by the same underlying system of 0s and 1s highlights binary's role as a universal *medium*.[1] It is a neutral container for information, agnostic to the nature or legality of the content it holds. This universality is crucial for understanding why the binary system itself cannot be inherently illegal; its legal status is entirely dependent on the specific message being conveyed through it at any given time.

II. The Legal Status of Binary Code: System vs. Substance

Despite the misconception suggested by the initial query, binary code itself enjoys no special, prohibited legal status. Laws regulate content and actions, not the fundamental systems of representation used to encode them. Clarifying this distinction is paramount.

A. Binary Code as a Neutral System: Not Inherently Illegal

A thorough search of legal statutes and regulations reveals no jurisdiction where the binary number system or the concept of binary code is outlawed.[20] The binary system is a foundational concept in mathematics and computer science, analogous in its neutrality to the decimal system, the letters of the alphabet, or musical notation. It is simply a method for encoding information.

Legal frameworks worldwide focus on the *content* represented by binary code or the *actions* undertaken using software expressed in binary, not the representational system itself.[20] For instance, laws prohibit the distribution of malware or the infringement of copyright, regardless of whether the offending item is stored or transmitted using binary, decimal, or any other encoding scheme. The illegality resides in the substance or the act, not the notation.

B. The Critical Distinction: Representation vs. Represented Content

The core of the matter lies in separating the *medium* from the *message*. Binary code is the medium—a sequence of 0s and 1s. The legality of that sequence depends entirely on what message or data it represents.[20]

Consider these examples:

- The binary code constituting a legally licensed operating system like Windows or macOS is perfectly legal to possess and use according to the license terms.
- The binary code constituting an open-source program distributed under the GNU General Public License (GPL) is legal to use, modify, and distribute under the terms of the GPL.[25]
- However, the binary code constituting a malicious virus designed to damage computer systems is illegal because of the harmful *function* it performs.[26] Its representation in binary is incidental to its illegality.
- Similarly, the binary code representing a movie file that has been copied and distributed without the copyright holder's permission is illegal because it embodies an act of copyright infringement.[28] The binary format is merely how the infringing copy is stored.
- The binary sequence representing classified government documents would be illegal to possess or transmit for unauthorized individuals due to the *sensitive nature* of the information encoded.[20]

An analogy helps clarify this: the English alphabet is a neutral system for representing language. It is not illegal. However, using the alphabet to write libelous statements, death threats, or state secrets constitutes illegal activity. The illegality stems from the content and intent, not the letters used.[20] Binary code functions in precisely the same way for digital information.

It is worth noting a potential point of semantic confusion: within specific technical contexts, certain binary patterns might be referred to as "illegal." For example, in Binary-Coded Decimal (BCD) encoding, only ten specific 4-bit patterns (representing decimal digits 0-9) are considered valid or "legal." The remaining six possible 4-bit patterns are "illegal" in the context of BCD because they don't correspond to a valid decimal digit and might cause errors if processed by BCD arithmetic units.[3] This is a purely *technical* definition of illegality within a specific encoding standard and bears no relation to the *legal* status of binary code under jurisdictional law. Such technical usage, however, could potentially confuse non-experts researching the term.

C. Legal Protections for Code: Copyright and Licensing

While binary code as a system is not illegal, specific instances of code, including binary code, are subject to legal frameworks, primarily copyright and contract law (licensing). Computer programs, whether in human-readable source code or machine-readable binary/object code, are generally classified as "literary works" under copyright law and are eligible for protection.[30] This protection arises automatically upon creation in a tangible medium.[28] Copyright safeguards the particular *expression* of the programmer's ideas—the specific sequence of instructions and structure—but not the underlying algorithms, concepts, or functions themselves, which may be implemented differently by others.[30]

Binary code is frequently considered a derivative work of the original source code, meaning it inherits copyright protection from the source.[14] Consequently, the unauthorized reproduction, distribution, or creation of derivative works based on copyrighted binary code (such as commercial software) constitutes copyright infringement and is illegal.[28] Penalties for infringement can be severe, including substantial statutory damages and potentially criminal charges for willful infringement.[36]

Furthermore, most software is distributed not through a sale of the code itself, but under the terms of a license agreement (e.g., End User License Agreements (EULAs) for commercial software [37], or open-source licenses like the GPL [25]). These licenses grant users specific permissions while imposing restrictions on how the binary (and sometimes source) code can be used, copied, modified, or redistributed.[28] Common restrictions in commercial licenses include prohibitions on making copies (beyond potentially one archival backup [28]), modification, decompilation (converting binary back towards source code), or reverse engineering.[28] Violating the terms of a software license is a breach of contract and can lead to legal action. While copyright law and licenses restrict decompilation, legal exceptions may exist, such as the right under EU law to decompile to achieve interoperability between programs under specific conditions.[34]

The existence of these legal controls *around* specific binary files likely contributes to the misconception that binary itself is restricted. Users constantly interact with software primarily in its binary form and encounter copyright warnings [28] and EULAs [42] stating that unauthorized copying or modification is illegal. This repeated association of *specific binary programs* with legal prohibitions could easily lead someone unfamiliar with the nuances to incorrectly generalize that the underlying binary format itself carries inherent legal restrictions or is somehow "illegal."

Table 1: Binary Code (System) vs. Data Represented in Binary (Content)

Concept	**Representation**	**Legal Status**	**Example Context**
Binary Number System	Method using 0s and 1s	Legal (Fundamental Concept)	The base-2 system itself [1]
Legitimate Software (e.g., OS, App)	Specific sequence of 0s/1s	Legal (Copyrightable, subject to license)	Licensed commercial or open-source programs [30]
Malware (e.g., Virus, Ransomware)	Specific sequence of 0s/1s	Illegal (Harmful Content & Function)	Code designed to damage systems or extort users [26]
Pirated Software	Specific sequence of 0s/1s	Illegal (Copyright Infringement via Unauthorized Copy)	Unauthorized copies of copyrighted programs [28]
Circumvention Tool (e.g., DeCSS Binary)	Specific sequence of 0s/1s	Illegal to Traffic (DMCA Violation)	Tools designed to bypass copyright protection measures [44]
Encryption Key (e.g., AACS Key)	Specific sequence of 0s/1s	Illegal to Traffic (DMCA Violation)	Keys used as part of a circumvention process [20]
Illegal Image (e.g., Obscenity)	Specific sequence of 0s/1s	Illegal (Prohibited Content)	Digital files containing legally prohibited imagery [20]
Invalid BCD Value	Specific 4-bit sequence (e.g., 1111)	"Illegal" (Technical Term - Invalid Data	Values outside 0-9 in Binary-Coded Decimal

		Format)	representation [3]

III. Sources of the Misconception: Binary Code's Association with Unlawful Acts

The notion that binary code might be illegal, while fundamentally incorrect, does not arise in a vacuum. It stems from the undeniable fact that binary code is the medium through which various forms of illegal digital activity are conducted. The constant association of binary with these negative contexts fosters the misconception.

A. The Vehicle for Malware and Viruses

Malicious software, encompassing a wide range of threats like viruses, worms, Trojan horses, ransomware, spyware, and botnets, exists and operates as executable binary code.[26] When malware infects a system, it is the execution of this binary code by the computer's processor that causes harm, such as data theft, system damage, unauthorized access, or extortion.[48]

Cybersecurity professionals and researchers frequently engage in binary code analysis to understand how malware functions, identify its capabilities, and develop defenses.[26] This is often necessary because the original source code for malware is typically unavailable or deliberately withheld by its creators.[26] Malware authors further strengthen this association by employing techniques specifically designed to manipulate the binary representation of their malicious programs. Code obfuscation involves rewriting the binary code into a functionally equivalent but more confusing form to hinder analysis and defeat signature-based detection systems.[50] Binary padding involves adding junk data to the executable file to increase its size, potentially bypassing security scanners with file size limits, or to alter the file's hash value, thus evading hash-based detection lists.[51]

The sheer scale of the malware problem reinforces the link between binary code and malicious activity. Hundreds of thousands of new malware samples are detected daily [52], with over a billion distinct malware programs estimated to exist.[52] News headlines and security reports constantly discuss threats posed by malicious binaries.[52] This pervasive discourse creates a strong, albeit misleading, association in the public mind. Adding to the confusion, sometimes legitimate software can be misidentified as malware by overzealous antivirus programs due to similarities in their binary structure, particularly if they include large runtime libraries, as seen with some Go programs.[60]

B. The Medium for Software Piracy and Cracking Tools

Software piracy, the unauthorized reproduction and distribution of copyrighted software, overwhelmingly involves the distribution of the software in its compiled, binary executable format.[28] End users obtain and run these illegal copies, violating copyright law.[28]

Closely related are software cracking tools—programs like patches, key generators (keygens), and loaders—specifically designed to bypass or disable software protection mechanisms such as license verification, activation requirements, or copy protection schemes.[29] These tools often function by directly modifying the target application's binary code (a process known as "patching") to alter its execution flow, for example, by removing checks that verify a legitimate license.[29] Keygens work by reverse-engineering the algorithms used to generate valid serial numbers or activation keys from the software's binary code, allowing users to generate codes that trick the software into activating.[29]

The distribution channels for pirated software and cracking tools—peer-to-peer (P2P) networks, file-sharing websites, dubious online marketplaces—are notorious not only for facilitating copyright infringement but also for frequently bundling malware with the pirated goods.[29] Studies have shown a high prevalence of malware infections originating from the installation of pirated software.[49] This creates a double layer of illegality associated with these binary files. Legal frameworks like copyright law directly prohibit unauthorized copying [28], while laws such as the Digital Millennium Copyright Act (DMCA) in the US specifically outlaw the trafficking in tools designed to circumvent technological protection measures.[41] Since both the act of piracy and the tools used for cracking involve the distribution and manipulation of binary code, the connection between binary and illegality is strongly reinforced in this context.

C. The Language of Hacking and Cybercrime

Hacking and various forms of cybercrime often necessitate interacting with computer systems at a fundamental level, which frequently involves analyzing, manipulating, or injecting binary code.[51] Techniques classified as "Living Off The Land" (LOTL) attacks, for instance, might involve "binary planting" (also known as DLL hijacking), where an attacker replaces a legitimate system binary file (a Dynamic Link Library) with a malicious one, causing the operating system or applications to load and execute the malicious code.[67]

Cybercrime legislation, such as the US Computer Fraud and Abuse Act (CFAA) [22] and the UK's Computer Misuse Act (CMA) [43], criminalizes activities like unauthorized access to computer systems, intentionally causing damage through the transmission of malicious programs or code, and trafficking in passwords or other access credentials. These illegal acts are typically carried out using software tools that operate on or generate binary data. The CMA, for example, explicitly includes an offense (Section 3A) related to the creation, supply, or obtaining of articles (often software in binary form) intended for use in computer misuse offenses, directly targeting malware creation and distribution.[43]

Furthermore, sophisticated techniques like steganography can be employed by malicious actors to conceal harmful binary code within seemingly harmless files, such as images or audio tracks.[68] This practice of hiding illicit binary data within legitimate-looking binary data further cements the association of binary with covert and illegal activities. The increasing use of Artificial Intelligence (AI) by cybercriminals to automate the generation of polymorphic malware (malware that constantly changes its binary structure to evade detection), develop new exploits, and create sophisticated phishing tools also contributes to the perception of binary code as a tool intrinsically linked to advanced cyber threats.[72]

D. The Format for Transmitting Illegal Content

Beyond software-related crimes, binary code serves as the universal format for *all* digital information that might be deemed illegal to possess, transmit, or distribute under various laws. This category is broad and can include classified government secrets, proprietary corporate trade secrets, illegal pornography (particularly child exploitation material), certain forms of hate speech, or other types of legally proscribed content.[20]

Because any digital file—be it a document, an image, a video, or a database—can be represented as a sequence of binary digits (effectively, a very large number), the legal status of the *content* dictates the legal implications of possessing or transmitting that specific binary sequence.[12] If the underlying information is illegal, then the binary string representing it becomes a carrier of illegal content.

The constant stream of news coverage, security alerts, and public awareness campaigns focusing on malware, software piracy, hacking incidents, and the digital transmission of illegal materials creates a significant perception bias. The *negative* and *illegal* applications of binary code are far more visible in public discourse than the routine, legitimate functioning of the vast majority of binary code that underpins our digital infrastructure and daily activities. This imbalance naturally leads observers to disproportionately associate the term "binary" with its illicit uses.

Adding to the complexity is the dual nature of many tools involved. Debuggers, disassemblers, hex editors, and network analysis tools are essential for legitimate software development, debugging, security research, and ensuring interoperability.[27] However, these same tools can be used by malicious actors to reverse engineer software for cracking purposes, analyze systems for vulnerabilities to exploit, or develop malware.[29] The illegality arises not from the tools themselves or their ability to operate on binary code, but from the *intent* and *context* of their use. This inherent duality makes it difficult to simply ban the tools or techniques associated with binary manipulation, and this nuance is often lost, leading to a potential conflation of the tool/medium with the illegal act.

Finally, the sheer volume and increasing automation of malicious binary code creation contribute to the perception problem.[52] When faced with reports of millions of attacks and hundreds of thousands of new malware binaries appearing daily, it becomes easier to perceive the binary format itself as inherently problematic or intrinsically linked to illegality, simply due to the overwhelming scale of its misuse.

IV. When Numbers Become Contentious: The "Illegal Number" Cases

Perhaps the most direct source of the misconception that binary code can be illegal stems from specific, high-profile legal controversies where particular sequences of binary digits, represented as numbers or keys, were declared "illegal" to possess or distribute under specific laws, most notably the US Digital Millennium Copyright Act (DMCA).

A. The Concept of "Illegal Numbers" Explained

The term "illegal number" refers to a number whose digital representation (usually binary) corresponds to information that is restricted by law in a particular jurisdiction.[20] Since any piece of digital data—an encryption key, software code, a classified document—can be expressed as a sequence of bits, and that sequence of bits can be interpreted as a number, laws restricting the possession or dissemination of that data effectively render the corresponding number "illegal" in that context.[12]

This concept gained prominence primarily through legal battles surrounding technologies designed to circumvent copyright protection mechanisms, falling under the purview of the DMCA's anti-circumvention clauses.[20] It could theoretically also apply to numbers representing trade secrets or classified information.[20] The core idea is that the prohibition attaches to the information encoded within the number, making the number itself a restricted item.

B. Case Study: DeCSS, Copyright Circumvention, and the Prime Number Protest

A landmark example involves DeCSS (Content Scramble System Decoder), a computer program developed in 1999 primarily by Norwegian teenager Jon Johansen and others.[44] DeCSS was designed to decrypt the CSS encryption used to protect commercial DVDs from unauthorized copying.[13] The initial motivation cited by its creators was to enable DVD playback on computers running the Linux operating system, for which no licensed players existed at the time.[44]

However, the motion picture industry viewed DeCSS as a tool facilitating piracy. Eight major studios filed lawsuits against individuals and websites, most notably Eric Corley and his publication *2600: The Hacker Quarterly*, for publishing the DeCSS source code and binary, or linking to sites that did.[13] The studios argued that distributing DeCSS constituted trafficking in a circumvention device, violating Section 1201(a)(2) of the DMCA.[13]

In the key ruling, *Universal City Studios, Inc. v. Reimerdes* (later *Universal City Studios, Inc. v. Corley* on appeal), the courts largely agreed with the motion picture industry.[13] Judge Lewis Kaplan issued an injunction prohibiting 2600.com from posting the DeCSS code or even linking to other sites hosting it.[75] This case represented a significant early victory for copyright holders under the DMCA's anti-circumvention provisions and established a precedent that distributing such code could be legally prohibited.[44]

The ruling sparked outrage among free speech advocates and members of the tech community who saw it as censorship of code. As a form of protest, computer scientist Phil Carmody discovered a large prime number (a number divisible only by 1 and itself) whose binary representation corresponds exactly to the compressed C source code for the DeCSS program.[20] The act was intended to highlight the perceived absurdity of making a fundamental mathematical entity—a prime number—illegal simply because of the information it happened to encode.[24] This "illegal prime" became a symbol of the conflict. Others engaged in similar protests, embedding the DeCSS code steganographically within images ("Free Speech Flags"), music, poetry, haikus, and even T-shirt designs, challenging the boundaries of the injunction and the definition of "code" versus "speech".[20]

C. Case Study: The AACS Encryption Key Controversy and the DMCA

A similar controversy erupted in 2007 concerning the Advanced Access Content System (AACS), the encryption technology used for HD DVD and Blu-ray discs.[46] A specific 128-bit processing key, often represented in hexadecimal as 09 F9 11 02 9D 74 E3 5B D8 41 56 C5 63 56 88 C0, was discovered by hackers (reportedly by examining the memory of software DVD players) and published online.[20] This key could be used as part of the process to decrypt AACS-protected content.

The AACS Licensing Administrator (AACS LA), the consortium controlling the technology, swiftly responded by sending DMCA cease-and-desist notices to websites hosting the key.[20] Their legal position was that the key itself constituted a circumvention device or technology under the DMCA, making its publication and distribution illegal.[46] The key was widely dubbed an "illegal number".[46]

The attempt to suppress the key backfired spectacularly, particularly on the popular news aggregation site Digg. When Digg administrators initially complied with the DMCA notice and started removing posts containing the key, the user community revolted.[46] In a phenomenon now known as the Streisand effect, users flooded Digg and other internet platforms with posts containing the key, often disguised in creative ways—encoded in images, songs, poems, T-shirt designs, and even flags.[46] The sheer volume of postings made censorship effectively impossible, and Digg eventually reversed its decision to remove the posts.[80]

This incident further intensified the public debate surrounding "illegal numbers" and the feasibility of censoring information in the digital age.[46] It demonstrated the practical challenges faced by rights holders attempting to control the dissemination of cryptographic keys or code once released into the wild. Ultimately, the AACS LA shifted its strategy towards revoking compromised keys in the encryption schemes used for future disc releases, rendering the leaked key less useful over time, rather than relying solely on legal threats and censorship attempts.[78]

These high-profile cases involving DeCSS and the AACS key are likely the most significant contributors to the misconception that "binary is illegal." They provide concrete examples where specific sequences of 0s and 1s, representing functional code or cryptographic keys, were indeed targeted by legal actions and labeled as illegal circumvention devices under the DMCA. While the legal action targeted the specific *content* and *function* of these binary sequences (as circumvention tools), a casual observer might easily misinterpret these events as implying that binary code itself can be inherently illegal.

The DMCA itself, particularly its anti-circumvention provisions in Section 1201 [45], plays a crucial role. By shifting the legal focus from traditional copyright infringement (unauthorized copying of the work) to prohibiting the tools and technologies (often embodied in binary code) used to bypass access controls, the DMCA created the specific legal context in which certain types of code became subject to prohibition, regardless of whether actual infringement occurred.[65] This legislative framework is central to the "illegal number" phenomenon.

Ironically, the various forms of protest surrounding these cases—the illegal prime, the free speech flags, the Digg revolt—while intended to critique and resist the notion of illegal numbers, also served to amplify the controversy and cement the association between "binary" and "illegality" in the public consciousness.[20] The very act of protesting the concept inadvertently reinforced the idea that binary representations could be subject to legal prohibition, further muddying the waters for those seeking clarity.

D. Broader Implications: Free Speech, Censorship, and the "Code as Speech" Debate

The "illegal number" controversies directly engage fundamental principles of free speech, particularly under the First Amendment of the US Constitution. Defendants and critics in the DeCSS and AACS cases argued vigorously that computer code, including both human-readable source code and machine-executable binary code, constitutes a form of expressive speech.[13] From this perspective, banning the publication or distribution of code amounts to government censorship of expression.

U.S. courts have generally acknowledged the expressive aspects of code.[13] Code communicates ideas, algorithms, and logic between programmers and, in the case of source code, can be read and understood by humans. However, courts have consistently grappled with the dual nature of code: it is not only expressive but also inherently *functional*—it directly causes computers to perform actions.[13] In the *Corley* decision regarding DeCSS, Judge Kaplan explicitly recognized code as speech but held that its functional capacity to circumvent CSS justified restricting its distribution under the DMCA, applying a lower level of scrutiny than typically afforded to "pure speech".[13]

This "code as speech" debate remains a complex and evolving area of law.[20] Treating all code as fully protected speech could severely hamper necessary regulation of harmful software, such as malware, discriminatory algorithms used in decision-making systems, or code facilitating illegal activities.[85] Conversely, denying code any speech protection and treating it purely as functional conduct could chill innovation, security research, and legitimate forms of expression that rely on code.[85] The central challenge lies in applying traditional First Amendment doctrines, developed primarily for human language and symbolic expression, to a medium that is simultaneously a language for human communication and a direct instruction set for machines.[33]

Furthermore, the notion of making numbers illegal raises profound philosophical questions.[20] Can fundamental mathematical or informational entities, often discovered or generated rather than creatively authored, be subject to ownership or legal restriction? Is it appropriate to outlaw a number simply because of the information it represents, especially when that representation might be arbitrary or coincidental? These questions touch upon the nature of information, the limits of intellectual property, and the balance between protecting rights holders and ensuring the free flow of information and ideas.[20]

V. Legal Nuances and Related Concepts

Understanding the legal landscape surrounding binary code requires appreciating several related concepts and distinctions, including the differences between source and binary code in legal treatment, the implications of code obfuscation, and the role of trade secret law.

A. Source Code vs. Binary/Object Code in Law

Legally, both human-readable source code and machine-executable object code (binary code) are generally considered different forms of expression of the same underlying computer program and are thus protectable by copyright.[30] Object code is often legally viewed as a direct translation or derivative work of the source code, inheriting its copyright status.[14] The EU's directive on the legal protection of computer programs, for instance, explicitly states that protection applies to the expression "in any form" of a computer program.[34]

However, some legal distinctions and debates arise. In the context of the First Amendment "code as speech" arguments, some have suggested that source code warrants stronger protection because it is directly intelligible to human programmers and serves as a medium for communicating complex ideas, whereas object code is primarily functional, intended for machine execution.[33] Despite this argument, U.S. courts have often found expressive elements in both forms, although the functional aspect of object code has been used to justify greater regulation (as in the DeCSS case).[13]

Practical differences are significant. Commercial software licenses typically provide users only with the binary/object code and explicitly restrict access to the source code, viewing it as a valuable trade secret.[14] Conversely, open-source licenses (like the GPL) often mandate the availability of source code along with the binary, facilitating collaboration and modification.[25] Legal actions involving reverse engineering or decompilation inherently focus on the object/binary code, as the goal is often to understand its functionality or recreate aspects of the original source code.[28] Anti-circumvention laws, like the DMCA and the EU's Copyright Directive (EUCD), frequently impact the legality of analyzing binary code, especially when it is protected by Technological Protection Measures (TPMs).[34]

B. Code Obfuscation and Its Legal Relevance

Code obfuscation refers to techniques used to deliberately make source code or machine code more difficult for humans or automated tools (like decompilers or debuggers) to understand, while preserving the code's original functionality.[42] Legitimate software developers may use obfuscation as a form of "security through obscurity" to protect intellectual property embedded in the code (such as proprietary algorithms or trade secrets) or to make it harder for pirates to crack licensing mechanisms by hindering reverse engineering efforts.[35]

Conversely, malware authors make extensive use of obfuscation techniques to disguise their malicious code, bypass antivirus detection signatures, and impede analysis by security researchers.[50]

From a legal standpoint, applying obfuscation techniques to someone else's copyrighted code without permission would almost certainly be considered the creation of a derivative work, thus constituting copyright infringement.[95] Obfuscation does not strip the original code of its copyright protection or grant the obfuscator any rights.[95]

The DMCA's anti-circumvention provisions could potentially come into play if the obfuscation itself is sophisticated enough to be considered a "technological measure" controlling access to the underlying copyrighted work (the code).[42] Bypassing such obfuscation could then potentially violate Section 1201. However, disputes involving obfuscated code are more commonly addressed under standard copyright law (as unauthorized derivative works) or contract law (if the obfuscation violates terms of a license agreement).

C. Trade Secrets Embodied in Code

Computer software, particularly its source code, but sometimes also unique algorithms or data structures within compiled binary code, can qualify for protection as a trade secret.[30] To qualify, the information must derive independent economic value from not being generally known, and the owner must take reasonable measures to maintain its secrecy (e.g., through confidentiality agreements, limited access, and potentially obfuscation).[30]

The theft or misappropriation of trade secrets embodied in code is illegal under various laws, including state implementations of the Uniform Trade Secrets Act and federal statutes like the Defend Trade Secrets Act (DTSA) [98] and the Economic Espionage Act (EEA).[96] The EEA imposes criminal penalties, particularly for theft intended to benefit foreign governments or entities.[96]

Numerous court cases involve allegations of trade secret theft via code, often where employees allegedly copy proprietary source code or binary files before leaving to join a competitor or start a rival company.[70] These cases sometimes involve sophisticated methods to exfiltrate the code, such as encryption or steganography (hiding code within seemingly innocuous files like images).[71]

While trade secret law generally does not prohibit discovering secrets through legitimate means like independent invention or reverse engineering of a lawfully acquired product [41], it strongly protects against misappropriation through improper means. This includes breaching confidentiality agreements, industrial espionage, or unauthorized access to and copying of computer systems containing the secret code.[70]

The existence of these multiple, overlapping legal frameworks—copyright, contract law (licensing), trade secret law, anti-circumvention statutes (DMCA/EUCD), and potentially First Amendment considerations—means that the legality of any action involving binary code is highly context-dependent. An act like decompiling a binary file could simultaneously touch upon copyright (potential unauthorized reproduction), license terms (potential breach of contract), DMCA rules (if TPMs are bypassed), and trade secret law (if confidential algorithms are revealed), while potentially being permissible under narrow exceptions like those for interoperability or security research. This inherent legal complexity undoubtedly contributes to confusion about the status of binary code.

This complexity also reflects the persistent tension between code's dual nature as both expression and function. Copyright law primarily aims to protect the creative expression.[30] However, the practical value and the focus of legal disputes often lie in the code's *functionality*—what it enables a computer to *do*. Anti-circumvention laws target the function of bypassing protection.[45] Trade secret laws protect the functional advantage provided by secret code.[96] This demonstrates that while copyright formally protects expression, other legal regimes grapple more directly with the power and consequences of the actions that binary code performs, leading to different forms of regulation and restriction.

Table 2: Key Legal Frameworks Governing Code and Digital Content

Legal Framework	Primary Focus Concerning Code/Binary	Key Prohibitions/Protections	Example Snippet References
Copyright Law (e.g., 17 U.S.C.)	Protects original expression in source/binary code	Unauthorized copying, distribution, creation of derivative works	28
DMCA § 1201 (Anti-Circumvention - US)	Prohibits bypassing TPMs & trafficking in circumvention tools (code/binary)	Circumventing access controls; Manufacturing/distributing circumvention tools	20
EU Copyright Directive (InfoSoc) (Art. 6 & 8)	Protects TPMs & prohibits circumvention/trafficking of tools	Circumventing effective TPMs; Manufacturing/distributing circumvention tools/services	93
Computer Fraud & Abuse Act (CFAA - US)	Prohibits unauthorized computer access & damage	Hacking, transmitting malware, password trafficking, exceeding authorized access	22
Computer Misuse Act (CMA - UK)	Prohibits unauthorized computer access, modification, & tool supply	Hacking, impairing computer operation (e.g., DDoS), making/supplying malware	43
Trade Secret Law (DTSA, EEA - US; State UTSA)	Protects confidential information (can be code) providing	Misappropriation (theft) of trade secrets through	30

	economic value	improper means (breach of duty, etc.)	
Contract Law (Licenses)	Governs use/distribution of specific software (binary/source)	Violating license terms (e.g., decompilation, unauthorized copying, exceeding users)	37
First Amendment (US Constitution)	Potential protection for code as expressive speech	Government censorship/regulation of code (subject to balancing tests for functionality)	13

VI. Conclusion: Demystifying Binary Code's Legality

The question of whether binary code is illegal stems from a fundamental misunderstanding of its nature and its relationship with the law. By dissecting the technical definition, exploring its association with illicit activities, and examining relevant legal frameworks and cases, a clear conclusion emerges.

A. Reaffirming the Distinction: Binary is Not Illegal

The central finding of this report is unequivocal: binary code, the base-2 system of 0s and 1s used to represent all digital information, is not illegal. It is a foundational concept in mathematics and computer science, serving as a neutral medium for encoding data and instructions. It is analogous to the alphabet, musical notation, or the decimal number system—tools for representation that are inherently legal.

Legality, in the context of digital information, attaches to the *content* being represented or the *actions* being performed by software, not to the binary system used for representation. Illegality arises when binary code is used to embody malware, infringing copies of copyrighted works, prohibited content like child exploitation material, stolen trade secrets, or tools specifically designed to circumvent lawful technological protection measures under statutes like the DMCA. The binary format itself remains neutral and lawful.

B. Summarizing the Sources of Confusion

The misconception that binary code might be illegal appears to originate from several overlapping factors:

1. **Conflation of Medium and Message:** A failure to distinguish between the binary system (the medium) and the specific data or software it represents (the message).
2. **Association with Illicit Activities:** The constant and highly visible use of binary code as the vehicle for malware distribution, software piracy, hacking tools, and the transmission of illegal content creates a strong negative association.
3. **"Illegal Number" Controversies:** High-profile legal battles surrounding DeCSS and the AACS encryption key, where specific binary sequences were targeted under the DMCA and labeled "illegal numbers," provided concrete—though context-specific—examples that fueled the misconception.
4. **Copyright and Licensing Restrictions:** Users encounter legal restrictions (copyright notices, EULAs) tied to specific *binary files* (software), which can be misinterpreted as restrictions on the binary format itself.
5. **Technical Jargon:** The use of terms like "illegal" in specific technical contexts (e.g., invalid BCD values) can add to the confusion for non-experts.

C. Recommendations for Clarity and Understanding

To combat this misconception and foster a clearer understanding of the legal status of binary code, the following points are crucial:

1. **Use Precise Language:** It is important to differentiate between "binary code" as the fundamental system and specific instances like "a program's binary code," "malicious binary code," or "a binary representation of illegal data."
2. **Emphasize the Representation/Content Distinction:** Educational efforts should consistently highlight that binary is merely a way to represent information, and legality depends on the nature and use of that information.
3. **Nuanced Understanding of Laws:** Discussions about laws like the DMCA should clarify that they target specific *functions* (circumvention) or *types* of code (trafficking in circumvention tools), not the binary system itself. The existence of exceptions (e.g., for security research, interoperability) should also be noted where applicable.
4. **Acknowledge Complexity:** The ongoing "code as speech" debate illustrates that the legal system is still adapting to the unique dual nature of code as both expression and function. Acknowledging this complexity is more accurate than seeking overly simplistic legal classifications for all code.

In conclusion, binary code is a fundamental and essential component of modern technology, and it holds no inherent illegal status. The laws governing the digital world focus on the substance of information and the impact of actions, not the underlying system of 0s and 1s used to represent them. Recognizing this distinction is key to navigating the complexities of technology law and demystifying the relationship between code and legality.

Works cited

1. en.wikipedia.org, accessed April 19, 2025, https://en.wikipedia.org/wiki/Binary_code#:~:text=A%20binary%20code%20represents%20text,each%20character%2C%20instruction%2C%20etc.
2. Binary code | Definition, Numbers, & Facts - Britannica, accessed April 19, 2025, https://www.britannica.com/technology/binary-code
3. Binary code - Wikipedia, accessed April 19, 2025, https://en.wikipedia.org/wiki/Binary_code
4. Binary Number System | Encyclopedia.com, accessed April 19, 2025, https://www.encyclopedia.com/computing/news-wires-white-papers-and-books/binary-number-system
5. Binary Code explained - IONOS, accessed April 19, 2025, https://www.ionos.com/digitalguide/websites/web-development/binary-code/
6. What Is Binary Code? - Coursera, accessed April 19, 2025, https://www.coursera.org/articles/binary-code
7. Teaching binary code with a secret word challenge - Science in School, accessed April 19, 2025, https://www.scienceinschool.org/article/2021/teaching-binary-code-secret-word-challenge/
8. ELI5: How exactly has binary code led to modern day programming languages? - Reddit, accessed April 19, 2025, https://www.reddit.com/r/explainlikeimfive/comments/1iudgki/eli5_how_exactly_has_binary_code_led_to_modern/
9. Binary - So Simple a Computer Can Do It - KerryR.net, accessed April 19, 2025, https://www.kerryr.net/pioneers/binary.htm
10. How does a computer "understand" what code/binary means? : r/computerscience - Reddit, accessed April 19, 2025, https://www.reddit.com/r/computerscience/comments/p7nprt/how_does_a_computer_understand_what_codebinary/
11. Binary number - Wikipedia, accessed April 19, 2025, https://en.wikipedia.org/wiki/Binary_number
12. Illegal Numbers and the Dangers of Banned Information, accessed April 19, 2025, https://blog.serindu.com/2013/10/15/illegal-numbers-and-the-dangers-of-banned-information/
13. Advanced Constitutional Law : Universal City Studios, Inc. v. Reimerdes - Open Casebooks, accessed April 19, 2025, https://opencasebook.org/casebooks/523-advanced-constitutional-law/resource

s/6.1-universal-city-studios-inc-v-reimerdes/
14. I believe that compiled binary programs should not be protected by copyright law. CMV : r/changemyview - Reddit, accessed April 19, 2025, https://www.reddit.com/r/changemyview/comments/1l89fh/i_believe_that_compiled_binary_programs_should/
15. History of Binary Code in Computer Science - Google Docs, accessed April 19, 2025, https://docs.google.com/document/d/1bP0dYWh_BdsavnUybyjwnPEfBbTg0b8h3Jyr4NnhXFI/edit
16. Development of the Binary Number System and the Foundations of Computer Science - ScholarWorks at University of Montana, accessed April 19, 2025, https://scholarworks.umt.edu/cgi/viewcontent.cgi?article=1315&context=tme
17. Origin and History of the Binary System - Luis Llamas, accessed April 19, 2025, https://www.luisllamas.es/en/origin-and-history-of-the-binary-system/
18. Development of the Binary Number System and the Foundations of Computer Science, accessed April 19, 2025, https://scholarworks.umt.edu/tme/vol11/iss3/6/
19. ELI5: How was Binary Code "discovered" in the 1600's? What were its applications? - Reddit, accessed April 19, 2025, https://www.reddit.com/r/explainlikeimfive/comments/59xmq7/eli5_how_was_binary_code_discovered_in_the_1600s/
20. Illegal number - Wikipedia, accessed April 19, 2025, https://en.wikipedia.org/wiki/Illegal_number
21. Illegal Explosives | Bureau of Alcohol, Tobacco, Firearms and Explosives - ATF, accessed April 19, 2025, https://www.atf.gov/explosives/tools-services-explosives-industry/explosive-products-and-devices/illegal-explosives
22. 18 U.S. Code § 1030 - Fraud and related activity in connection with computers, accessed April 19, 2025, https://www.law.cornell.edu/uscode/text/18/1030
23. 18 USC 1030: Fraud and related activity in connection with computers - OLRC Home, accessed April 19, 2025, https://uscode.house.gov/view.xhtml?req=(title:18%20section:1030%20edition:prelim)
24. If everything is ultimately binary, does't that just make programs a specific number? - Reddit, accessed April 19, 2025, https://www.reddit.com/r/learnprogramming/comments/ue5hh6/if_everything_is_ultimately_binary_doest_that/
25. Is there a loophole in the GPL allowing binary-only distribution? - Law Stack Exchange, accessed April 19, 2025, https://law.stackexchange.com/questions/27768/is-there-a-loophole-in-the-gpl-allowing-binary-only-distribution
26. Static Analysis on Binary Code, accessed April 19, 2025, https://engineering.lehigh.edu/sites/engineering.lehigh.edu/files/_DEPARTMENTS/cse/research/tech-reports/2012/lu-cse-12-001.pdf
27. How Far Have We Gone in Stripped Binary Code Understanding Using Large Language Models - arXiv, accessed April 19, 2025,

https://arxiv.org/html/2404.09836v1
28. Guide to Legal and Ethical Use of Software | WashU, accessed April 19, 2025, https://washu.edu/policies/guide-to-legal-and-ethical-use-of-software/
29. Software cracking - Wikipedia, accessed April 19, 2025, https://en.wikipedia.org/wiki/Software_cracking
30. Intellectual Property Protections for Software - Barr Group, accessed April 19, 2025, https://barrgroup.com/blog/intellectual-property-protections-software
31. Can Computer Code be Copyrighted? - Easler Law, accessed April 19, 2025, https://easlerlaw.com/software-computer-code-copyrighted
32. Legal Rights in Computer Software - Santa Clara Law Digital Commons, accessed April 19, 2025, https://digitalcommons.law.scu.edu/cgi/viewcontent.cgi?article=1123&context=chtlj
33. Code as Speech: a discussion of Bernstein v. USDOJ, Karn v. USDOS, and Junger v. Daley in light of the US Supreme Court's recent shift to Federalism - L Jean Camp, accessed April 19, 2025, http://www.ljean.com/files/CODE_FEDERALISM.pdf
34. Reference for a preliminary ruling – Copyright and related ... - CURIA, accessed April 19, 2025, https://curia.europa.eu/juris/document/document_print.jsf;jsessionid=65A4A57EAA6D78A82DD59E1FC9E08B02?docid=238707&text=&dir=&doclang=EN&occ=first%C3%A2%C2%88%C2%82%3D1&mode=DOC&pageIndex=0&cid=285070
35. Protecting Source Code from Copyright Infringement: A Guide | ScoreDetect Blog, accessed April 19, 2025, https://www.scoredetect.com/blog/posts/protecting-source-code-from-copyright-infringement-a-guide
36. Digital Millennium Copyright Act (DMCA) - UCI Office of Academic Integrity & Student Conduct, accessed April 19, 2025, https://conduct.uci.edu/dmca/
37. Sun Binary Code License Agreement, accessed April 19, 2025, https://wordhoard.library.northwestern.edu/userman/thirdparty/sunbinary.html
38. Sun Microsystems, Inc. Binary Code License Agreement - Klocwork Documentation, accessed April 19, 2025, https://help.klocwork.com/2024/en-us/concepts/sunmicrosystemsincbinarycodelicenseagreement.htm
39. Oracle Binary Code License Agreement for the Java SE Platform Products and JavaFX, accessed April 19, 2025, https://www.oracle.com/downloads/licenses/binary-code-license.html
40. Licensing/Sun Binary Code License Agreement - Fedora Project Wiki, accessed April 19, 2025, https://fedoraproject.org/wiki/Licensing/Sun_Binary_Code_License_Agreement
41. Reverse Engineering and the Law: Understand the Restrictions to Minimize Risks, accessed April 19, 2025, https://ipwatchdog.com/2021/03/27/reverse-engineering-law-understand-restrictions-minimize-risks/id=131543/
42. Coders' Rights Project Reverse Engineering FAQ - Electronic Frontier Foundation,

accessed April 19, 2025, https://www.eff.org/issues/coders/reverse-engineering-faq
43. Cybercrime - prosecution guidance, accessed April 19, 2025, https://www.cps.gov.uk/legal-guidance/cybercrime-prosecution-guidance
44. Anticircumvention under the Digital Millennium Copyright Act: Universal Studios v. Corley - EveryCRSReport.com, accessed April 19, 2025, https://www.everycrsreport.com/reports/RL31257.html
45. 17 U.S. Code § 1201 - Circumvention of copyright protection systems - Cornell Law School, accessed April 19, 2025, https://www.law.cornell.edu/uscode/text/17/1201
46. AACS encryption key controversy - Wikipedia, accessed April 19, 2025, https://en.wikipedia.org/wiki/AACS_encryption_key_controversy
47. Static Analysis of Binary Code to Isolate Malicious Behaviors. - ResearchGate, accessed April 19, 2025, https://www.researchgate.net/publication/221015581_Static_Analysis_of_Binary_Code_to_Isolate_Malicious_Behaviors
48. Design of Malicious Code Detection System Based on Binary Code Slicing, accessed April 19, 2025, http://www.csroc.org.tw/journal/JOC32-4/JOC3204-18.pdf
49. Unveiling the Connection Between Malware and Pirated Software in Southeast Asian Countries: A Case Study - DigitalCommons@UNL, accessed April 19, 2025, https://digitalcommons.unl.edu/cgi/viewcontent.cgi?article=1377&context=csearticles
50. Experiences in malware binary deobfuscation - Virus Bulletin, accessed April 19, 2025, https://www.virusbulletin.com/conference/vb2010/abstracts/experiences-malware-binary-deobfuscation/
51. How Hackers Use Binary Padding to Outsmart Sandboxes and Infiltrate Your Systems, accessed April 19, 2025, https://intezer.com/blog/how-hackers-use-binary-padding-to-outsmart-sandboxes/
52. 30+ Malware Statistics You Need To Know In 2025 - Astra Security, accessed April 19, 2025, https://www.getastra.com/blog/security-audit/malware-statistics/
53. 2024 Cybersecurity Statistics: The Ultimate List Of Stats, Data & Trends | PurpleSec, accessed April 19, 2025, https://purplesec.us/resources/cybersecurity-statistics/
54. Gen Q3/2024 Threat Report, accessed April 19, 2025, https://www.gendigital.com/blog/insights/reports/threat-report-q3-2024
55. Cybersecurity Statistics 2024: Key Insights and Numbers - NordLayer, accessed April 19, 2025, https://nordlayer.com/blog/cybersecurity-statistics-of-2024/
56. 82 Must-Know Data Breach Statistics [updated 2024] - Varonis, accessed April 19, 2025, https://www.varonis.com/blog/data-breach-statistics
57. 2024 Malware & Ransomware Defense Report | SpyCloud, accessed April 19, 2025, https://spycloud.com/resource/2024-malware-ransomware-defense-report/

58. Ransomware Statistics, Data, Trends, and Facts [updated 2024] - Varonis, accessed April 19, 2025, https://www.varonis.com/blog/ransomware-statistics
59. 2024 Data Breach Investigations Report | Verizon, accessed April 19, 2025, https://www.verizon.com/business/resources/reports/2024-dbir-data-breach-investigations-report.pdf
60. Golang binary getting detected as malware by multiple antivirus programs—how to avoid false positives? - Reddit, accessed April 19, 2025, https://www.reddit.com/r/golang/comments/1g5ioes/golang_binary_getting_detected_as_malware_by/
61. The Legal and Security Perils of Using Cracks and Keygens - Netizen Corporation, accessed April 19, 2025, https://www.netizen.net/news/post/4182/the-legal-and-security-perils-of-using-cracks-and-keygens
62. WHITE PAPER The Risks of Obtaining and Using Pirated Software - Microsoft, accessed April 19, 2025, https://download.microsoft.com/download/0/5/1/051C127F-3B9E-4A80-BD77-78B128301E39/IDCWhitePaperRisksofPiratedSoftware.pdf
63. Pirated Software May Contain Malware - FBI, accessed April 19, 2025, https://www.fbi.gov/news/stories/pirated-software-may-contain-malware1
64. The Anti-Circumvention Rules of the Digital Millennium Copyright Act - Stoel Rives LLP, accessed April 19, 2025, https://www.stoel.com/insights/publications/the-anti-circumvention-rules-of-the-digital-millen
65. CS-IP: Copyrights - DMCA - Duke Computer Science, accessed April 19, 2025, https://courses.cs.duke.edu/cps182s/fall02/cscopyright/Copyrights/Copyright-DMCA.htm
66. Hackers are duping developers with malware-laden coding challenges | IT Pro - ITPro, accessed April 19, 2025, https://www.itpro.com/security/hackers-are-duping-developers-with-malware-laden-coding-challenges
67. Living-Off-the-Land (LOTL) Attacks: Everything You Need to Know - Kiteworks, accessed April 19, 2025, https://www.kiteworks.com/risk-compliance-glossary/living-off-the-land-attacks/
68. Hackers' Latest Weapon: Steganography - IEEE Computer Society, accessed April 19, 2025, https://www.computer.org/publications/tech-news/research/how-steganography-works
69. Computer Fraud and Abuse Act: 10 Must-Know Facts & Tips, accessed April 19, 2025, https://chargebacks911.com/computer-fraud-and-abuse-act/
70. Corporate Recruiter Convicted of Computer Fraud and Trade Secret Theft By San Francisco Jury | Seyfarth Shaw, accessed April 19, 2025, https://www.tradesecretslaw.com/2013/04/articles/trade-secrets/corporate-recruiter-convicted-of-computer-fraud-and-trade-secret-theft-by-san-francisco-jury/
71. Were GE trade secrets hidden in photo headed for China? - E&E News by

POLITICO, accessed April 19, 2025, https://www.eenews.net/articles/were-ge-trade-secrets-hidden-in-photo-headed-for-china/
72. 5 Ways cybercriminals are using AI: Malware generation | Barracuda Networks Blog, accessed April 19, 2025, https://blog.barracuda.com/2024/04/16/5-ways-cybercriminals-are-using-ai--malware-generation
73. If computer code is ultimately just binary, and a string of binary can be converted into a number, does that mean I can communicate an entire program with a number? Can I count to doom given enough time? : r/askmath - Reddit, accessed April 19, 2025, https://www.reddit.com/r/askmath/comments/1inmcgj/if_computer_code_is_ultimately_just_binary_and_a/
74. The ethics of DeCSS posting: towards assessing the morality of the Internet posting of DVD copyright circumvention software - Information Research, accessed April 19, 2025, https://informationr.net/ir/11-4/paper273.html
75. DeCSS code-crack dispute back in court - ZDNet, accessed April 19, 2025, https://www.zdnet.com/article/decss-code-crack-dispute-back-in-court/
76. Banned Code Lives in Poetry and Song - CMU School of Computer Science, accessed April 19, 2025, https://www.cs.cmu.edu/~dst/DeCSS/Gallery/wsj-04-12-2001.html
77. Are DeCSS T-Shirts Dirty Laundry? Wearable, Non-Executable Computer Code as Protected Speech, accessed April 19, 2025, https://ir.lawnet.fordham.edu/cgi/viewcontent.cgi?article=1295&context=iplj
78. Advanced Access Content System - Wikipedia, accessed April 19, 2025, https://en.wikipedia.org/wiki/Advanced_Access_Content_System
79. AACS encryption key controversy - YouTube, accessed April 19, 2025, https://www.youtube.com/watch?v=tlB_cl3gXXE
80. Digg Users Revolt Over AACS Key - CITP Blog - Freedom to Tinker - Princeton University, accessed April 19, 2025, https://blog.citp.princeton.edu/2007/05/02/digg-users-revolt-over-aacs-key/
81. ELI5: The AACS encryption key controversy : r/explainlikeimfive - Reddit, accessed April 19, 2025, https://www.reddit.com/r/explainlikeimfive/comments/2xhbwh/eli5_the_aacs_encryption_key_controversy/
82. Why the 09ers Are So Upset - CITP Blog - Freedom to Tinker, accessed April 19, 2025, https://blog.citp.princeton.edu/2007/05/03/why-09ers-are-so-upset/
83. Implementing Sovereignty on the Internet, accessed April 19, 2025, https://cs.stanford.edu/people/eroberts/cs181/projects/implementing-sovereignity/page4.html
84. THE INTERNET MEETS OBI-WAN KENOBI IN THE COURT OF NEXT RESORT - Boston University, accessed April 19, 2025, https://www.bu.edu/law/journals-archive/scitech/volume82/chase.pdf
85. wlr.law.wisc.edu, accessed April 19, 2025, https://wlr.law.wisc.edu/wp-content/uploads/sites/1263/2021/12/15-Wang-Camera

-Ready.pdf
86. How Speech Lost Its Voice: The Informational Turn in US Free Speech Law - MPG.PuRe, accessed April 19, 2025, https://pure.mpg.de/rest/items/item_3334922/component/file_3401747/content
87. The Doctrinal Toll of Information as Speech | Loyola University Chicago Law Journal, accessed April 19, 2025, https://loyola-chicago-law-journal.scholasticahq.com/article/77039-the-doctrinal-toll-of-information-as-speech/attachment/161423.pdf
88. Code is Speech: Legal Tinkering, Expertise, and Protest Among Free and Open Source Software Developers - ResearchGate, accessed April 19, 2025, https://www.researchgate.net/publication/227730131_Code_is_Speech_Legal_Tinkering_Expertise_and_Protest_Among_Free_and_Open_Source_Software_Developers
89. Crypto's First Amendment Hustle - Yale Journal of Law & Technology, accessed April 19, 2025, https://yjolt.org/sites/default/files/langvardt_26yalejltech130.pdf
90. Hacking Speech: Informational Speech and the First Amendment - Scholarly Commons: Northwestern Pritzker School of Law, accessed April 19, 2025, https://scholarlycommons.law.northwestern.edu/cgi/viewcontent.cgi?article=1069&context=nulr
91. Text, Speech, Machine: Metaphors for Computer Code in the Law, accessed April 19, 2025, http://computationalculture.net/text-speech-machine-metaphors-for-computer-code-in-the-law/
92. Can I legally decompile my own binary code created by copyrighted compiler?, accessed April 19, 2025, https://law.stackexchange.com/questions/3542/can-i-legally-decompile-my-own-binary-code-created-by-copyrighted-compiler
93. Technological Protection Measures and Copyright Exceptions in EU27: Towards the Harmonization - DePaul College of Law, accessed April 19, 2025, https://law.depaul.edu/academics/centers-institutes-initiatives/center-for-intellectual-property-law-and-information-technology/programs/Documents/ipsc_2007/paper/Marcella_FavalePaper.pdf
94. How to license and protect your software? - PACE Anti-Piracy, accessed April 19, 2025, https://paceap.com/how-to-protect-your-software/
95. If I obfuscate someone else's source code, can I avoid copyright infringement?, accessed April 19, 2025, https://law.stackexchange.com/questions/24088/if-i-obfuscate-someone-elses-source-code-can-i-avoid-copyright-infringement
96. Recent Criminal Misappropriation of Trade Secrets to China | Insights - Holland & Knight, accessed April 19, 2025, https://www.hklaw.com/en/insights/publications/2019/07/recent-criminal-misappropriation-of-trade-secrets-to-china
97. Trade Secrets and Algorithms as Barriers to Social Justice - Center for Democracy & Technology (CDT), accessed April 19, 2025, https://cdt.org/wp-content/uploads/2017/08/2017-07-31-Trade-Secret-Algorithm

s-as-Barriers-to-Social-Justice.pdf

98. Apple v. Rivos: Lessons for Companies Facing Claims of Trade Secret Theft (US), accessed April 19, 2025, https://www.employmentlawworldview.com/apple-v-rivos-lessons-for-companies-facing-claims-of-trade-secret-theft-us/
99. Non-U.S. Companies and the DTSA: Parameters of a Developing Reality, accessed April 19, 2025, https://www.tradesecretslawblog.com/2020/06/non-us-companies-dtsa-misappropriation/
100. Justice Department Announces Trade Secret Theft and Other Charges Following Recently Launched Technology Strike Force, accessed April 19, 2025, https://www.tradesecretsandemployeemobility.com/justice-department-announces-trade-secret-theft-and-other-charges-following-recently-launched-technology-strike-force
101. Circumventing technological protection measures and website blocking orders: An EU perspective, accessed April 19, 2025, https://www.twobirds.com/en/insights/2020/uk/circumventing-technological-protection-measures-and-website-blocking-orders-an-eu-perspective
102. Unscrewing the Future: The Right to Repair and the Circumvention of Software TPMs in the EU, accessed April 19, 2025, https://www.jipitec.eu/jipitec/article/download/270/265/1277
103. TPM circumvention and website blocking orders: An EU perspective - The IPKat, accessed April 19, 2025, https://ipkitten.blogspot.com/2020/11/tpm-circumvention-and-website-blocking.html