

Notes

REASONABLENESS MEETS REQUIREMENTS: REGULATING SECURITY AND PRIVACY IN SOFTWARE

PAUL N. OTTO[†]

ABSTRACT

Software security and privacy issues regularly grab headlines amid fears of identity theft, data breaches, and threats to security. Policymakers have responded with a variety of approaches to combat such risk. Suggested measures include promulgation of strict rules, enactment of open-ended standards, and, at times, abstention in favor of allowing market forces to intervene. This Note lays out the basis for understanding how both policymakers and engineers should proceed in an increasingly software-dependent society. After explaining what distinguishes software-based systems from other objects of regulation, this Note argues that policymakers should pursue standards-based approaches to regulating software security and privacy. Although engineers may be more comfortable dealing with strict rules, this Note explains why both policymakers and engineers benefit from pursuing standards over rules. The nature of software development prevents engineers from ever guaranteeing security and privacy, but with an effective regulatory standards framework complemented by engineers' technical expertise, heightened security, and privacy protections can benefit society.

INTRODUCTION

On October 20, 2008, Anne Pressly, a television anchorwoman in Little Rock, Arkansas, was discovered in her home after having been attacked and severely beaten.¹ Although she spent the next week at a

Copyright © 2009 by Paul N. Otto.

[†] Duke University School of Law, J.D. expected 2010; North Carolina State University Department of Computer Science, Ph.D. expected 2010; North Carolina State University Department of Computer Science, M.S. 2007; University of Virginia, B.S. 2004. I would like to

hospital, Ms. Pressly never regained consciousness and ultimately passed away on October 25.² The attack quickly gained national media attention,³ especially because the beating was particularly savage and yet apparently random.⁴

In addition to attracting national media attention, the situation surrounding Ms. Pressly's attack and subsequent hospitalization also inspired curiosity among hospital employees within the St. Vincent Health System. Within a month of Ms. Pressly's death, the hospital announced the firing of several employees for "improperly accessing [her] medical records."⁵

In cases the media follows, there have been many breaches of patients' privacy rights through unauthorized access to medical records.⁶ The suspension or termination of hospital employees often

thank Professor Annie I. Antón for her research guidance and mentoring throughout my graduate studies; Professor Jennifer Jenkins for advising me on this Note; Professor Jeremy Mullem for providing invaluable writing assistance in the early stages of this Note; and Aaron Massey for reviewing early drafts from the perspective of a software engineer. For helpful comments and suggestions, I also thank the participants in Duke Law School's Student Paper Series and the entire staff of the *Duke Law Journal*. Special thanks to Joyce, my wonderful wife, for her love and support.

1. *Arkansas TV Anchor Dies Days After Attack*, CNN.COM, Oct. 26, 2008, <http://www.cnn.com/2008/CRIME/10/25/tv.anchor.attack/>.

2. Jacob Quinn Sanders, *Hospital Fires Up to 6 for Accessing Pressly's Files*, ARK. DEMOCRAT-GAZETTE (NW. ARK. ED.), Nov. 20, 2008, available at <http://www.nwanews.com/adg/News/244079/print/>.

3. *E.g.*, *Arkansas TV Anchor Dies Days After Attack*, *supra* note 1.

4. *See* Steve Barnes, *Robbery Suspected as Motive in Beating Death of Anchor*, N.Y. TIMES, Nov. 29, 2008, at A17 (indicating that robbery appeared to be the motive for the attack and that "the police had no reason to think Ms. Pressly had been singled out before the evening her assailant entered her . . . home").

5. Sanders, *supra* note 2. The hospital indicated that between two and six employees accessed Ms. Pressly's medical records without valid reasons. *Id.* The hospital detected the illicit access to Ms. Pressly's records because the medical records "were being audited every day . . . and as soon as [the hospital] learned of a possible breach, [it] investigated." *Id.* The employees' actions likely constituted violations of federal health privacy law. *Id.*

6. *See, e.g.*, Jack Brill, Note, *Giving HIPAA Enforcement Room to Grow: Why There Should Not (Yet) Be a Private Cause of Action*, 83 NOTRE DAME L. REV. 2105, 2105 (2008) (describing the suspension of twenty-seven hospital employees in 2007 for snooping on the medical records of George Clooney and his girlfriend); Charles Ornstein, *Hospital to Punish Snooping on Spears*, L.A. TIMES, Mar. 15, 2008, at A1, available at <http://articles.latimes.com/2008/mar/15/local/me-britney15> (detailing how the "UCLA Medical Center [was] taking steps to fire at least 13 employees and has suspended at least six others for snooping in the confidential medical records of pop star Britney Spears" in 2008, even though the medical center had circulated a memo upon Spears's hospitalization reminding employees of patients' privacy rights and indicating that unauthorized access would lead to disciplinary action); Sanders, *supra* note 2 (noting how a hospital fired employees in 2006 for unauthorized access to the records of Dick Cheney's hunting friend after he was accidentally shot by Dick Cheney).

follows such breaches, owing to the strict privacy protections put in place by the Health Insurance Portability and Accountability Act of 1996 (HIPAA)⁷ and its resulting regulations regarding the security and privacy of medical records.⁸ As a news article describing the Pressly situation mentions, however, “you still have to wonder . . . why is there not more limited access to those [medical] records—especially with a prominent individual when you could really expect an unauthorized person would get overly curious? Why does the hospital allow any employee access to records they do not need to see?”⁹

The problem of unauthorized access to private information is not limited to the healthcare domain, nor is the general problem restricted to unauthorized access. Personally identifiable information¹⁰—whether financial, medical, or otherwise private—is threatened by identity theft,¹¹ data breaches,¹² and fraud,¹³ among

7. Health Insurance Portability and Accountability Act of 1996, Pub. L. No. 104-191, 110 Stat. 1936 (codified as amended in scattered sections of 18 U.S.C., 26 U.S.C., 29 U.S.C., and 42 U.S.C.).

8. See *infra* notes 79–95 and accompanying text.

9. Sanders, *supra* note 2. The quote is from an expert in health privacy law. *Id.* Note that this concern reflects a lack of proper access control mechanisms, which are discussed further in Part II.B.2.

10. Personally identifiable information (PII) is “information traceable to the individual and that person’s behavior.” RAYMOND T. NIMMER, *LAW OF COMPUTER TECHNOLOGY* § 16.75 (1985); see also NETWORK ADVERTISING INITIATIVE, 2008 NAI PRINCIPLES 5 (2008), available at http://www.networkadvertising.org/networks/2008%20NAI%20Principles_final%20for%20Web%20site.pdf (“PII includes name, address, telephone number, email address, financial account number, government-issued identifier, and any other data used or intended to be used to identify, contact or precisely locate a person.”). Sector-specific laws and regulations often define PII for their sector. See, e.g., 16 C.F.R. § 313.3(o) (2009) (defining “personally identifiable financial information”); 45 C.F.R. § 160.103 (2008) (defining “individually identifiable health information”).

11. See SYNOVATE, FEDERAL TRADE COMMISSION—2006 IDENTITY THEFT SURVEY REPORT 4 (2007), available at <http://www.ftc.gov/os/2007/11/SynovateFinalReportIDTheft2006.pdf> (“[Survey results] suggest[] that approximately 8.3 million U.S. adults discovered that they were victims of some form of ID theft in 2005.”).

12. See Paul N. Otto, Annie I. Antón & David L. Baumer, *The ChoicePoint Dilemma: How Data Brokers Should Handle the Privacy of Personal Information*, IEEE SECURITY & PRIVACY, Sept.–Oct. 2007, at 15 (discussing the increase in data breach disclosures starting with data broker ChoicePoint’s breach, which became public in February 2005); Privacy Rights Clearinghouse, *A Chronology of Data Breaches*, <http://privacyrights.org/ar/ChronDataBreaches.htm> (last visited May 12, 2009) (documenting 1220 breaches resulting in over 261,759,380 “records containing sensitive personal information involved in security breaches in the U.S. since January 2005”—the actual count is likely higher because many breaches involve an unknown number of exposed records).

other threats. Misuse of personally identifiable information has increased as more information enters electronic form, thus facilitating both its exchange and exposure on a larger scale. The transition to electronic record systems has necessitated the development of complex software systems¹⁴ to manage the creation, storage, and transmission of electronic information.

Increasingly, laws and regulations specify how software systems must implement data security and privacy measures. Some legal requirements regarding security and privacy emerge in advance of software system development to control the direction of software use.¹⁵ Other security and privacy requirements emerge in response to perceived excesses or threats from existing software systems.¹⁶ In both scenarios, policymakers¹⁷ must make decisions about the means through which they seek to control software design, development, and deployment.¹⁸ In particular, policymakers must decide which approach—rules, standards, or nonintervention—is most appropriate to protect security and privacy within software.¹⁹

Once laws and regulations take effect, the policymakers' task may appear complete, as auditors and regulators take over the job of overseeing compliance with and enforcement of these security and privacy requirements. The software engineering community,²⁰

13. One major area of fraud involves unauthorized credit card transactions. See Edward A. Morse & Vasant Raval, *PCI DSS: Payment Card Industry Data Security Standards in Context*, 24 *COMPUTER L. & SECURITY REP.* 540, 543–44 (2008), available at <http://ssrn.com/abstract=1303122> (discussing the problem of unauthorized charges and how the costs are distributed within the industry).

14. For the purposes of this Note, a software system is any system in which software plays a primary role (for example, an electronic medical records system for managing patient records both within and across hospitals). By contrast, a non-software-centric system primarily would utilize nonsoftware means to accomplish its purposes (for example, a paper-based medical records system). This definition reflects the fact that software permeates almost every aspect of modern-day life and attempts to distinguish as unique regulatory subjects systems in which software is the primary actor.

15. HIPAA is one example. See *infra* Part II.B.1.

16. The Gramm-Leach-Bliley Act, Pub. L. No. 106-102, 113 Stat. 1338 (1999) (codified in scattered sections of 12 U.S.C. and 15 U.S.C.), is one such example. See *infra* Part II.B.2.

17. Throughout this Note, the term “policymakers” is used to refer equally to decisionmakers in Congress or administrative agencies who are considering whether to regulate software.

18. See *infra* Part II. The differences between the design, development, and deployment stages of software are discussed in Part I.

19. See *infra* Part II.

20. Software engineering is defined as “[t]he application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the

however, has struggled to manage the implementation of these legal requirements within the software development process.²¹ The software engineering community has recognized compliance monitoring as a significant problem,²² but researchers still struggle to develop methodologies for even establishing, much less monitoring, compliance.²³ Extracting security and privacy requirements directly from legal texts has proven too difficult and error-prone to address the need for compliance.²⁴ Furthermore, the ambiguity inherent in legal texts raises numerous problems for engineers seeking to implement legal requirements directly into software systems.²⁵

The disconnect between legal requirements and engineering realities raises serious concerns about the efficacy of emerging data security and privacy protections. Both engineers and policymakers recognize compliance as essential to protecting security and privacy,²⁶

application of engineering to software.” IEEE COMPUTER SOC’Y, IEEE STANDARD GLOSSARY OF SOFTWARE ENGINEERING TERMINOLOGY, IEEE STANDARD 610.12, at 67 (1990). The software engineering community thus reflects the body of practitioners and researchers in that field.

21. For an explanation of what the software development process entails, see *infra* note 35.

22. Compliance monitoring is a general problem facing software engineers that extends beyond the context of requirements originating in laws and regulations. See, e.g., William N. Robinson, *Implementing Rule-Based Monitors Within a Framework for Continuous Requirements Monitoring*, 38 HAW. INT’L CONF. ON SYS. SCI. 188a, 188a (2005) (“Monitoring information systems for requirements compliance is an important and growing problem.”). Several new academic and practitioner conferences and workshops have emerged with a focus on compliance with legal requirements. E.g., Workshop, *Requirements Engineering and Law*, 16 IEEE INT’L REQUIREMENTS ENGINEERING CONF. (2008).

23. See Robinson, *supra* note 22, at 188a (“[R]esearch has found that systems are often misaligned with their policies, there are no systematic design methodologies for requirements monitoring systems, and there is limited support for real-time requirements monitoring.”).

24. See Ambrosio Toval, Alfonso Olmos & Mario Piattini, *Legal Requirements Reuse: A Critical Success Factor for Requirements Quality and Personal Data Protection*, 10 IEEE JOINT INT’L CONF. ON REQUIREMENTS ENGINEERING 95, 96 (2002) (discussing the lack of efforts to extract requirements directly from legal texts); Paul N. Otto & Annie I. Antón, *Addressing Legal Requirements in Requirements Engineering*, 15 IEEE INT’L REQUIREMENTS ENGINEERING CONF. 5, 11 (2007) (discussing the failures of Toval et al.’s approach to extracting requirements directly from legal texts).

25. See Otto & Antón, *supra* note 24, at 7 (cataloging efforts within the software engineering community to categorize ambiguities).

26. For a discussion of the importance placed on compliance by the engineering community, see *supra* note 22. Regulators have also stressed compliance through the imposition of mandatory audits as part of remedies for privacy and security breaches. See, e.g., Providence Health & Services, Complaint Nos. 06-47465 & 06-52268 (Dep’t of Health & Human Servs. July 15, 2008) (resolution agreement), <http://www.hhs.gov/ocr/privacy/hipaa/enforcement/examples/agreement.pdf> (mandating a “corrective action plan,” which includes a yearly review of all security policies by the Department of Health & Human Services (HHS), quarterly onsite audits by HHS, and yearly compliance reports submitted to HHS); *In re* The TJX Cos., Inc., No. 072-

yet the technical means to establish and maintain compliance are lagging behind the legal mandates.²⁷ The result is a situation in which Ms. Pressly and her family ostensibly have the protection of the law safeguarding Ms. Pressly's medical records, yet the software systems managing her records are ill-equipped to provide the protection without the possibility of unauthorized access.

This Note seeks to explore the relationship between law and software with regard to security and privacy. Specifically, this Note argues that legal requirements governing security and privacy must take the form of broad standards rather than specific rules.²⁸ Although software engineers may prefer the ease of implementing rules—with their specific technological mandates—to ambiguous and open-ended standards, security and privacy interests are best protected through standards that leave room for evolution. Broad standards allow the law to capture moving targets; by requiring reasonable software security, for example, the law can continue to mandate strong security measures as industry best practices evolve

3055 (Fed. Trade Comm'n July 29, 2008) (decision and order), *available at* <http://www.ftc.gov/os/caselist/0723055/080801tjxdo.pdf> (including requirements for TJX to implement a “comprehensive information security program” and to undergo biennial third-party audits for the next twenty years); *United States v. ChoicePoint, Inc.*, No. 1:06-CV-0198 (N.D. Ga. Feb. 15, 2006) (stipulated final judgment), *available at* <http://www.ftc.gov/os/caselist/choicepoint/0523069stip.pdf> (imposing the same requirements on ChoicePoint as those listed above for TJX).

27. Implementation difficulties have caused the healthcare industry to lag behind in meeting certain compliance requirements. For specific details, see Travis D. Breaux & Annie I. Antón, *Towards Regulatory Compliance: Extracting Rights and Obligations to Align Requirements with Regulations*, 14 IEEE INT'L REQUIREMENTS ENGINEERING CONF. 49, 49 (2006).

28. Cf. Edward Lee, *Rules and Standards for Cyberspace*, 77 NOTRE DAME L. REV. 1275, 1277–78 (2002) (arguing that courts should favor standards over rules in evaluating cases involving cyberspace). This Note adopts the general terminology for differentiating rules from standards: “the only distinction between rules and standards is the extent to which efforts to give content to the law are undertaken before or after individuals act.” Louis Kaplow, *Rules Versus Standards: An Economic Analysis*, 42 DUKE L.J. 557, 560 (1992) (emphasis omitted). A rule is an *ex ante* statement of the law, *id.* at 559, such as a requirement to use a specific encryption algorithm in a software system. With a standard, by contrast, the law's meaning is determined *ex post*, *id.*, as would occur with a legal requirement to use reasonable encryption in a software system. See Ehud Guttel & Alon Harel, *Uncertainty Revisited: Legal Prediction and Legal Postdiction*, 107 MICH. L. REV. 467, 480 (2008) (“Rules and standards may both generate uncertainty. Standards are legal norms whose interpretation is provided only *ex post* by the courts. Standards, therefore, produce future uncertainty resulting from the indeterminacy of the interpretation given to them *ex post* by the courts. Rules are concrete norms that leave no (or little) discretion to decision makers.”). For a broader discussion of the rules-versus-standards debate, see generally Kaplow, *supra*. For further examples from security and privacy laws and regulations, see *infra* Part II.

and improve. Similarly, by including broad privacy protections, the law can capture new threats to privacy as understandings evolve regarding what constitutes personal information or whether previously innocuous information is in fact personally identifiable.²⁹

Part I of this Note discusses the various aspects of software that make it unique as compared with other regulatory subjects. Part II discusses the spectrum of choices that policymakers face in regulating software. Part III presents the principal argument of this Note: with regard to security and privacy protections, standards are more appropriate than rules for requirements regarding software systems that are made at the original policymaking level. This Note concludes with some thoughts on how policymakers' use of standards in the first instance is most effective when engineers create rules to meet the standards given to them.

I. THE NATURE OF SOFTWARE

There are several characteristics of software systems that make them unique subjects of regulation. These characteristics center

29. There are several examples of how information can go from seemingly innocuous to personally identifiable. One example is census data. See Philippe Golle, *Revisiting the Uniqueness of Simple Demographics in the US Population*, 5 ACM WORKSHOP ON PRIVACY IN THE ELECTRONIC SOC'Y 77, 77 (2006) ("A famous study of the 1990 census data showed that 87% . . . of the population in the United States reported characteristics [in the 1990 census] that likely made them unique based only on gender, 5-digit ZIP code and full date of birth. The study further reported that 53% of the U.S. population is uniquely identified only by {gender, place, date of birth}, where 'place' is basically the city, town, or municipality in which the person resides. Even at the county level, {gender, county, date of birth} uniquely identifies 18% of the U.S. population." (citation omitted)). See generally Paul Ohm, *Broken Promises of Privacy: Responding to the Surprising Failure of Anonymization* (Univ. of Colo. Law Sch., Legal Studies Research Paper No. 09-12, 2009), available at <http://ssrn.com/abstract=1450006> (discussing the impact of "reidentification science"—the ability to reidentify previously anonymized data—on privacy protections). Another area in which there has been a change in perception of what constitutes proper privacy protections involves the movement to make public records available online. See Daniel J. Solove, *Access and Aggregation: Public Records, Privacy and the Constitution*, 86 MINN. L. REV. 1137, 1142 (2002) ("From the beginning of the twentieth century, we have witnessed a vast proliferation in the number of government records kept about individuals as well as a significant increase in public access to these records. These trends together have created a problematic state of affairs—a system where the government extracts personal information from the populace and places it in the public domain, where it is hoarded by private sector corporations that assemble dossiers on almost every American citizen."); Privacy Rights Clearinghouse, *Public Records on the Internet: The Privacy Dilemma*, <http://www.privacyrights.org/ar/onlinepubrecs.htm> (last visited May 12, 2009) ("One of the most challenging public policy issues of our time is the balancing act between access to public records and personal privacy - the difficulty of accommodating both personal privacy interests and the public interest of transparent government.").

around the nature of software design, development, and deployment as compared with non-software-centric systems. To illustrate these unique characteristics, this Part will compare and contrast software with physical buildings at three key stages: design, development, and deployment.³⁰ Understanding these characteristics is essential to making an informed decision regarding what form of regulation is appropriate for safeguarding security and privacy in software systems.

A. Design

The crucial differentiating factor of software-based systems is that software is virtually unlimited in flexibility and scope.³¹ Unlike the construction of a building, in which the laws of physics constrain the possibilities lying before the architect, software can be molded into any shape necessary to perform a desired set of functions.³² From the perspective of the software engineering community, this flexibility has been described in lofty terms: “The programmer, like the poet, works only slightly removed from pure thought-stuff. He builds his castles in the air, from air, creating by exertion of the imagination.”³³ Software’s malleable nature allows it to encompass far greater complexity than physical counterparts can manage.³⁴ This same flexibility, however, comes at a great cost: the overall software development process³⁵ is prone to scheduling delays and constant flux,

30. The similarities and differences noted in this Part are by no means an exhaustive list; this Note highlights only those similarities and differences that may influence the decision of which legal regime is appropriate in regulating software in a given instance, as discussed in Part III. For an explanation of the software development process, see *infra* note 35.

31. This characteristic also has been referred to as software being “plastic.” James Grimmelmann, Note, *Regulation by Software*, 114 YALE L.J. 1719, 1723 (2005).

32. The practicing programmer may take issue with this characterization of software’s flexibility, noting the numerous constraints placed on software design by choice of programming language, operating system, and so on. Although these are limits on flexibility, they are more properly characterized as limitations that emerge during development—unless the initial requirements specify those constraints directly. It is therefore important to separate the initial design phases, in which decisions regarding scope and direction of the project are made, from the development phase, discussed *infra* Part I.B. This view considers software as the product of a software engineering process, which is more appropriate for the types of systems typically contemplated by laws and regulations.

33. FREDERICK P. BROOKS, JR., *THE MYTHICAL MAN-MONTH: ESSAYS ON SOFTWARE ENGINEERING* 7 (Anniversary ed. 1995).

34. See Grimmelmann, *supra* note 31, at 1731–32, 1734 (“Software can successfully apply rules whose complexity would make them collapse under their own weight if humans were forced to apply them.”).

35. This is a term of art in the software engineering community, in which the word “development” encompasses more than simply the development phase as the word is used in

as stakeholders³⁶ treat flexibility as a license to change requirements at any time.³⁷

B. Development

The need to define system functionality³⁸ is the greatest limitation on the scope of software (that is, the tasks that the software system will perform and the features it will contain). Once decisions on the software's scope and function are made, the "creativity of the programmer, the complexity or sophistication of the software itself, or the environment in which it operates" serve to limit the software's theoretical flexibility.³⁹ Thus, the narrow selection of permitted inputs

Part I.B. The classic formulation of the software development process includes the following key stages: requirements, design, implementation, testing, and deployment. IEEE COMPUTER SOC'Y, *supra* note 20, at 67. Depending on the model in use, the development process may involve a linear progression through each stage or a highly iterative process by which software is designed, developed, and deployed. *Id.* The software lifecycle, by contrast, is defined as all of the phases in the development process as well as the subsequent operation and maintenance of the deployed system. *Id.* at 68.

36. In software engineering, the term "stakeholders" refers to the "individuals or organisations who stand to gain or lose from the success or failure of a system . . . includ[ing] customers or clients (who pay for the system), developers (who design, construct, and maintain the system), and users (who interact with the system to get their work done)." Bashar Nuseibeh & Steve Easterbrook, *Requirements Engineering: A Roadmap*, 22 INT'L CONF. ON SOFTWARE ENGINEERING: FUTURE OF SOFTWARE ENGINEERING TRACK 35, 37 (2000). An alternative definition of the term "stakeholders" refers to any entity with a stake in the outcome of a given software system. See Travis D. Breaux & Annie I. Antón, *Analyzing Regulatory Rules for Privacy and Security Requirements*, 34 IEEE TRANSACTIONS ON SOFTWARE ENGINEERING 5, 8 (2008) (including all entities mentioned in a source document as stakeholders).

37. See, e.g., Annie I. Antón & Colin Potts, *Functional Paleontology: System Evolution As the User Sees It*, 29 IEEE TRANSACTIONS ON SOFTWARE ENGINEERING 151, 151 (2003) ("Requirements volatility (customer-desired, short-term functional change) has been identified as a principal obstacle to software development."); Nuseibeh & Easterbrook, *supra* note 36, at 39 ("[I]t is usually the case that requirements change during development and evolve after a system has been in operation for some time.").

38. Within the software engineering community, the task of defining system functionality is known as requirements engineering or "the process of discovering [software's intended] purpose, by identifying stakeholders and their needs, and documenting these in a form that is amenable to analysis, communication, and subsequent implementation." Nuseibeh & Easterbrook, *supra* note 36, at 35. Requirements engineering plays an instrumental role in the earliest phases of software development. See JOHN BERGEY ET AL., RESULTS OF SEI INDEPENDENT RESEARCH AND DEVELOPMENT PROJECTS AND REPORT ON EMERGING TECHNOLOGIES AND TECHNOLOGY TRENDS 21-22 (2004), <http://www.sei.cmu.edu/pub/documents/04.reports/pdf/04tr018.pdf> ("It is well recognized in the industry that requirements engineering is critical to the success of any major development project . . ." (citations omitted)); IEEE COMPUTER SOC'Y, *supra* note 20, at 62-63 (defining the requirements phase of software development and its associated tasks).

39. R. Polk Wagner, *On Software Regulation*, 78 S. CAL. L. REV. 457, 479 (2005).

and outputs⁴⁰ for a specific program necessarily limits the virtually infinite reach of software that exists in theory.

Despite the emergence of significant limits on flexibility and creativity imposed during the development phase, these limits are even more constraining in the development of a physical building. First, there are physical limits to the size and scope of a given building; in software, however, it is relatively easy to add new functionality or address evolving requirements even during development.⁴¹ Second, constructing a physical building proceeds in clearly defined stages, and generally architects and builders try to avoid having to undo any portions that have been situated. In software, however, there is comparatively lower cost in changing many elements of the software, no matter how much has been built.⁴²

C. Deployment

After development or construction has concluded, the first major contrast between software systems and physical buildings concerns the ease of replication. Once developed, a software system can be deployed in a virtually unlimited number of locations with minimal additional effort; replication of physical structures can be accomplished through reuse of the original design, but development must begin anew at each new building site.

40. Software engineers often define the scope of software in terms of the inputs that the system permits and the outputs that the system creates. The process of defining the inputs and outputs most often occurs after the requirements specification is complete.

41. See, e.g., Barry W. Boehm & Phillip N. Papaccio, *Understanding and Controlling Software Costs*, 14 IEEE TRANSACTIONS ON SOFTWARE ENGINEERING 1462, 1466 (1988) (“[T]he cost of fixing or reworking software is much smaller (by factors of 50 to 200) in the earlier phases of the software life cycle than in the later phases.” (citations omitted)). Such changes are not always cheap, however, as discussed *infra* note 42.

42. This is not to say that the cost of making such changes during development is low. Empirical studies of software engineering, for example, have demonstrated that the cost of correcting an error increases substantially at each phase of the software lifecycle. See, e.g., BERGEY ET AL., *supra* note 38, at 21–22 (“As compared with defects found during requirements evaluations, defects cost 10–200 times as much to correct once fielded [and] 10 times as much to correct during testing. . . . A recent study by IBM’s System Sciences Institute found that the relative cost of fixing software defects after deployment is almost 15 times greater than detecting and eliminating them in development.”); Mark Curphey & Rudolph Araujo, *Web Application Security Assessment Tools*, IEEE SECURITY & PRIVACY, July–Aug. 2006, at 35, 35 (“In 1996, Capers Jones showed that, if the unit cost of finding a bug during development were US\$1, failing to find the same bug until deployment would cost \$16,000 . . .”). The key difference is that there are physical, temporal, and pecuniary limitations in the construction of a physical building, whereas in software the temporal and pecuniary limitations would be the only controlling factors.

Further, once its deployment is complete, software is “automated” in its function: no further intervention is required for software to make its determinations of allowable and prohibited behaviors.⁴³ This behavior may not at first glance seem so different from a physical building: once conceived, constructed, and completed, a building is available for full use. The key difference in this regard between software systems and other regulated entities, such as a building, is the degree of interaction that software possesses. Software may take actions without any human oversight.⁴⁴ This automated nature can be a great strength: software can just as easily handle one case as it can handle one billion cases,⁴⁵ meaning that software systems can manage a larger number of transactions than any human-driven system could hope to address. A significant difference exists in scale of usage as well: whereas most physical buildings have a relatively low occupancy limit, software may simultaneously accommodate many orders of magnitude more users.

Along with its automated nature, software also produces immediate results. Software therefore can bar prohibited behavior without ever allowing a violation to occur in the first place;⁴⁶ similarly, software can permit actions to take place immediately, again without requiring human oversight. In contrast, a physical building can take no action to control its usage: once built, the building’s structure may have implications for the building’s usage, but not enforcement of the developer’s intent. Legal requirements implemented in software provide an immediate interpretation of the requirements as they are

43. Grimmelmann, *supra* note 31, at 1723. Professor Wagner refers to this characteristic as software being “preprogrammed.” Wagner, *supra* note 39, at 478.

44. The default behavior of software is that “[t]he programmed algorithm is followed without deviation.” Wagner, *supra* note 39, at 478. It is possible for systems to design oversight into the process, as for example by requiring a supervisor to approve an action before the software will proceed. As Professor Wagner notes, however, “software-implemented regulations are freestanding mechanisms and do not generally require recourse to other institutional players for . . . rule determinations.” *Id.* at 479 (footnote omitted). Human intervention most often is included when software systems are required to provide audit capabilities or supplement existing, non-software-based processes.

45. *See* Grimmelmann, *supra* note 31, at 1729 (“Once a piece of software has been written, the marginal cost of running it to handle another case can be vanishingly small.”).

46. *Id.* at 1723 (“Rather than relying on sanctions imposed after the fact to enforce its rules, [software] simply prevents the forbidden behavior from occurring.”). Note that some forms of software, such as distributed software, may impose legal restraints retroactively. *Cf.* JONATHAN ZITTRAIN, THE FUTURE OF THE INTERNET—AND HOW TO STOP IT 63, 102–06 (2008) (discussing how tethered devices, through the use of distributed software, can limit “generativity” by controlling what actions are permitted and prohibited).

represented in the software system—even if the result was not fully understood or even contemplated by the original software developers.⁴⁷

The immediate results provided by software systems directly contrast with another characteristic of these systems: the lack of transparency in software's decisionmaking process. Although an individual interacting with a software system will see immediate results from an attempt to engage in a particular action, software does not provide any explanation of its decisions unless an explanation has been included *ex ante* within the software. Furthermore, unless the software is updated, software is locked into providing the results programmed in during software development.⁴⁸ This differs sharply from interactions with a physical building, in which many (if not all) design decisions are exposed directly to the building's occupants.⁴⁹

The final significant difference between software and its physical counterparts is that software is commonly deployed despite the existence of known problems in the software. There is an understanding in the software engineering community that large software systems cannot be constructed perfectly (that is, without a single bug or vulnerability).⁵⁰ This tolerance for bugs and system failures contrasts sharply with expectations for physical buildings, in which compliance with building plans and specifications is generally fairly precise and accurate. This difference in expectations leads to a great variance in maintenance costs after deployment: both buildings

47. See Danielle Keats Citron, *Technological Due Process*, 85 WASH. U. L. REV. 1249, 1254 (2008) ("Although programmers building automated systems may not intend to engage in rulemaking, they in fact do so. Programmers routinely change the substance of rules when translating them from human language to computer code." (footnote omitted)); Grimmelman, *supra* note 31, at 1730 ("Software cannot—as law can—adapt its response in light of later-available information or a later determination that such information is relevant."). Note, however, that software systems can be updated to reflect new legal understandings, just as software developers patch systems to resolve other types of errors.

48. See *supra* note 47.

49. Note that such transparency in physical buildings does not mean that the intent behind such decisions is evident, simply that the decisions themselves are viewable.

50. The software engineering community has standardized precise language to distinguish different types of problems. There are three deviations from normal operation of a software system: a failure occurs when the system behavior detectably deviates from expected or correct behavior; an error is the deviation in system behavior that led to the failure; and a fault is the underlying cause of the error. Algirdas Avizienis et al., *Basic Concepts and Taxonomy of Dependable and Secure Computing*, 1 IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING 11, 13 (2004). Bugs and vulnerabilities simply describe different types of unknown faults. *Id.* at 17.

and software have an expected maintenance phase after deployment, but for software, the cost of testing and maintaining the final product may exceed the costs of design and development.⁵¹

II. THE INTERSECTION OF SOFTWARE AND REGULATION

With the increased use of computing technologies in modern society, there has been a corresponding—though lagging—increase in laws and regulations targeting actions that software systems can and cannot perform. Although these laws and regulations target various types and aspects of software systems,⁵² this Note will focus on laws and regulations concerning data security and privacy protections.

The idea of regulating software through laws has been called a form of legal preemption.⁵³ The spectrum of choices facing policymakers who regulate the behavior of software systems includes specifying software functionality directly through the promulgation of specific rules, defining broad standards with which software must comply, and enabling software to take the place of explicit regulation.⁵⁴ In addition, policymakers may choose to not regulate software systems at all, leaving security and privacy protections to be handled by market forces.⁵⁵ Each of these options is discussed in turn.

51. See, e.g., Barry Boehm & Victor R. Basili, *Software Defect Reduction Top 10 List*, 34 IEEE COMPUTER 135, 137 (2001) (noting that “low-dependability software costs about 50 percent per instruction more to maintain than to develop, whereas high-dependability software costs about 15 percent less to maintain than to develop”).

52. Professors Kesan and Shah provide a list of several areas in which government shapes software design, development, and deployment. See Jay P. Kesan & Rajiv C. Shah, *Shaping Code*, 18 HARV. J.L. & TECH. 319, 322–23 (2005) (“In addition to [spam and cell phone number portability], the government is also involved in shaping the development of code for reasons involving antitrust, national security, protection of intellectual property rights, accessibility, safety, and content labeling.” (footnotes omitted)). Other areas in which the government recently has sought to regulate software include electronic voting machines and critical infrastructure.

53. Wagner, *supra* note 39, at 485.

54. The various approaches to regulating software, however, “are neither exhaustive nor mutually exclusive.” *Id.* at 487.

55. The Clinton Administration famously adopted such a market-based approach. See, e.g., Peter P. Swire, *Markets, Self-Regulation, and Government Enforcement in the Protection of Personal Information*, in NAT’L TELECOMMS. AND INFO. ADMIN., U.S. DEP’T OF COMMERCE, PRIVACY AND SELF-REGULATION IN THE INFORMATION AGE, ch. 1.A (1997), available at http://www.ntia.doc.gov/reports/privacy/privacy_rpt.htm (advocating industry self-regulation to fill the gaps between “pure market” and “pure [government] enforcement” approaches). This approach drew significant criticism. See, e.g., Neil Weinstock Netanel, *Cyberspace Self-Governance: A Skeptical View from Liberal Democratic Theory*, 88 CAL. L. REV. 395, 476 (2000) (“Far from its promise of Pareto optimality, the proffered combination of self-regulation and

A. Preemption through Legal Rules

In the strongest form of legal preemption, laws or regulations “directly establish[] formal boundaries or requirements for software code.”⁵⁶ Such a rules-based approach by policymakers would include instances in which laws or regulations specify that a particular software feature is prohibited or another software feature is required. This direct legal preemption has been increasingly common in recent laws and regulations.

The clearest example of a rules-based approach governing software systems is the Audio Home Recording Act (AHRA).⁵⁷ The statute imposes a direct requirement on system development: “[n]o person shall import, manufacture, or distribute any digital audio recording device or digital audio interface device that does not conform to—(1) the Serial Copy Management System.”⁵⁸ The Serial Copy Management System thus serves as a specific functional requirement for software system design and development.⁵⁹

Two state laws mandating data encryption, both scheduled to take effect in January 2010, reflect a growing trend among states to regulate security and privacy in an effort to slow the flood of data breaches occurring nationwide.⁶⁰ Nevada enacted a new statute in 2009 requiring all businesses in Nevada to not “[t]ransfer any personal information through an electronic, nonvoice transmission other than a facsimile to a person outside of the secure system of the data collector unless the data collector uses encryption to ensure the

market forces would likely fail adequately to protect data privacy. Industry self-regulation, a group’s regulation of its members’ practices with the goal of reducing harmful externalities to outsiders, is notoriously inadequate to its task. As trenchant critics have shown, such self-regulation can only work under conditions of stringent government oversight.”)

56. Wagner, *supra* note 39, at 485. Professor Wagner refers to this method of regulating software as “direct” legal preemption. *Id.* at 487.

57. Audio Home Recording Act of 1992, Pub. L. No. 102-563, 106 Stat. 4237 (codified as amended at 17 U.S.C. §§ 1001–10 (2006)).

58. 17 U.S.C. § 1002(a). The statute goes on to provide two alternatives to implementing the Serial Copy Management System: implementing a system with the “same functional characteristics,” *id.* § 1002(a)(2), or a system that has been “certified by the Secretary of Commerce” as accomplishing the same objectives, *id.* § 1002(a)(3).

59. For a brief description of the Serial Copy Management System, see Aaron L. Melville, Note, *The Future of the Audio Home Recording Act of 1992: Has It Survived the Millennium Bug?*, 7 B.U. J. SCI. & TECH. L. 372, 380–81 (2001). For a more detailed discussion, see Nika Aldrich, *An Exploration of Rights Management Technologies Used in the Music Industry*, 2007 B.C. INTELL. PROP. & TECH. F. 051001 (2007), http://bciprf.org/index.php?option=com_content&task=view&id=30&Itemid=30.

60. For a discussion of the data breach problem, see *supra* note 12.

security of electronic transmission.”⁶¹ Nevada law defines encryption broadly, rather than specify a particular form of encryption.⁶²

Massachusetts enacted a more comprehensive set of regulations targeting information security. The new regulations require “encryption of all transmitted records and files containing personal information that will travel across public networks, and encryption of all data containing personal information to be transmitted wirelessly,” as well as “[e]ncryption of all personal information stored on laptops or other portable devices.”⁶³ In addition, the regulation outlines several software security requirements for all systems involving personal information.⁶⁴ Like Nevada’s law, the Massachusetts regulations define encryption broadly rather than mandate use of a particular algorithm.⁶⁵

Another form of rules-based regulation of software involves policymakers setting rules for the transactions in which software is involved.⁶⁶ This would include situations in which the law requires software to be used in a particular way to facilitate transactions, thus using software as a means to satisfy some other transactional legal requirement. An early effort to prescribe specific rules for software development involved the encryption used for communications. The federal government proposed a rule known as the Escrowed Encryption Standard,⁶⁷ which focused on providing an encrypted telecommunications in a manner that still allowed law enforcement

61. NEV. REV. STAT. § 603A (2009) (effective 2010); *see also* S.B. 227, 75th Leg., Reg. Sess. § 3 (Nev. 2009), 2009 Nev. Stat. 1603, 1604 (noting the new statute “becomes effective on January 1, 2010”).

62. *Id.* (referencing compliance with generally accepted encryption standards).

63. 17 MASS. CODE REGS. 17.04 (2008); *see also id.* 17.05 (“Every person who owns, licenses, stores or maintains personal information about a resident of the Commonwealth shall be in full compliance with 201 CMR 17.00 on or before January 1, 2010.”).

64. *Id.* In addition to encryption, the regulations detail requirements related to “user authentication protocols,” “access control measures,” firewalls, and malware and antivirus software. *Id.*

65. *See id.* 17.02 (defining “[e]ncrypted” as “the transformation of data through the use of an algorithmic process, or an alternative method at least as secure, into a form in which meaning cannot be assigned without the use of a confidential process or key, unless further defined by regulation by the Office of Consumer Affairs and Business Regulation”).

66. *See* Wagner, *supra* note 39, at 486 (“[Another] form of legal preemption would be legal rules that specify the use of particular software in a transactional rather than regulatory manner.”).

67. Despite the word “Standard” appearing in its name, this proposal qualifies as a type of rule when applying the nomenclature used in this Note. The federal government’s use of “rule” and “standard” in naming various proposals often does not track this Note’s terminology. These variances are noted in footnotes as each law or regulation is introduced throughout this Note.

access when authorized.⁶⁸ This rule imposed detailed requirements on the technical implementation of encryption.⁶⁹ The Department of Homeland Security regulations implementing the REAL ID Act,⁷⁰ which require compliance with a particular ISO Standard,⁷¹ offer another example of transactional rules.

B. Preemption through Legal Standards

The second major form of legal preemption is when “legal regulations establish the framework within which software will operate.”⁷² This would include laws or regulations governing a general area in which software systems ultimately will be responsible for satisfying particular requirements. Regulatory standardization is weaker than rules-based legal preemption in terms of its impact on how software is designed, developed, and deployed: rules provide clear requirements and constraints, whereas standards may provide only limited or generalized guidance for software engineering.⁷³ A glimpse at a few instances of modern, standards-based approaches highlights the great amount of flexibility that standards allow software engineers. This Section details two recent laws—with accompanying regulations—that provide examples of the standards-based approach to protecting data security and privacy in software systems.⁷⁴

68. NAT'L INST. OF STANDARDS AND TECH., U.S. COMMERCE DEP'T TECH. ADMIN., FEDERAL INFORMATION PROCESSING STANDARDS PUB. 185, ESCROWED ENCRYPTION STANDARD (1994), available at <http://www.itl.nist.gov/fipspubs/fip185.htm>.

69. See *id.* (specifying the required functions and parameters for the encryption algorithm). For a broader discussion of the implications underlying the Escrowed Encryption Standard, see generally A. Michael Froomkin, *The Metaphor Is the Key: Cryptography, the Clipper Chip, and the Constitution*, 143 U. PA. L. REV. 709 (1995). For a discussion of how the Escrowed Encryption Standard fits in with the idea of software-as-regulator, see Lee Tien, *Architectural Regulation and the Evolution of Social Norms*, 7 YALE J.L. & TECH. 1, 18–20 (2004).

70. REAL ID Act of 2005, Pub. L. No. 109-13, 119 Stat. 302 (codified in scattered sections of 8 U.S.C.).

71. 6 C.F.R. § 37.19 (2009) (“For the machine readable portion of the REAL ID driver’s license or identification card, States must use the ISO/IEC 15438:2006(E) Information Technology—Automatic identification and data capture techniques—PDF417 symbology specification.”).

72. Wagner, *supra* note 39, at 485. Professor Wagner refers to this as “regulatory standardization.” *Id.*

73. For a more in-depth discussion of this crucial difference, see *infra* notes 156–162 and accompanying text.

74. There are several examples of other recent laws and regulations that adopt a standards-based approach to regulating security and privacy in software. See, e.g., CAL. CIV. CODE § 1798.81.5(b) (West 2006) (“A business that owns or licenses personal information about

1. *HIPAA*. With the passage of HIPAA,⁷⁵ Congress set in motion the development of specific security and privacy guidelines for the healthcare domain through standards-based regulation. Congress passed HIPAA with the expectation that patient health records would transition from paper-based systems to electronic health record systems.⁷⁶ HIPAA required the creation of regulations governing privacy⁷⁷ and security⁷⁸ for electronic health records. The Department of Health and Human Services (HHS) responded with the enactment of the Privacy Rule⁷⁹ and Security Standards,⁸⁰ respectively.⁸¹

The Privacy Rule begins by laying out a key guiding principle: the standard restricts uses or disclosures of “protected health information”⁸² to what is expressly and explicitly authorized by the Privacy Rule.⁸³ The Privacy Rule also recognizes certain health care activities as generally permitted uses and disclosures of protected health information: “treatment, payment, or health care operations.”⁸⁴

a California resident shall implement and maintain reasonable security procedures and practices appropriate to the nature of the information, to protect the personal information from unauthorized access, destruction, use, modification, or disclosure.”). This Note focuses on the two laws—with their accompanying regulations—due to the breadth and depth of their coverage of security and privacy issues in software systems.

75. Pub. L. No. 104-191, 110 Stat. 1936 (codified as amended in scattered sections of 18 U.S.C., 26 U.S.C., 29 U.S.C., and 42 U.S.C.).

76. In fact, several HIPAA regulatory sections were written for specific aspects of the transition to electronic health records. *See, e.g.*, Health Insurance Reform: Modifications to Electronic Data Transaction Standards and Code Sets, 68 Fed. Reg. 8381 (Feb. 20, 2003) (codified at 45 C.F.R. pt. 162).

77. HIPAA § 264, 110 Stat. at 2033 (codified at 42 U.S.C. § 1320d-2 note).

78. HIPAA § 1173, 110 Stat. at 2024–26 (codified at 42 U.S.C. § 1320d-2).

79. Standards for Privacy of Individually Identifiable Health Information, 67 Fed. Reg. 53,181 (Aug. 14, 2002) (codified at 45 C.F.R. pts. 160 & 164). For an excellent account of the passage of the Privacy Rule, as well as criticisms of its approach to privacy protection, see generally Meredith Kapushion, Note, *Hungry, Hungry HIPAA: When Privacy Regulations Go Too Far*, 31 FORDHAM URB. L.J. 1483 (2004).

80. Health Insurance Reform: Security Standards, 68 Fed. Reg. 8333 (Feb. 20, 2003) (codified at 45 C.F.R. pts. 160, 162 & 164).

81. Despite the names, both the Privacy Rule and Security Standards operate as standards under this Note’s nomenclature. *See supra* note 67.

82. 45 C.F.R. § 160.103 (2008) (defining “protected health information” generally as “individually identifiable health information,” with limited exclusions for information contained in certain types of education and employment records).

83. *See id.* § 164.502(a) (“A covered entity may not use or disclose protected health information, except as permitted or required by this subpart or by subpart C of part 160 of this subchapter.”).

84. *Id.* § 164.502(a)(1)(ii). The extent of the use or disclosure of protected health information is limited by section 164.506(c).

The Privacy Rule goes on to require authorizations for many other types of uses or disclosures.⁸⁵ When protected health information is anonymized, the restrictions on use or disclosure of such information are lessened.⁸⁶ A significant requirement within the Privacy Rule is that regulated health organizations must restrict access to protected health information to the “minimum necessary” use or disclosure in a given situation.⁸⁷ The Privacy Rule then specifies requirements for informing individuals of their privacy rights.⁸⁸

The Privacy Rule includes a general requirement that covered entities have “*appropriate* administrative, technical, and physical safeguards to protect the privacy of protected health information.”⁸⁹ These measures “*must reasonably* safeguard protected health information from any intentional or unintentional use or disclosure that is in violation of the standards, implementation specifications or other requirements of [the Privacy Rule],”⁹⁰ as well as “limit incidental uses or disclosures made pursuant to an otherwise permitted or required use or disclosure.”⁹¹

The other standards enacted in response to HIPAA are the Security Standards, which explicitly focus on security for “electronic

85. The Privacy Rule lays out the requirement for authorization before the use or disclosure of psychotherapy notes, *id.* § 164.508(a)(2), and for uses or disclosures related to marketing, *id.* § 164.508(a)(3); authorization may be waived for use or disclosure with respect to research studies if an oversight board approves the waiver, *id.* § 164.512(i).

86. *See id.* § 164.514(a)–(b) (defining what constitutes de-identification).

87. *See id.* § 164.514(d)(1)–(5) (elaborating the “minimum necessary requirements” for uses, disclosures, requests for information, and other requirements).

88. Covered entities generally are required to provide individuals with notice of their organizations’ privacy practices with respect to the “uses and disclosures of protected health information.” *Id.* § 164.520(a). The Privacy Rule goes on to provide detailed requirements for the content of such privacy notices, *id.* § 164.520(b), as well as the method of delivery, *id.* § 164.520(c). Individuals have the right to access most of their protected health information. *See id.* § 164.524(a) (setting forth the “right of access” as well as defining when covered entities may deny individuals the right). Individuals also have a matching right to amend their protected health information under section 164.526(a)(1), again subject to limitations, *id.* § 164.526(a)(2). Significantly, individuals also have a general “right to receive an accounting of disclosures of protected health information made by a covered entity.” *Id.* § 164.528(a)(1). The standards for the “content of the accounting” are laid out in section 164.528(b).

89. *Id.* § 164.530(c)(1) (emphasis added). Each of the three types of safeguards is defined within the Security Standards portion of the HIPAA regulations, *id.* § 164.304; technical safeguards are defined as “the technology and the policy and procedures for its use that protect electronic protected health information and control access to it,” *id.*

90. *Id.* § 164.530(c)(2)(i) (emphasis added).

91. *Id.* § 164.530(c)(2)(ii).

protected health information.”⁹² The Security Standards begin by laying out four general security requirements for covered entities:

- (1) Ensure the confidentiality, integrity, and availability of all electronic protected health information the covered entity creates, receives, maintains, or transmits.
- (2) Protect against any *reasonably* anticipated threats or hazards to the security or integrity of such information.
- (3) Protect against any *reasonably* anticipated uses or disclosures of such information that are not permitted or required
- (4) Ensure compliance with this subpart by its workforce.⁹³

Notably, a covered entity is free to “use any security measures that allow the covered entity to *reasonably* and *appropriately* implement” the general rules.⁹⁴ The Security Standards dedicate an entire section to detailing standards for technical safeguards.⁹⁵

2. *GLB Act*. Congress laid out a set of standards for software in the financial sector with its passage of the Gramm-Leach-Bliley Act (GLB Act).⁹⁶ Congress passed the GLB Act to meet the modern needs of financial institutions by enabling increased competition and the combination of diverse financial services within a single institution.⁹⁷

92. *Id.* § 164.302.

93. *Id.* § 164.306(a) (emphases added).

94. *Id.* § 164.306(b)(1) (emphases added). The Security Standards go on to list four factors that a covered entity must weigh in determining which security measures to employ. *Id.* § 164.306(b)(2).

95. *Id.* § 164.312. Each standard may be accompanied by some implementation specifications, which are marked as either required or addressable. *Id.* § 164.306(d)(1). The addressable label indicates that covered entities must weigh the appropriateness of the implementation specification, *id.* § 164.306(d)(3)(i), and either implement the requirement, *id.* § 164.306(d)(3)(ii)(A), or both “[d]ocument why it would not be reasonable and appropriate to implement the implementation specification,” *id.* § 164.306(d)(3)(ii)(B)(1), and consider implementation of a reasonable alternative, *id.* § 164.306(d)(3)(ii)(B)(2). The technical safeguards include the following requirements: access control—restricting access to only authorized individuals, *id.* § 164.312(a)(1); audit controls—maintaining records of all activity within the system, *id.* § 164.312(b); integrity—guarding against “improper alteration or destruction” of data, *id.* § 164.312(c); authentication—verifying the identity of those seeking access to data, *id.* § 164.312(d); and transmission security—protecting information in transit over a network, *id.* § 164.312(e).

96. Gramm-Leach-Bliley Act, Pub. L. No. 106-102, 113 Stat. 1338 (1999) (codified in scattered sections of 12 U.S.C. and 15 U.S.C.).

97. *See id.* pmb1, 113 Stat. at 1338 (noting the purpose of the GLB Act is “[t]o enhance competition in the financial services industry by providing a prudential framework for the

The GLB Act contains numerous provisions regarding the security and privacy of “nonpublic personal information.”⁹⁸ The Act specifies that “each financial institution has an affirmative and continuing obligation to respect the privacy of its customers and to protect the security and confidentiality of those customers’ nonpublic personal information.”⁹⁹ The Act tasks regulators such as the Federal Trade Commission (FTC) with creating “appropriate standards . . . (1) to insure the security and confidentiality of customer records and information; (2) to protect against any anticipated threats or hazards to the security or integrity of such records; and (3) to protect against unauthorized access to or use of such records or information.”¹⁰⁰

The FTC partly satisfied its requirements under the GLB Act by enacting the Safeguards Rule.¹⁰¹ The Safeguards Rule begins by broadly requiring that “[all regulated financial institutions] shall develop, implement, and maintain a comprehensive information security program.”¹⁰² Specifically regarding software systems, regulated financial institutions must “assess the sufficiency of any safeguards in place to control these risks . . . [as relevant to] [i]nformation systems, including network and software design, as well as information processing, storage, transmission and disposal.”¹⁰³ Financial institutions must react to this risk assessment by “design[ing] and implement[ing] information safeguards to control the risks” and additionally must “test or otherwise monitor the effectiveness of the safeguards’ key controls, systems, and procedures.”¹⁰⁴ In addition, financial institutions are required to “[e]valuate and adjust [their] information security program in light of the results of the testing and monitoring.”¹⁰⁵

affiliation of banks, securities firms, insurance companies, and other financial service providers, and for other purposes”).

98. 15 U.S.C. § 6801 (2006).

99. *Id.* § 6801(a).

100. *Id.* § 6801(b).

101. Standards for Safeguarding Customer Information, 67 Fed. Reg. 36,484 (May 23, 2002) (codified at 16 C.F.R. pt. 314). This is another instance of a mismatch in nomenclature, here created by nongovernmental actors referring to this standard as a “Rule.” *See supra* note 67.

102. 16 C.F.R. § 314.3(a) (2009).

103. *Id.* § 314.4(b).

104. *Id.* § 314.4(c).

105. *Id.* § 314.4(e).

To finish satisfying its requirements under the GLB Act, the FTC enacted the Privacy Rule.¹⁰⁶ The GLB Privacy Rule (as distinguished from the HIPAA Privacy Rule discussed in Part II.B.1) largely focuses on issues relating to privacy notices and opt-out procedures.¹⁰⁷ The GLB Privacy Rule, however, still contains some potential restrictions on software, although they are presented indirectly. For example, the standards restrict the situations in which regulated financial institutions may disclose “nonpublic personal information about a consumer.”¹⁰⁸ These restrictions may be implemented and managed by software systems, which can check whether exceptions have been satisfied before sharing personal information with a third party. The standards also contain restrictions on “redisclosure and reuse of information,”¹⁰⁹ which may require software systems to maintain records regarding the disclosure status of customer data.

C. *Software-as-Regulator*

The third form of legal preemption occurs when a policymaker chooses to support “software-as-regulator.”¹¹⁰ This form focuses on structuring the law to favor regulation by software, in lieu of specifying legal requirements directly. The fundamental idea is that policymakers allow self-regulation at first, then react to how the market and industry evolve by protecting the innovations that develop. In other words, policymakers validate *ex post* the role that software plays in controlling actions, rather than legislating or regulating *ex ante*.

Although policymakers have not employed the software-as-regulator approach in protecting data security and privacy, they have regulated software systems more generally using this approach. One example is the Digital Millennium Copyright Act,¹¹¹ which broadly prohibits using software to circumvent digital rights management

106. Privacy of Consumer Financial Information, 65 Fed. Reg. 33,646 (May 24, 2000) (codified at 16 C.F.R. pt. 313). Under this Note’s nomenclature, this “Rule” is actually a standard. *See supra* note 67.

107. *See* 16 C.F.R. § 313.4–.9 (2009).

108. *Id.* § 313.10(a)(1).

109. *Id.* § 313.11.

110. Wagner, *supra* note 39, at 486.

111. Digital Millennium Copyright Act, Pub. L. No. 105-304, 112 Stat. 2860 (1998) (codified in scattered sections of 17 U.S.C.).

(DRM) systems.¹¹² By enacting this restriction, policymakers entrusted DRM software systems with enforcing the rights of copyright holders. Another example, the E-Sign Act,¹¹³ took a slightly different approach to the software-as-regulator idea by requiring a technology-neutral stance with respect to the treatment of electronic signatures in commerce.¹¹⁴

With the explosion of software in cyberspace, legal scholars began to note how software was filling a legal function in advance of actual laws and regulations. This idea was best summarized in Professor Lawrence Lessig's pronouncement that "[c]ode is law."¹¹⁵ The theory is that software can fulfill a regulatory function—or at least have the same effects as regulation—through the choices made in its implementation.¹¹⁶ The essential characteristic of software-as-regulator is that "[a] rule is defined, not through a statute, but through the code that governs [a software system]."¹¹⁷

The "code is law" theory has been criticized as a disingenuous representation of the role of software in regulation.¹¹⁸ It is important to understand the differences in the way that software regulates, as

112. 17 U.S.C. § 1201 (2006).

113. Electronic Signatures in Global and National Commerce Act, Pub. L. No. 106-229, 114 Stat. 464 (2000) (codified at 15 U.S.C. §§ 7001–31 (2006)).

114. See 15 U.S.C. § 7002(a)(2)(A)(ii) (2006) (barring states from "accord[ing] greater legal status or effect to, the implementation or application of a specific technology or technical specification for performing the functions of creating, storing, generating, receiving, communicating, or authenticating electronic records or electronic signatures").

115. LAWRENCE LESSIG, CODE: VERSION 2.0, at 5 (2006).

116. See *id.* ("Cyberspace demands a new understanding of how regulation works. It compels us to look beyond the traditional lawyer's scope—beyond laws, or even norms. It requires a broader account of 'regulation,' and most importantly, the recognition of a newly salient regulator. . . . In real space, we recognize how laws regulate—through constitutions, statutes, and other legal codes. In cyberspace we must understand how a different 'code' regulates—how the software and hardware . . . that make cyberspace what it is also regulate cyberspace as it is."). Although Professor Lessig's book focuses on cyberspace, *id.*, the arguments extend equally to software systems more broadly.

117. *Id.* at 24.

118. For one critical analysis of the idea that code is law, see Wagner, *supra* note 39, at 460–61. Another critique focuses on how the "code is law" idea breaks down when "the shifting patterns of legal compliance in the 2000s" are considered. See Tim Wu, *When Code Isn't Law*, 89 VA. L. REV. 679, 681–82 (characterizing code as "an anti-regulatory mechanism" that influences laws, rather than replacing them). Note that such criticisms have emerged despite Professor Lessig's disclaimer that there are important differences between software and law with respect to regulatory effects. See LESSIG, *supra* note 115, at 5 ("I don't deny these differences. I only assert that we learn something useful from ignoring them for a bit.").

compared to traditional regulation by law.¹¹⁹ Of particular importance to this Note, the “code is law” theory has been attacked for its misrepresentation of important privacy considerations.¹²⁰

The “code is law” concept raises interesting questions regarding the role of software as an alternative to regulation. Scholars have attempted to explain when policymakers may favor regulation by software over regulation by law, weighing the impact of each form of regulation as well as each approach’s costs and benefits.¹²¹

D. Nonregulation

Policymakers may choose to not regulate software using any of the preceding three options. In the absence of laws or regulations, there are two possibilities for how the software development process still may be subject to guidelines. The first possibility occurs when industry standards emerge regarding elements of software design, development, and deployment;¹²² the second approach leaves it to the market to set minimum standards.

A recent example of industry attempting to self-regulate software systems is the payment card industry’s effort to develop the

119. See Wagner, *supra* note 39, at 460–61 (exploring “the basic truth of the regulatory effects of both software and legal code, yet rejecting their equivalence”).

120. See Marc Rotenberg, *Fair Information Practices and the Architecture of Privacy (What Larry Doesn’t Get)*, 2001 STAN. TECH. L. REV. 1, ¶¶ 37–38, 68–69 (2001) (noting how Professor Lessig’s analysis misrepresented privacy law and failed to account for various emerging statutory privacy protections).

121. See Kesan & Shah, *supra* note 52, at 321 (“Policymakers, however, have had to rely on their own insights and experiences when developing code-based solutions, as no comprehensive analysis is available to help guide the government in regulating, shaping, and reshaping the architecture of information technology.”); *id.* at 326–27 (detailing five ways in which policymakers can influence software development: prohibition, setting standards, market-based regulation, modifying liability, and disclosure).

122. For a discussion of how industry standards emerge, see generally Mark A. Lemley & David McGowan, *Legal Implications of Network Economic Effects*, 86 CAL. L. REV. 479, 496–97 (1998). In many markets, there is a “natural tendency toward de facto standardization, which means everyone using the same system.” *Id.* at 496 (quoting Michael L. Katz & Carl Shapiro, *Systems Competition and Network Effects*, 8 J. ECON. PERSP. 93, 105 (1994)).

Some state laws include explicit reference to industry standards as part of their security and privacy requirements. See, e.g., 17 MASS. CODE REGS. 17.03 (2008) (requiring a company’s “comprehensive information security program [to] be reasonably consistent with industry standards”); NEV. REV. STAT. § 603A (effective 2010) (“If a data collector doing business in this State accepts a payment card in connection with a sale of goods or services, the data collector shall comply with the current version of the Payment Card Industry (PCI) Data Security Standard, as adopted by the PCI Security Standards Council or its successor organization, with respect to those transactions . . .”).

Data Security Standard (DSS).¹²³ DSS seeks to enhance the security of credit card transactions by establishing general standards for securing the software systems managing such transactions.¹²⁴ It includes a set of twelve high-level requirements,¹²⁵ each with detailed lower-level requirements and matching testing procedures for participating entities to evaluate their compliance with each element of the Standard.¹²⁶

DSS includes a mixture of specific rules and broad standards for the payment card industry to consider. Some requirements provide very high-level guidance; for example, Requirement 3.1 instructs entities to minimize data storage and develop appropriate data retention policies.¹²⁷ Other requirements are worded in standards-like language, but provide guidance bordering on specific rules. For example, Requirement 4.1 instructs entities to employ strong security protocols, but gives two examples of such protocols and provides specific testing procedures for use of one of these protocols.¹²⁸ Finally, some requirements are intended to dictate specific rules for how entities manage transactions; for example, Requirement 1 provides strict rules on how network security must be handled.¹²⁹ As a whole, DSS seeks to provide a comprehensive framework to protect data security in payment card transactions.¹³⁰

123. PCI SEC. STANDARDS COUNCIL, PAYMENT CARD INDUSTRY DATA SECURITY STANDARD: REQUIREMENTS AND SECURITY ASSESSMENT PROCEDURES VERSION 1.2 (2008), https://www.pcisecuritystandards.org/security_standards/download.html?id=pci_dss_v1-2.pdf.

124. *Id.* at 3.

125. *Id.*

126. The in-depth elaboration of the high-level requirements into detailed requirements and testing procedures begins on page thirteen of the document. This elaboration might be construed as a set of rules under this Note's nomenclature, despite the use of the word "Standard" in naming this document, as discussed *infra* notes 127–129 and accompanying text. *See supra* note 67.

127. PCI SEC. STANDARDS COUNCIL, *supra* note 123, at 20.

128. *Id.* at 26.

129. *See id.* at 15–16 (providing strict rules for restricting Internet access, such as requiring entities to "[i]mplement stateful inspection . . . [and] IP masquerading").

130. For a more thorough analysis of the Data Security Standard and its likely impact within the payment card industry, see Morse & Raval, *supra* note 13, at 550–53. Recent criticism has called the Data Security Standard a failure, *see, e.g.*, Andrew Conry-Murray, *PCI and Schrodinger's Cat*, INFORMATIONWEEK, Feb. 25, 2009, http://www.informationweek.com/blog/main/archives/2009/02/pci_and_schrodi.html (noting that compliance monitoring occurs only once a year, because more frequent monitoring "would be obscenely expensive," and arguing that "the only value of PCI is to the card brands, which can use it as a shield against federal regulation"), which elicited a reply from a PCI member, *see* Adrian Phillips, *Feedback: In Defense of the PCI Data Security Standard*, INFORMATIONWEEK, Mar. 14, 2009,

The second nonregulatory option occurs when the industry does not set standards, but instead the market informally sets the minimum requirements for the software development process.¹³¹ For example, consumers may demand that a certain level of security and privacy protections be included in the software they purchase or services with which they interact.¹³² In the absence of clear regulatory guidance regarding software design, development, and deployment, the resulting constraints on the software development process will be whatever the market will bear. The government may even choose to use its weight in the market as a large buyer of goods and services in order to reach the desired level of security and privacy protection.¹³³

One example of this market reaction concerns general privacy protections in the business environment. The United States lacks any comprehensive data privacy law, but instead has adopted targeted laws in specific sectors.¹³⁴ As a result, companies have been slow to

<http://www.informationweek.com/news/security/attacks/showArticle.jhtml?articleID=215802153> (“[T]he PCI DSS has proven to be a highly effective foundation of minimum security standards when properly implemented across all systems handling cardholder data. In fact, no compromised entity to date has been found to be in compliance with PCI DSS at the time of the breach.”).

131. This theory drove the market-based approach employed by the Clinton Administration. *See supra* note 55; *see also* Netanel, *supra* note 55, at 475–76 (discussing how the Clinton Administration believed “the virtual ‘invisible hand’ will generate a set of data protection alternatives, ranging from no protection to significant protection,” from which consumers could freely select based on their level of concern); *id.* at 476 (“If enough consumers are sufficiently concerned about data privacy to refuse to visit nonprotective sites, the Administration believes, market pressure will push sites to provide protection.” (citing Swire, *supra* note 55)).

132. Such market-driven activity can be the product of network effects, which refers to the impact on markets “in cases in which ‘the utility that a user derives from consumption of a good increases with the number of other agents consuming the good.’” Lemley & McGowan, *supra* note 122, at 483 (quoting Michael L. Katz & Carl Shapiro, *Network Externalities, Competition, and Compatibility*, 75 AM. ECON. REV. 424, 424 (1985)). Consumers may develop expectations for certain security and privacy protections given their experiences with other software systems.

133. For example, the Obama Administration’s May 2009 report on cybersecurity strategy recommended leveraging the government’s purchasing power in order to improve security in software. *See* WHITE HOUSE, CYBERSPACE POLICY REVIEW: ASSURING A TRUSTED AND RESILIENT INFORMATION AND COMMUNICATIONS INFRASTRUCTURE 34 (2009), available at http://www.whitehouse.gov/assets/documents/Cyberspace_Policy_Review_final.pdf (recommending the administration “[d]efine procurement strategies through the General Services Administration, building on work by the National Security Agency for the Department of Defense, for commercial products and services in order to create market incentives for security to be part of hardware and software product designs, new security technologies, and secure managed services”).

134. The GLB Act, which governs financial privacy, and HIPAA, which governs the privacy of healthcare information, are two such laws. *See supra* Part II.B.

adopt privacy protections, as evidenced by the slew of data breaches made public since 2005.¹³⁵ The lack of specific guidance has prompted some consideration of an industry standard along the lines of the payment card industry's response to the need for heightened security protections in the absence of specific regulatory requirements.¹³⁶

III. IN SUPPORT OF STANDARDS OVER RULES FOR SOFTWARE SYSTEMS

As discussed in Part II, policymakers have a wide range of options for ensuring that software systems protect data security and privacy. They may choose not to regulate software at all,¹³⁷ instead trusting the market to protect security and privacy adequately. Similarly, policymakers may pass laws and regulations that enable software systems to fulfill a regulatory function, in line with the "code is law" theory.¹³⁸ But neither of these choices is satisfactory in light of how, in the absence of laws and regulations, the market generally has failed to protect the security and privacy of personally identifiable information.¹³⁹ The numerous problems in protecting data security

135. Privacy Rights Clearinghouse, *supra* note 12. Many of these instances would not have been made public but for the passage of data breach notification laws, starting with California's passage of such a law in 2003. For example, evidence suggests that ChoicePoint made its breach public only because of the California law. Otto et al., *supra* note 12, at 16–17.

There is a larger debate as to whether the market's slow adoption of privacy protections simply mirrors consumers' consistent undervaluing of privacy. Alternatively, privacy may be a special type of good requiring particular protection, as once lost it is generally impossible to restore; for such an explanation, see generally Alessandro Acquisti & Jens Grossklags, *What Can Behavioral Economics Teach Us About Privacy*, in *DIGITAL PRIVACY: THEORY, TECHNOLOGIES AND PRACTICES* 363 (Alexandro Acquisti et al. eds., 2007).

136. Multinational corporations that face a patchwork of privacy requirements are spearheading this market response. See Miriam Wugmeister, Karin Retzer & Cynthia Rich, *Global Solution for Cross-Border Data Transfers: Making the Case for Corporate Privacy Rules*, 38 *GEO. J. INT'L L.* 449, 450 (2007) (describing the need for "Corporate Privacy Rules" to manage the patchwork of global privacy laws, wherein "businesses would establish their own set of rules for the transmission of personal information via the Internet . . . [which] would incorporate internationally accepted principles of fair information practices"). The challenge facing Corporate Privacy Rules in the absence of regulatory guidance is how to make these rules enforceable. *Id.*

137. See *supra* Part II.D.

138. See *supra* Part II.C.

139. See *supra* Part II.D. One problem is the difficulty in enforcing whatever self-regulation or market solution emerges in the absence of legal preemption. See, e.g., Wugmeister et al., *supra* note 136, at 488 (noting that "significant concerns remain about how to make Corporate Privacy Rules 'binding' when businesses volunteer to adhere to a set of rules"). Another problem is that the market consistently ignores privacy concerns as expensive externalities. See Bruce Schneier, *The "Hidden Cost" of Privacy*, SCHNEIER ON SECURITY, June 15, 2009,

and privacy demonstrate the need for legal requirements governing software systems; the question then becomes what approach is best suited to providing the necessary protections.

The numerous laws and regulations targeting data security and privacy evidence a strong desire and intent on the part of policymakers to regulate software systems. Part II described two specific approaches to regulating software systems: direct preemption through specific rules¹⁴⁰ and broader regulatory standards.¹⁴¹ Of these forms of legal preemption for software systems, this Note now explains how a focus on standards—achieved through regulatory standardization—offers the best approach for policymakers to protect data security and privacy.

A. *Why Standards Make Sense for Policymakers*

There are many reasons why standards are preferable to rules when it comes to regulating software systems to protect security and privacy. A major advantage of standards over rules relates to the institutional competence (or lack thereof) of policymakers (and courts) to regulate software.¹⁴² As policymakers contemplate increasingly detailed and complex technical requirements for software systems, they are more likely to exceed their knowledge of both software generally and technical feasibility specifically. Although policymakers can consult with technical experts in crafting specific rules for software systems, policymakers' competence aligns

http://www.schneier.com/blog/archives/2009/06/the_hidden_cost.html (“The meta-problem is simple to describe: those entrusted with our privacy often don't have much incentive to respect it. . . . What this all means is that protecting individual privacy remains an externality for many companies, and that basic market dynamics won't work to solve the problem.”). During the Clinton administration, advocates for a market-based approach acknowledged shortcomings in what markets provide as compared to results obtained through government action. *See, e.g.*, Swire, *supra* note 55 (“[T]here are significant reasons to believe that government regulation will be stricter in enforcing the protection of personal information than this sort of self-regulation. The difficult question will be to balance these gains in privacy protection against the likely higher administrative and compliance costs of government regulation.”). For more debate regarding the efficacy of market solutions, see *supra* note 130.

140. *See supra* Part II.A.

141. *See supra* Part II.B.

142. Wagner, *supra* note 39, at 492; *see also* Lee, *supra* note 28, at 1307–11 (discussing how courts benefit from standards-based approaches due to a narrower focus on the facts of the case at hand).

much better with a focus on clear expressions of their intent to protect security and privacy.¹⁴³

It is much easier for policymakers to enact a standard requiring reasonable safeguards for medical records and then allow regulated entities to determine how best to implement those safeguards¹⁴⁴ than it is to spell out exactly what safeguards are considered reasonable. By leaving technical details for later consideration by technical experts and regulated entities, policymakers can rely on auditors and regulators to monitor for compliance with broad standards.¹⁴⁵

Another primary concern with rules-based approaches involves “the difficulty in directly addressing software in legal regulations.”¹⁴⁶ Attempts to impose strict rules on software may fail to account for the inherent imperfection associated with software development.¹⁴⁷ Because software development never seeks to eliminate all errors in a system, but rather seeks to reach a tolerable amount of faults,¹⁴⁸ strict rules may be impractical unless they make concessions for imperfection. In a worst-case scenario, a set of strict rules may even be impossible to implement in conjunction with functional software because of the nature of software and policymakers’ lack of institutional competence to make specific decisions *ex ante*.

143. Cf. Cass R. Sunstein, *Problems with Rules*, 83 CAL. L. REV. 953, 992 (1995) (noting how cyberspace is an area in which many have argued that policymakers “lack enough information to produce rules that will yield sufficiently accurate results”).

144. This was the approach taken in the HIPAA Security Standards. See *supra* note 94 and accompanying text.

145. The concern with institutional competence comports with Professor Kaplow’s general assessment of the efficiency advantage of standards as cheaper to promulgate than rules. Kaplow, *supra* note 28, at 562. By deferring the difficult task of determining the law’s content, policymakers avoid the high costs of attempting to learn enough to effectively create rules.

146. Wagner, *supra* note 39, at 492.

147. See *supra* note 50 and accompanying text.

148. See *id.* There is also substantial research in software engineering on how to prioritize faults to fix during software testing, given that all faults cannot be eliminated. See, e.g., John D. Musa, *Operational Profiles in Software-Reliability Engineering*, 10 IEEE SOFTWARE 14, 28–31 (1993) (explaining how testing can use operational profiles to identify “the failures that occur most frequently,” taking into account factors such as criticality and relatedness). This holds even for important data security and privacy safeguards. See generally JOHN VIEGA & GARY MCGRAW, BUILDING SECURE SOFTWARE: HOW TO AVOID SECURITY PROBLEMS THE RIGHT WAY (2002) (describing the security gap in modern software development and listing principles to manage security during development). In fact, privacy and security faults are harder to detect than other bugs in software systems because they are not evident until the software system has been tested, whereas other faults generally manifest as errors or failures during development (and therefore are addressed through the development process).

The nature of software¹⁴⁹ makes rules-based approaches unlikely to succeed in protecting security and privacy in the long term, because “software development is a rapidly moving, nearly unpredictable target.”¹⁵⁰ The rapid pace of technological change surrounding software system development heavily favors a standards-based approach that allows software engineers to operate within the framework of standards, instead of being chained to rules that no longer mirror reality.¹⁵¹ Any attempt to codify rules regarding software systems therefore will be thwarted by the pace of technological change and the virtually unlimited flexibility of software systems. Given the rapid obsolescence of software and technology,¹⁵² specifying rules for software systems may lead to costly compliance efforts that provide minimal long-term benefit (because security and privacy are not protected adequately by obsolete technical safeguards). If policymakers instead clearly identify their high-level goals and provide standards for software systems, compliance efforts can evolve with the changes in software and technology.

The risk of rules quickly becoming outdated highlights another significant factor that weighs in favor of standards: the cost incurred in specifying rules. Rules-based approaches incur significant costs in specification as compared to standards-based approaches to regulating software because they must be much more intricately designed.¹⁵³ These specification costs outweigh the potential savings in compliance costs. A general concern for any system is “the increased cost of error inherent in any legal preemption scheme,”¹⁵⁴ but the specification cost is much higher for rules than standards.¹⁵⁵ The increased cost of specifying rules stems from the lack of institutional competence with respect to software systems when the rulemaking is left to policymakers, as mentioned above. When nonexperts craft rules, there is a much higher chance that expensive-to-specify rules will have decreased or no actual utility because they do not apply effectively to software. A standards-based approach to regulating

149. See *supra* Part I.

150. Wagner, *supra* note 39, at 492.

151. Sunstein, *supra* note 143, at 993–94 (“In the face of rapidly changing technology, current rules for regulation . . . will become ill-suited to future markets.”).

152. *Id.*

153. See, e.g., *id.* at 992 (“Production of rules entails high *ex ante* investment of political and informational costs.”).

154. Wagner, *supra* note 39, at 492.

155. See *supra* note 153.

software systems mitigates the specification cost by better matching policymakers' competence and avoiding complex technical implementation details.

Although specification costs are higher for rules, as one would expect, compliance costs are likely to be higher for standards.¹⁵⁶ This is because rules—once specified—provide clearer guidance on the precise legal requirements to be addressed by software systems. It is much easier for software engineers to translate legal rules into software requirements.¹⁵⁷ The heightened cost of compliance with standards can be offset, however, in two ways. First, the cost generally associated with standards-based approaches can be mitigated if regulated entities—with the expertise of software engineers on hand—define rules to elaborate standards.¹⁵⁸ These rules, crafted by technical experts rather than policymakers, can evolve alongside technological advances without requiring a retooling of the underlying legal standards. Second, compliance costs also can be mitigated by the creation and evolution of industry best practices with respect to legal standards for software systems.¹⁵⁹ Best practices can both provide a clear benchmark for compliance and evolve with rapid technological advances. Although the Data Security Standard for the payment card industry emerged in lieu of regulatory guidance,¹⁶⁰ it is an excellent example of how industry can define more specific rules to elaborate legal standards.¹⁶¹ But if policymakers specify rules, the evolution of best practices is preempted and replaced with a need to focus on specific compliance goals—leading to the problems discussed above.

156. This holds true for rules versus standards more generally. *See* Kaplow, *supra* note 28, at 562–63 (“Rules are more costly to promulgate than standards because rules involve advance determinations of the law’s content, whereas standards are more costly for legal advisors to predict or enforcement authorities to apply because they require later determinations of the law’s content.”).

157. *See supra* note 38.

158. This practice is already common in technical fields. For example, technical organizations such as the IEEE, ISO, and ANSI promulgate a variety of standards governing all aspects of engineering.

159. Proposals for industry standards have highlighted lower compliance costs as a major selling point. *See, e.g.*, Wugmeister et al., *supra* note 136, at 449–50 (noting how a patchwork of international privacy laws raises the cost of compliance); *id.* at 478 (touting lower compliance costs as a primary advantage of the proposed Corporate Privacy Rules).

160. *See supra* Part II.D.

161. With such implementation rules, situations like the invasion of Pressly’s medical privacy, *see supra* notes 1–6 and accompanying text, may well have been averted.

Finally, even though software engineers may prefer rules because of the certainty they provide, standards provide sufficient guidance for implementing and complying with legal requirements. As noted above, standards bodies and other technical groups can still address the technical details related to regulatory standardization.¹⁶² Standards, accompanied by clear statements of policymakers' intent, provide sufficient information to lead to the development of industry best practices.

B. How Software Engineers Can Respond to Standards-Based Legal Requirements

The nature of software systems is mirrored in the software development process. The design and development of software is rules-driven, with software engineers seeking to capture system requirements in precise specification documents.¹⁶³ It is therefore relatively straightforward for software engineers to adapt to new legal requirements for software systems when those legal requirements are in the form of clear rules.

Standards-based approaches thus present a significant challenge to the current rules-focused model of software development.¹⁶⁴ Software engineers do not generally possess the competence to identify and interpret legal texts. As standards require more interpretation than rules, software engineers are at a significant early disadvantage.¹⁶⁵ In addition to requiring more interpretation, several other institutional hurdles confront software engineers as they adjust to managing legal standards' interpretation and potential evolution.¹⁶⁶

162. *See supra* note 158 and accompanying text.

163. *See supra* note 38. This description of the software development process follows traditional models of development. Some new models, such as agile development, may be considered more standards-driven, as they eschew formal processes in favor of rapid development strategies.

164. Agile development, *see supra* note 163, would be much more amenable to standards-based approaches, given the natural inclination to standards over rules in that development process. The ensuing discussion instead focuses on more traditional development models.

165. This disadvantage presumably is overcome as standards mature and software engineers interact more with such standards over time; the initial hurdles to adoption are the focus of this discussion.

166. *See* Otto & Antón, *supra* note 24, at 6–7 (recognizing several hurdles: identification of legal requirements; interpretation of legal rules; evolution of law through amendments, revisions, and case law; managing ambiguity; and providing traceability for compliance monitoring).

Although standards will be harder to assimilate into the software development process, there are several reasons why software engineers should embrace standards rather than rules governing software systems. In fact, several of the advantages that policymakers gain in choosing standards over rules translate into reasons for software engineers similarly to favor standards-based approaches.

Instead of making software engineers implement technical details drafted by policymakers, standards-based approaches recognize the institutional competence of software engineers to flesh out technical details regarding security and privacy protections in software systems. Software engineers have various tools available for ensuring data security and privacy; with the guidance of a standards framework and clear statements of policymakers' intent, software engineers can select the most appropriate approach for satisfying the legal requirements. Software engineers are also better positioned to make use of the institutional knowledge of standard-setting organizations and technical groups.

Standards also mirror the general understanding of software development as an imperfect process,¹⁶⁷ with all software systems containing some number of errors. Tolerating imperfection is more compatible with a standards-based approach; for example, HIPAA Security Standards allow regulated entities to employ whatever security measures are reasonable and appropriate, rather than dictating the implementation of specific measures.¹⁶⁸ If the software engineering community at large recognizes imperfection as a necessary evil,¹⁶⁹ then a security measure can be considered reasonable even if it is imperfect, assuming that it is still generally accepted by the industry.

The rapid pace of technological change should also garner support among software engineers for standards rather than rules.¹⁷⁰ In one sense, technological evolution will require software engineers to update systems regardless of whether the law provides rules or standards for software. Standards, however, allow software engineers to abandon obsolete security and privacy safeguards as technology advances without falling out of compliance with strict rules that embody understandings of a bygone era.

167. *See supra* note 50 and accompanying text.

168. *See supra* note 94 and accompanying text.

169. *See supra* note 50 and accompanying text.

170. *See supra* notes 150–154 and accompanying text.

As discussed above, standards-based approaches to protecting security and privacy in software systems are likely to incur higher compliance costs than rules-based approaches.¹⁷¹ The same mitigating factors apply here, however, making standards still more beneficial to the software engineering process. The technical expertise of software engineers (and regulated entities more generally) allows the industry as a whole to respond to legal standards by crafting more detailed technical rules—or in the alternative, to create industry best practices for complying with the legal standards for protecting security and privacy in software systems.¹⁷² The payment card industry's efforts with the Data Security Standard reflect an effective use of industry expertise in meeting a particular need for security and privacy protections.¹⁷³ Technical rules or best practices enable software engineers to bypass the difficult problem of understanding legal standards, as the software engineers can be confident that these rules have been crafted to capture the intent and requirements of the legal standards. In addition, the detailed guidance created by technical experts is more easily updated to match the rapid pace of technological advances.

CONCLUSION

Legal protections for data security and privacy in software systems remain an emerging area of law. When software engineers lack regulatory guidance, the market has proven ineffective in providing adequate protection for data security and privacy. Recent approaches to providing protection through laws and regulations have favored the use of broad standards in lieu of specific rules. Both policymakers and engineers benefit from the choice of broad standards; through these standards, data security and privacy are protected even as technology rapidly evolves and new threats emerge. If specific rules are still required, it is preferable for policymakers to

171. *See supra* notes 156–61 and accompanying text.

172. This model mimics the general reliance on experts in the agency decisionmaking process. *See, e.g.*, Adrian Vermuele, *The Parliament of the Experts*, 58 DUKE L.J. 2231, 2232 (2009) (“In the administrative state, a great deal of agency decisionmaking draws upon the aggregate view of a group of experts, especially when there is an expert ‘consensus.’”); *see also id.* at 2234 (arguing that “when agencies are uncertain of facts, causation, or future consequences of alternative policies, following the consensus or majority view of experts is a perfectly rational decisionmaking strategy”).

173. *See supra* notes 123–30 and accompanying text.

leave that task to entities with the technical expertise to create and maintain such rules.

Rules, when appropriate, are not detrimental in software engineering. Rather, this Note argues that software engineers are the appropriate people to make these rules. If a policymaker makes rules *ex ante*, the rules may be inflexible and ill-informed as to the realities facing software engineers. When a rule is crafted by software engineers or industry experts in response to a standard put forward by policymakers, however, it retains flexibility to be changed easily and more effectively as the rule becomes obsolete. In fact, the rule would have to be altered to stay in line with the controlling standard when it is no longer effective.

In situations like the media attention surrounding Ms. Pressly, the combination of policymaker-initiated standards and engineer-crafted rules is most likely to provide the protection of security and privacy that was lacking in her case. Although HIPAA provided the necessary standards-based framework, the lack of technical rules to implement HIPAA's reasonableness requirements led to the unsatisfactory result of prying eyes trumping personal privacy. As policymakers and engineers learn to better create standards-based frameworks backed by implementation rules, society can benefit from improved security and privacy protections in software systems.