Setting the Stage for a Software Liability Discussion

MICHAEL B. SCHER

Mike Scher is VP and General Counsel with the network security firm Nexum. An attorney and security technologist by trade, and an erstwhile legal anthropologist, his focus is on risk mitigation, from the legal to the social, and the technical to the procedural. Mike has been working where the policy tires meet the implementation pavement since 1993.
mscher@nexuminc.com

Software liability isn't what most people seem to think it is. It varies by jurisdiction, market, and more. The current state of affairs—nasty EULAs on the one hand, and dread of liability for ordinary bugs on the other—is probably less than optimal. A patchwork of state, federal, and "judge-made" law is inconsistent by its very nature, varying in complex ways for each situation. In this article, I try to equip the reader with some key concepts around product liability from the perspective of an attorney and security geek.

Among friends and coworkers (on the systems and security side of the industry), and among clients and partners, there is renewed interest in questions of liability for software in its many forms. A lot of the talk addresses controlling risk from a technical or legal perspective, and some addresses how things "ought to be" from a technical or legal perspective. They are frustrated and looking for change, but don't think new laws or regulations will do anything but make matters worse.

Many of them appear to have assumptions about the forms of liability operating today—assumptions that are at odds with how the various areas of liability in fact operate. Ultimately, it is critical that discussions about law build on an accurate sense of, generally, what the law is and how it operates. Bad policies, like bad arguments, are built on false premises. This article is an effort to help lay the conceptual groundwork for developers and sysadmins to engage effectively in discussions on future policy and law regarding software liability, security, safety, and responsibility. My take is US-centric, but the principles should stand one in good stead broadly, and some references here may help others explore the state of the law outside the US.

As with the 2006 ;login: article on negligence [1], "The content and positions contained in this article should not be taken as legal advice—the discussion is simply far too general and the subject matter too complex to safely use that way." The purpose is to make readers more conversant in the issues to apply that knowledge to policy discussions regarding their own areas of expertise.

Negligence Quick Review

Some seem to think we are headed for a negligence standard when we discuss the possibility of liability for flaws in software. A negligence standard is sort of the US default liability standard for anything not specifically and exclusively legislated, regulated, or otherwise under a different standard through common law.

In short, under a negligence standard, one need do the "reasonable" thing. Applied to software, "reasonable" has a lot to do with what a reasonable end-user should expect—which has a lot to do with industry standard practices, but (see again [1]) sometimes industry standards themselves aren't (legally speaking) "reasonable." Negligence is always "there," but licensing agreements, including shrink-wrap style agreements, can disclaim a lot of that liability, for almost anything short of reckless or willfully harmful conduct. Because shrink-wrap licenses vary in the extent to which they are enforceable (by content and by jurisdiction), only customers in a strong negotiation position relative to the licensor will consistently have the ability to shift risk back onto the licensor/reseller. The rest of us end-users might

just have to take it as given or drive on. Due to the economics of combining contracts with torts litigation, practical liability from provider to end-user can disappear in a puff of EULA.

Warranties

With regard to negligence, the industry handles risk today through EULAs applying to licensed software. EULAs disclaim virtually all errors, as well as many of the warranties stemming from states' common law and statutes governing the sale of goods, such as the broadly adopted Uniform Commercial Code (UCC) Article 2. Most software isn't "sold" as a "good"—rather, it is licensed. Thus, as some scholars and practitioners will rapidly point out, UCC Article 2 doesn't even apply to many software transactions [2]. Readers may recall that the 1990s saw the controversial Uniform Computer Information Transactions Act (UCITA), which started as an outgrowth of UCC Article 2 warranties for goods. It proposed broad warranties for licensed software, but allowed virtually all warranties to be disclaimed. Only two states have adopted UCITA in any form [3, 4]. Some courts and a few states have declared all software to be a good subject to UCC Article 2, and sometimes software is delivered incorporated in a good that is subject to warranties. Still, EULAs and similar agreements disclaim many warranties, opening questions regarding general consumer protection law and contracts of adhesion, the answers to which, of course, vary by jurisdiction; however, warranty actions are not how most serious harms caused by product failure are handled.

Product Liability: Software and "the Market"

We're used to using free and commercial software to perform important functions—yet, when introduced, such software may be rife with functional problems that can corrupt data, cause halting, open a system to compromise, or bring about other significant issues. The discovery of security issues in software is a regular occurrence across most popular packages.

The market accepts file corruption and routine rebooting in early edition software, including some operating systems, suggesting that the marketplace has a period of adoption elasticity in which the benefit of inexpensive adoption outweighs the issues. At some point, in theory, mounting competition and pressure for stability and security influence the package producer. Even as we seem to expect the market to perform that function, the notion that critical-use software could fail as badly as the latest app we dropped on our smartphone is an alarming one—especially since consumer market-pressure correction comes after adoption. Still, we're not crazy for thinking people actively making choices can influence quality. We as a society click "Accept" to low standards for many reasons, some historical, some market structural, and some as part of the cost of doing business and keeping software prices low.

That last sounds like almost any competitive commercial goods-producing sphere: we want prices as low as possible, and are willing to accept some drop in quality in exchange, but we still want those goods to be without significant defect. The discussion around software liability hinges on that point: what form will liability take and where is the line that will permit bountiful software development while steering us away from a caveat emptor marketplace? We've discussed negligence for acts and omissions, and how warranty may apply to goods. The US (and much of the world) handles product liability for harms suffered from "defective goods" differently from other forms of liability, and quite differently from the way we handle most licensed software today.

Strict Liability and Products

Ordinary negligence can be a case-by-case, time-consuming, and not-always-predictable process, to say the least. Modern product liability ultimately posits we shouldn't have court cases looking at the micro-facts of each $100 buyer's case, that a buyer should be able to have a base-level confidence that products released into the marketplace are without "defect" to the extent of the product's "intended use." Due diligence and proximate causation are two key issues in negligence—did defendant's behavior fall below a reasonable standard and, if so, did that cause a foreseeable harm in a manner to which liability attaches?

In product liability, the causation question is often simpler. The complicated question is whether the root problem is a "defect." Product liability is inherently a strict liability regime, not a "due diligence" one. Once there is a harm, and a defect leading to the harm is identified, the product maker (and others in the chain of sale) are generally held liable. The definition of "defect" itself subsumes many issues similar to negligence. Because the defect affects many in similar fashion, and because each affected individual's contributory negligence need not be weighed on a case-by-case basis, product liability cases are generally brought as class actions (thus avoiding spending courts' time for each $100 case, permitting class-wide disposition of the matter, and allowing the company and those affected to move on).

Let's take a quick look at how the Restatement of the Law Third, Torts: Products Liability talks about the key term "defect" in goods. There are three forms of defect, broadly defined [5].

First, manufacturing defects, "when the product departs from its intended design, even if all possible care was exercised." Note that negligence isn't the issue with this form of defect; it focuses on the market and the good, not the maker's degree of care. It is possible to have a product be defective and its maker liable for harm even if all reasonable care was exercised. As a matter of public policy, one could say the sale of such a good is inherently unreasonable, but again, the negligence standard simply does not apply [6].

Second are design defects, "when the foreseeable risks of harm posed by the product could have been reduced or avoided by the adoption of a reasonable alternative design, and failure to use the alternative design renders the product not reasonably safe." Here the focus is on both the maker and the market. Liability for failing to use an alternative design hinges to a degree on the reasonable nature of the alternative design, and in that aspect is reminiscent of negligence questions, only to the extent of examining the availability and viability of alternatives.

Third are inadequate instructions or warnings defects, "when the foreseeable risks of harm posed by the product could have been reduced or avoided by reasonable instructions or warnings, and their omission renders the product not reasonably safe." We've all seen what we consider ridiculous instructions (e.g., "do not eat" on silica packets). Here the focus is on the maker interacting with the intended market—which market, from the news, will seem to many readers to have an ever-decreasing mentality. Many of the cases making news as if of the third type are actually of the second. The press tends to repeat these PR pitches uncritically. What, the press carry water for a PR firm? How unreasonable!

Software Liability Generally, Today

Depending on deal size, at the corporation level, an end-user company can push to have UCC Article 2 warranties explicitly apply, and go well beyond that, assuming the software company is eager for the business. That's a contractual engagement where sophisticated parties each with some degree of negotiating power negotiate a deal on price and license/liability terms.

Consumer-facing software is currently subject to a patchwork of liability standards, even at the federal level, with a negligence standard applying only to the extent EULAs can't disclaim it, which means most software won't see a negligence suit in some jurisdictions (but again, reckless or other egregious conduct generally can't be disclaimed). Warranties are a little harder to disclaim, again varying by jurisdiction and case specifics, but EULA language disclaims them broadly anyway.

When UCITA was proposed, a few states drafted "anti-UCITA" statutes that declared software a "good" subject to UCC Article 2, even if licensed, and some courts have also held software should be treated as a good. When software is licensed and treated not as a good, UCC Article 2 warranties don't apply (although when "sold" rather than licensed, it is a "good" in most jurisdictions). Even if and where UCC warranties for sold goods apply to licensed software, they may be subject to disclaimer in EULAs, subject to courts' interpretation of contracts of adhesion in the context of EULAs [7].

For example, some software licenses disclaim just about everything—even violation of intellectual property rights, which could see the end-user sued for patent violation and left to deal with it. Such EULAs essentially say, "this does more or less what we say it does; otherwise, use at own risk. Pay here." In some jurisdictions, software liability is today essentially under a contracts regime, subject to some consumer protection law related to contracts made between parties in unequal bargaining positions. Thus, with negligence and warranty generally disclaimed, subject perhaps to a complicated court battle, some consumers are left to pay for "your problem—deal with it" contract terms on software because they are in a significantly unequal bargaining position with the software producer or seller. Adding further complication, some jurisdictions treat such contracts as unenforceable.

Software liability can thus take the form of liability in negligence, in products liability, in contract (license terms providing a broad range of risk-shifting), and consumer or inter-business contracts for goods (warranty terms, explicit and implied). One almost needs to apply multivariable differential equations to solve for any particular jurisdiction along three major axes, each containing subordinate axes [8, 9]:

1. Liability regime: negligence, products liability, contract, warranty
2. Sold as: license or good
3. Shrink/clickwraps: enforceable or not, and to what degree

All that, without even looking at the complexities of other areas of federal and constitutional law, let alone criminal law.

The Future Isn't What It Used to Be

The complexity in liability for software calls out for a considered standard, even if it is one with broad flexibility. Courts are slowly, but not broadly, rejecting the ability to disclaim warranty in consumer software. But court-considered law is going to be inconsistent by the nature of the market and jurisdiction.

If we push toward a model for software liability, what could it be? If modeled on "goods," would that just be UCC-type warranty plus negligence law, and how much effect should a shrink/clickwrap have? Should we select a products strict liability regime? Is it easier for the industry to measure its "reasonable" behavior or to determine whether a product has "defects" (under the definitions above)?

When software is incorporated in hardware, the combination is sold "as a good" subject to UCC Article 2, with failure due to "defect" likely subject to product liability law. The reasons for that liability include that courts are presented with a product that failed, not an app ("plaintiff's microwave burst into flames"), even when software failure is the root cause. As a matter of public policy, it makes sense because the end-user is several steps removed from the software maker, and thus can't measure risk (the product manufacturer does that) or evaluate license terms (which, between a manufacturer and software supplier, don't look like what you and I normally get in EULAs).

Pressure to control risk is thus between supplier and manufacturer. So governed, a market risk-allocation still takes place, backed by Errors and Omissions/cyber liability insurance on the one hand, and products liability insurance on the other.

Some of my colleagues posit product liability for software will harm the industry. Yet the dizzying matrix of liability on software hasn't stymied software development in the US, from FOSS to mega-commercial. Software makers for products aren't running scared despite contracts between them and the product maker shifting risk onto them, from patent infringement to bodily harm.

To those who create software, a key concern is that the public does not understand the complexity of software, the mathematical impossibility of proving a system, the problems of design versus manufacture. There is concern that the vibrant and effective free software movement will be constrained. After all, haven't we seen the industry forced to improve in a market with viable, quality competition? These are valid concerns and any solution should distinguish among the various forms of license and market model ("sold," licensed for fee, FOSS). All complex systems are subject to subtle defect. Perfection in any form is impossible, its approximation expensive, and we're back to a cost versus quality discussion. Markets are supposed to be good at handling that kind of balance, though they tend to do so after harms appear.

To those outside the industry, it can seem like software makers want a "have their cake and eat it too" liability regime where they can both claim their software is perfect (e.g., "unbreakable") and be virtually without liability should it break, causing harm. That is also a valid concern and sits at the crossroads of a broad range of consumer-protection law.

Should the industry be satisfied with the current patchwork liability? Certainly, end-users of software incorporated in antilock braking systems probably would prefer the system not require a critical patch to prevent catastrophe 3-4 times a year (I am being generous). Such issues as they relate to the end-user are governed by products liability today. Could a reasonable dividing line for the form liability takes be the incorporation of software in a hard good sold as product? Perhaps a "shipped-with" divider between sold-as-good and "licensed"?

Could a manufacturer, rather than selling a good that incorporates software it has licensed, force the end-user to download and "relicense" the braking and other software on first "key-up"? Imagine starting up a new car and clicking through 20 EULAs (or one egregious one), waiving—subject to each state's consumer protection law, subject to each circuit's take on licensing vs. purchasing—all disclaimable liability for anything but mechanical failure. Those who have purchased provider-tied, app-laden smartphones have probably had a whiff of this experience.

These are the discussions we should be having. I hope this surface treatment of negligence, warranty, and product liability has helped arm you with terms and tools to better shape discussion of what "ought" to be, and to understand the complexity of how it "is" today.

References

[1] Michael Scher, "On Doing 'Being Reasonable'," *;login:*, vol. 31, no. 6, December 2006.

[2] For a brief discussion of software and UCC Article 2, see http://technologylicensinglitigation.com/applying-the-ucc-to-software-license-agreements/.

[3] For an excellent history, discussion, and description of UCITA, see http://www.jamesshuggins.com/h/tek1/ucita.htm.

[4] For a contemporaneous response to the UCC 2B proposal, see http://www.badsoftware.com/uccsqa.htm.

[5] American Law Institute, summary of *Restatement of the Law Third, Torts: Products Liability*: http://www.ali.org/index.cfm?fuseaction=publications.ppage&node_id=54.

[6] See, for humorous effect, http://snltranscripts.jt.org/76/76jconsumerprobe.phtml.

[7] Complex issues regarding licenses, EULAs, contracts of adhesion, and unconscionability are at play. A good summary of the development of cases through 2008 can be found at http://www.bicklaw.com/Publications/UnconscionableTermsandE-contracts.htm, and a discussion of click-through/browse-through terms can be found at https://ilt.eff.org/index.php/Contracts:_Click_Wrap_Licenses.

[8] In 1999, Clark Turner and Debra Richardson wrote "Software Defect Classes and No-Fault Liability," presenting an early discussion of the complexity of applying products-style "defect" and liability to software: http://www.users.csc.calpoly.edu/~csturner/fulltechreport.pdf.`

[9] For a similar discussion of the complexity of determining such issues, see Lloyd Rich, "If You Use A Shrinkwrap License It May Not Be Enforceable": http://corporate.findlaw.com/business-operations/if-you-use-a-shrinkwrap-license-it-may-not-be-enforceable-mass.html.

Other Resources:

Legal Information Institute, Products Liability: http://www.law.cornell.edu/wex/Products_liability.

HG.org, Legal Resources, Product Liability Law: http://www.hg.org/product-liability.html.

Macrothink Institute, "A Managerial Guide to Products Liability": http://www.macrothink.org/journal/index.php/ijld/article/view/1773/1458.