Interop

The Promise and Perils of Highly Interconnected Systems

John Palfrey

and

Urs Gasser

BASIC BOOKS
A MEMBER OF THE PERSEUS BOOKS GROUP
New York

CHAPTER FOUR

Privacy and Security

We do not want complete interoperability at all times and in all places. In certain contexts, it can introduce new vulnerabilities, and it can exacerbate existing problems. Privacy and security risks are, of course, the primary concerns when it comes to interoperability. An interoperable system has, by its nature, more points of open access to data. These points of access render systems vulnerable to bad actors—ranging from marketers to hackers with ill intentions—who can potentially access and misuse these data, inject malicious code, or otherwise compromise systems.

Sony learned this interoperability lesson the hard way. Sony discovered in April 2011 that 77 million PlayStation Network (PSN) service user accounts had been hacked. The problem, it turned out, was that its networked gaming system relied on technical and user controls that were not well established. The hacker had obtained user addresses (city, state, zip code), e-mail addresses, birthdates, PlayStation Network passwords and usernames, online IDs, and profile data, as well as possible access to purchase histories, billing addresses, and even credit card numbers. If the PlayStation systems had been stand-alone units, unconnected through a network, the security levels on the systems would not have mattered. Of course, interoperability was not the problem on its own—but it became a problem because the security and privacy controls were inadequate.

The vulnerability of privacy and security within networks is a consequence of the increased *complexity* of an interoperable framework. It is not interoperability per se that gives rise to increased privacy risks but, rather, the specificities of its implementation. The level of privacy and security protections within an interoperable system depends on how we design, implement, and monitor systems at the technology, data, human, and institutional layers. Unfortunately, the information and communications field is full of examples in which companies, in particular, have failed to meet this fundamental design challenge.

The way to frame this design challenge is to focus on the degree to which systems are made to interconnect—the *degree* of interoperability. Interop is not binary. There is no single form or optimal amount of interoperability that will suit every circumstance. Just as interoperability operates at multiple layers, there are many degrees—in fact, infinite degrees—of interoperability. The problems of privacy and security help bring the importance of this dimension into relief.

Most of the time we want an optimal, not a maximum, degree of interop. It is not always desirable to have complete interoperability in every circumstance when pursuing goals such as consumer empowerment or innovation if, at the same time, we want to protect core social values like privacy and security. Lower levels of interoperability—think of it as incomplete interoperability—are often desirable. Sometimes friction in a system is in the public interest. Sometimes it is best to have no interoperability at all—a perfect *lack* of interoperability. And sometimes, where interoperability exists, we have to be prepared to address, through markets or through regulation, heightened privacy and security risks.

Security systems offer instructive examples of how limitations on interoperability can be desirable. Security systems are designed to keep people out of certain places at certain times. A city dweller might leave his door open during the day to invite neighbors to wander through the house during a block party. The same city dweller might triple-lock that same door that same evening while he is asleep. The complex relationship between security and interoperability helps illuminate some of the drawbacks and limitations to complete, omnipresent interop.

Interoperability does not automatically make systems less secure. The fact that a given computer system can interoperate with others does not mean that more people have access to underlying data in the system, for instance. However, depending on the implementation at the different layers and without sound security measures, increased interop among systems can add to the vulnerability of the different components or systems. The opposite also might be the case: systems with higher degrees of interoperability can be rendered more secure than systems with low degrees of interoperability. There can also be high security risks associated with systems that are not at all interoperable.

Security problems are related not to interoperability itself but, rather, to what interoperability makes possible. This distinction makes a difference, especially when it comes to designing interoperable systems. Fundamentally, any system that has more points of open access—for example, one that makes it easier for people to come and go—might give rise to security problems. Put another way, in a highly interoperable system, intruders might be able to enter a space unencumbered and take something to which they are not entitled.

In the information-era version of the same example, interoperability might mean that data can flow through and across systems without restraint. As such, interop also might make it possible for a bit of malicious computer code—known as badware or malware—to slip into a computing system and introduce a virus. Or, to give another example, interop might make it possible for information a teenager has uploaded to Facebook to move from that context into another that the teenager did not mean it to reach. The problem of sexting works this way: a teenager sends a nude picture of herself to a boyfriend, who in turn sends the picture to his friends, who move it from their mobile phones to the web and beyond after the nasty, inevitable breakup. In all these examples, we might be tempted to get the law involved.

This complex relationship between security and interop reveals the fact that the optimal degree of interoperability varies by circumstance. Walls between systems, or rules that function like walls in certain cases, can serve important purposes, especially from a security perspective. Sometimes we want these walls to be permeable, other times not. These policy decisions should drive the way we design interoperable systems. In a digital environment, there are often sliding scales, so that reaching an optimal point of security is at least theoretically possible in some circumstances.

Every once in a while, it makes sense to engineer systems to allow no interoperability at all. Very often, the reason for designing systems in this noninteroperable way is to guard against human error. The example given in the Introduction of the different size nozzles at the gas station—for diesel or regular fuel, neither of which can work with the fuel tank aperture of the wrong type of vehicle—makes this point. This example points not to a drawback of interoperability but to an instance where it is better not to have interoperability at all. There is no foreseeable instance where the driver of a car that takes ordinary fuel would want to put diesel in the fuel tank. To do so would be to risk ruining the engine. At the same time, there are millions of transactions per day in which a consumer, without thinking about it, might reach for the wrong pump at the gas station and run an unnecessary risk. It is smart to keep certain systems from interoperating in order to prevent harm.

More often, situations call for limiting the degree of interoperability in order to accomplish certain goals, such as security. Systems are often designed to work best with a certain degree of friction built in. In such cases, it might be a good thing to make the systems work together in a fundamental sense, at the technology and data layers, and then introduce policy or legal requirements that the parties take certain steps before the systems are permitted to interoperate.

By using sound authentication mechanisms, we can design highly interoperable systems that are also secure. Someone seeking access to a certain physical environment might need to show a series of credentials before being admitted, for instance. Consider the case of a high-security research facility. Visitors are required, by rule, to call ahead to have their names put on the list of people expected on a given day. When they arrive, they are required to show certain kinds of identification. They might have to submit to retina scans, for instance, prior to being granted full access to the facility.

The design of this high-security research facility builds in friction, which is meant to allow interoperability in some cases but not in others. This design principle is *selective interoperability*. Operability is not meant to work for everyone all the time, by design. The idea is that one can permit certain things—such as potentially dangerous or especially sensitive research—to occur in a certain place and enable some people to have complete access, some people to have limited access, and some people to have no access at all.

There are important variants to this concept of selective interoperability. Contrast this approach of selective interoperability to the web services context, where we often see unilateral openness. For instance, a web services provider like Facebook or Google voluntarily creates an open application programming interface (API) that allows anyone to interoperate with its services without the need for further approval or cooperation. Google Maps or its Android platform are examples of systems where developers do not need to ask permission before they start to build. Engineers unrelated to Google are allowed to write code that interoperates so long as they agree to certain legal terms, often a simple click-through agreement with no lawyers involved. The only friction involved is the decision point on the part of the potential participants: the developer must decide whether or not to accept Google's invitation to interoperate, that is, whether the terms are acceptable or not.

It is important to distinguish selective interoperability *by design* from instances in which the exchange of information is limited *in fact*. In the context of office documents, for instance, a computer program puts up a warning screen that says data will be lost when a file is converted from one file format to another. The user may or may not care about what is lost in this transfer. The point is that the systems are not in fact fully interoperable:

some of the data, or data about the data (called metadata), does not flow from the first version to the second version.

This scenario is distinct from the access situation just described, in which all the data can flow from one system to another. In that case, though, *by design*, the system's architects decided to build organizational or procedural barriers making it harder to transfer some or all data across the systems. The high level of interoperability at the technology and data layers in the strict sense is preserved. The limitations occur at another level: in select cases, complete interoperability is not available to the person seeking access. The law or policy decisions, such as lists of approved people who may enter, can function as limits on an otherwise highly interoperable system.

This distinction is important for two reasons. First, it highlights in a precise way what interoperability in the information context means: it is about the flow of data across systems. Second, it exposes design options for addressing drawbacks to interoperability as applied. These design principles do not necessarily mean we have to accept a lower level of interoperability across the board. The example of calling before arriving at a high-security facility makes this point: rather than building permanent barriers, one might, for example, want to be able to give the chief scientist of the institution full access to the facility on the day she visits from the home office.

A variant of selective interoperability is *limited interoperability*. A system might be established to provide only partial interoperability to other systems or interoperability only in certain contexts. Think of the chief scientist coming to the research facility. The access system is designed to allow that visitor access to part but not all of the facility. There are rooms or labs to which she will not be admitted. This is limited interoperability.

Sometimes, the ideal system works like a screen door: it lets in the light but keeps out the flies. There are options as to what functions as the filter. The technology might allow both light and flies to pass through the aperture. Instead of using a screen door as a filter, we could require that a person with a flyswatter be stationed beside the doorway. His job would be to swat all the flies as they go through. The example suggests the upsides and downsides of both options: the screen door might be more consistently effective at keeping out flies, but it might limit the flow of more than just the flies; the man with the flyswatter might only hit flies, but he would surely miss some, which would enter the house.

The many ways to accomplish interoperability come with strengths and weaknesses. The most highly interoperable systems are very often the most permissive of information flowing among them, but that need not be true. Some computing systems are highly interoperable but are built with very high levels of security. Bank ATMs exemplify a system that is highly interoperable in terms of letting customers from virtually any financial institution take out cash from any other financial institution. And yet the system overall is highly secure. The choice is not whether or not to make systems interoperable but, rather, how to manage the flow of information within interoperable systems to ensure the desired level of security.

The loss of privacy is, rightly, what many of us fear most when it comes to interoperability. But it is important to remember that interop does not automatically, by its very nature, operate at cross-purposes with individual data privacy. A perfectly interoperable system can be used in such a way that no one's privacy is violated at all. One might even be able to imagine interoperable systems with privacy-enhancing qualities. But in practice, that is not usually the case. High degrees of interoperability among consumer-facing systems make it easier for someone to violate someone else's privacy. High degrees of interop can also make it more likely that information about a person will be shared with others.

Time after time, companies have had to face the fact that consumers get very unhappy when certain kinds of interop are introduced into systems without full disclosure or explanation. Google and Facebook both had highly publicized problems of this nature within about a year of one another. This privacy-threatening interop can occur within a given firm (Google) or among firms (Facebook). In both instances, the company involved decided to create interoperability of a sort that got it in trouble. These interoperable systems resulted in the exposure of individuals' data in unexpected contexts. The customers involved felt that the companies had violated their privacy: they said they had never agreed to that kind of information sharing.

Google's Buzz, introduced in February 2010, is an example of problematic interoperability within a single firm's products and service. Google's Gmail, which offers free, web-based e-mail with a large amount of storage capacity, has proved enormously popular. For many users, a Gmail account becomes the core of a series of related services they use at Google. Gmail allows users to create an account with Google that can be connected, seamlessly, with other Google-owned services. All the cloud-based office productivity services, such as Google docs, can be connected to this single account. A user's YouTube account, too, can be connected to this Gmail account. As part of this process of integration, many users also integrate their contacts into Gmail, either through an explicit import process from another program or through an implicit process of corresponding with others. Through whichever process, users of Gmail can develop extensive lists of contacts. So far so good: by most accounts, all this interoperability made Google customers' lives easier.[1]

Then along came Twitter, the microblogging platform that took the world by storm between 2007 and 2010. Facebook, too, became increasingly popular during this time, in fact surpassing Google as the most commonly used technology platform in the United States.[2] Both Twitter and Facebook enabled users, through the web or via a mobile device, to post short statements about their whereabouts or to provide comments online. These short messages would be accessible either to their "friends," in Facebook's terminology, or to those who decided to "follow" them, in Twitter's parlance. Google thought users would want such a service connected through Gmail. Google engineers designed a new product, Buzz, to meet this demand.

The problem with Buzz was not the product itself, nor the fact that it was made to interoperate with other parts of the Google suite of online services. In previous cases, customers had seemed to appreciate, for example, the ease of accessing a spreadsheet or a text document in the Google cloud from Gmail. The previous efforts by Google's engineers to render their systems internally interoperable had not been especially problematic from a user perspective.

But a furor broke out shortly after Buzz was introduced. The story is complicated and a bit murky. Users complained that Buzz automatically generated a list of "followers" for each Gmail user, based on the user's most frequent e-mail contacts. According to the court filings, these contacts were displayed in "follower" and "following" lists that were visible to Gmail users' automatically generated Buzz followers and, in some cases, to the public. So, for instance, before Buzz was introduced, there would be no easy way for other people to see the contacts of a given Gmail user (call her Alice). Let's imagine that Alice has a contact named Bob. After Buzz was rolled out, plaintiffs claimed, the names of Alice's contacts, including Bob, might be visible to a third Google customer (call her Claire), without either Alice or Bob having adequately consented to the disclosure.

Further complicating matters, users complained that these lists were generated without notice to, and without the explicit consent of, either user. Google's engineers swear that the services revealed nothing about a user who did not opt in to, or choose to turn on, Buzz. According to Google, Alice's contacts would never be revealed without Alice's explicit agreement. Furthermore, it argued that even for those who did choose to use the free service, nothing was done wrong after a user began to use the service. Google claims that the way the program works was spelled out in writing to the user in sufficient detail that Alice should have known exactly what was going to happen.[3]

Many users did not see it Google's way. A group of Gmail users, with Harvard Law School professor William Rubenstein as one of their lead lawyers, filed a class-action lawsuit against Google. The plaintiffs alleged that Google violated a series of US privacy-related laws through the way they introduced Buzz. At more or less the same time, privacy advocates at the Electronic Privacy Information Center (EPIC) filed a complaint with the Federal Trade Commission (FTC) against Google on similar grounds.[4]

The focus of user complaints was not the functionality of Buzz itself but the way it interoperated with other Google services—in this case, the contacts feature of Gmail—and the exposure of data (such as Alice's relationship to Bob) outside the original context in which Alice put Bob into her list of contacts. To be clear: the problem was not the interoperability as such. The Buzz system might be made to work perfectly from an interop perspective with Gmail and its contacts without giving rise to alleged privacy violations. The problems stemmed from the rules that made this interoperability kick in without, in the plaintiffs' view, sufficient notice about what it would allow others to see in users' contacts lists.[5]

Privacy concerns have emerged as Google's Achilles' heel, with interoperability playing a big part in that development. The short, painful story of the Buzz rollout makes this exposure obvious. Google quickly made changes to its Buzz service after the uproar, and within months the parties moved to settle in federal court. Under the terms of the settlement, Google agreed to pay $8.5 million but admitted to no wrongdoing. Google also made significant changes to its privacy policies to settle related charges filed against it by EPIC with the FTC. Ultimately, the Buzz rollout's total costs to Google will be much higher when one factors in legal fees, the cost of public relations firms, and the losses from reputational damage and continued erosion of trust among a class of users.

Facebook experienced a similarly unhappy experience with the rollout of its Beacon service about two years before Buzz. Beacon is an example of interoperability gone awry, not within a single firm's services but across multiple firms. Facebook enables users to express preferences about people, institutions, and products through a variety of mechanisms on its site and, increasingly, across the web. For instance, a Facebook user can indicate that she "likes" a status update that one of her friends posts online. When President Barack Obama posts to Facebook a short statement about the importance of establishing a clean energy future or of diminishing US reliance on foreign energy sources, thousands of users (or more) might click on a "like" button to indicate their agreement with the president on this topic.

Facebook has emerged as the most popular web service in its core market, the United States. It is quickly growing in popularity in markets around the world. Unlike Google, Microsoft, or Yahoo!, though, Facebook does not have an obvious way to make money from the service it provides. Many Facebook users sign up for the service and never click on the advertisements rendered along the right-hand side of many pages on the site. For those users, Facebook was functionally free; the company captured little value directly from these nonpaying, nonclicking customers. In the long run, Facebook is amassing value from these new sign-ups, of course. As co-founder and CEO Mark Zuckerberg correctly recognized, the huge value in the service rests in the "social graph" that connects hundreds of millions of Internet users to one another. That said, Facebook still needs to find ways to make money from its popularity. The rendering of contextual ads on most pages might make the company profitable, but not as profitable as some of its competitors.

In November 2007 Facebook's executives were determined to be more creative in extracting profit from their product. One of Facebook's ideas for a new revenue stream was to make its system more interoperable with companies that marketed products and services to Facebook users on the web. Beacon, as the service became known, was a program whereby Facebook teamed up with Internet retail companies to make their systems interoperate in order to generate higher profits for both firms. In total, forty-four partner websites, including Overstock.com, eBay, Fandango, Blockbuster, and many others, participated in the Beacon program.

The system was simple in its design, but it sounded pernicious to users when they came to understand what it did. When a Facebook user made a purchase on a partner website, such as Overstock.com, a marketer of discounted merchandise, a few bits of data would be able to flow back to Facebook. Facebook would then render a short notice on the user's Facebook page showing the product he or she had just purchased. This notification could also appear in the news feeds of the user's friends, depending on how the user had configured his or her settings on Facebook. The interoperability at work here: the flow of data about the user from Facebook to Overstock.com, and about the product purchased from Overstock.com back to Facebook.

Just as in the Buzz case, Beacon caused an uproar among web users and privacy advocates. The problem again was not the interoperability among systems but, rather, the exposure of data from one context in another context. The system was also based on a premise called "opt in by default," which meant that Facebook automatically enrolled users in the service unless they explicitly said they did not want to be part of it (which sounds a lot like "opt out" rather than "opt in"). Users expressed myriad concerns, ranging from the fundamental (alarm at the continued erosion of user privacy on the Internet) to the worrying but probably not common (fear that a racy item purchased for a lover by an adulterous husband might appear on a news feed and be seen by his spouse).

A class-action lawsuit ensued in the Beacon case, too, which resulted in a settlement of $9.5 million. For a small, fast-growing company like Facebook, the reputational hit and the hassles associated with the lawsuit represent a far greater loss than the monetary cost of the settlement. Privacy concerns about interconnected web services will only continue to grow and will threaten companies like Facebook and Google if they do not figure out how to address concerns before they blow up into class-action lawsuits.[6]

The problem in the Beacon and Buzz cases stemmed from the way Facebook and Google made their respective systems interoperate and from their failure (their adversaries claim) to notify customers about how data would be used. It is not a foregone conclusion that we will have less privacy and security in a highly interoperable world. But we need to be vigilant about how interop is introduced, in what degrees, and by whom, in order to ensure that we do not give up more than we gain as we progress.

Higher levels of interoperability are not always good, in large measure because they can give rise to problems of security and privacy. One way to avoid these problems is through design. Facebook and Google might simply refrain from designing products and services that interoperate in ways that lead consumers to feel that their privacy has been violated. Sony has no doubt already tightened the security controls on its networked PlayStations since its interop-related debacle.

The largest problem with interop and privacy has to do with businesses that link data about consumers without asking anyone at all, where an especially intrusive form of interop is core to their business model. Massive data aggregation firms such as ChoicePoint should stop drawing data from so many sources in order to enable others to market goods and services to us more effectively. At least in the ChoicePoint example, and to some extent in Google and Facebook examples as well, though, the companies involved generally do not want to design their systems to protect consumers. They have other drivers in mind: low costs, high-margin sales, protection of their competitive position, and so forth. ChoicePoint, and many others, exists *in order* to collect (and then sell) data about consumers; it is hardly realistic to expect the company to abate its core mission for the sake of protecting privacy.

The market will not take care of privacy and security issues on its own, and therefore the law plays an important role in managing this aspect of interoperability. First, and most fundamentally, background rules establish a playing field for companies that work with personal information. There are limitations, established in law, as to what companies can do with an individual's personal information. These limitations tend to be stricter in the European Union than they are in the United States, but in both cases there are legal rules. States occasionally set up specific laws governing particular forms of information: for instance, doctors cannot share information about their patients in an unregulated fashion, which necessarily limits interoperability but produces other benefits. And the law can create a "right of action" for interventions by private parties, as in the Google and Buzz cases, or by state agencies such as the FTC, when the public interest is at stake.

A good example of the state stepping in is the role the FTC played in the Buzz case. At the same time that the private lawsuit against Google was underway, the FTC investigated Google's practices. (EPIC's formal complaint against Google had prompted the agency to act.) The FTC's investigation of the matter was followed by a broad settlement of privacy-related concerns in March 2011. The consent order required Google to "establish and implement, and thereafter maintain, a comprehensive privacy program" to protect consumers.[7]

Security and privacy present clear cases for the need for specific rules that set limits on interoperability. In other examples, we have seen the need for state intervention in order to prompt interoperability in the first place.

These multiple examples come together in a principle about the law in general. The law can function in three productive ways: it can serve as an enabler (of beneficial forms of interoperability, as in the case of public safety and emergency communications), it can create a level playing field (to create a competitive market through interoperability, or so that interoperability can thrive), and it can function as a constraint (to ensure that interoperability does not lead to unwanted effects, such as privacy and security problems). If deployed with skill, the law can play a central role in ensuring that we get as close as possible to optimal levels of interoperability in complex systems. Against the backdrop of reasonable rules, companies can and should design interoperable systems that do not create these problems in the first place.