2.1. Reduce Harm

Harm reduction is a term borrowed from the public health community. In that context, it is used to describe efforts intended to reduce the harm caused by drug use and unsafe health practices, rather than on the eradication of the problem. For example, one of the tenets of harm reduction is that there will never be a drug-free society, and so preparations must be made to reduce the harm of drugs that currently exist since we will never be completely free of them [1,2]. This concept applies to software vulnerabilities as well: that it may be possible to reduce the potential for harm even if vulnerabilities cannot be fully eliminated.

At its core, harm reduction with respect to vulnerable software is about balancing the ability for system defenders to take action while avoiding an increase in attacker advantage. Experience has shown that nearly all software-centric products contain vulnerabilities, and this will likely remain true, especially as code complexity continues to increase.

In fact, the potential for vulnerabilities will likely never go away since a previously secure system can become vulnerable when deployed into a new context, or simply due to environmental changes or the development of novel attack techniques. Systems tend to outlive their threat models. The Flatiron Building in New York City stands as an example of this phenomenon in the physical world. Built prior to the Wright brothers’ flight at Kitty Hawk, NC, today it is vulnerable to attack using an airliner as a weapon. It’s difficult to argue that the designers should have “built security in” for attacks that would have been considered science fiction at the time of deployment [3].

Since vulnerabilities are likely to persist despite our best efforts, CVD works best when it focuses on reducing the harm vulnerabilities can cause. Some approaches to reducing the harm caused by vulnerable software and systems include the following:

- Publishing vulnerability information. Providing high-quality, timely, targeted, automated dissemination of vulnerability information enables defenders to make informed decisions and take action quickly.
- Encouraging the adoption and widespread use of exploit mitigation techniques on all platforms.
- Reducing days of risk. Selecting reasonable disclosure deadlines is one way of achieving the goal of minimizing the time between a vulnerability’s discovery and the remediation of its last deployed instance [4]. Another way is to shorten the time between vulnerability disclosure and patch deployment by automating patch distribution using secure update mechanisms that make use of cryptographically signed updates or other technologies.
- Releasing high-quality patches. Increasing defenders’ trust that patches won’t break things or have undesirable side effects reduces lag in patch deployment by reducing the defenders’ testing burden.
- When possible, automated patch deployment can improve patch deployment rates too.

References