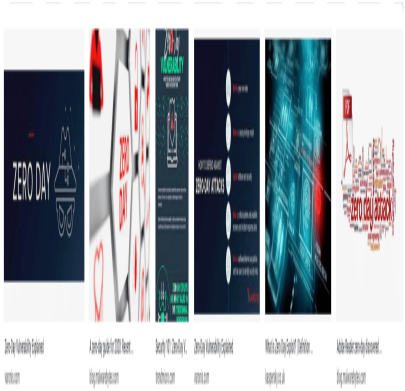

Failure for sale

Description

Things that donâ??t function properly are valuable commodities if their failure offers advantage for someone. In fact, itâ??s information about that failure thatâ??s the commodity. Consider an urban example.

You see that your neighbours have left one of their windows unlatched as they are about to move out for the weekend. So, youâ??ve detected a vulnerability in their security system. You could tell them, and they would close their window to restore their security. Alternatively, if you are untrustworthy you could monetise the neighbourâ??s vulnerability by waiting for them to leave, enter their house through the window and steal their tv set. Even less likely, you could sell the information about the unlatched window to a professional burglar. Even more unlikely, you could add the information to a portfolio, or stockpile, of vulnerabilities about that house and the properties of other neighbours in the street.

Monetising zero-day vulnerabilities

Some vulnerabilities are likely to be short term, such as a window left open while the neighbour is away for a few days. Others might be longer term, such as keys hidden under pot plants and doormats, or a non-functional burglar alarm. Once information about any vulnerability circulates around the neighbourhood then the property owners have *zero-days* to fix it: e.g. lock the window, abandon the spare key idea. Even more nefariously you could extort people in your neighbourhood to pay for you to keep their vulnerabilities secret.

This is a far-fetched scenario for most neighbourhoods, but in the digital world there is a ready supply of hackers ready to detect and exploit vulnerabilities in software, and do so with few risks of being caught. Vulnerabilities can also be communicated instantly and monetised through extortion.

Bugs

Most computer code has bugs. Any computer interaction, utility or function depends on components supplied from a range sources. The operating system provides the environment in which programs function. Programmers draw on shortcuts to various functions, i.e. libraries of subroutines. A weakness or incompatibility amongst any of these components introduces bugs. From the point of view of an end user, such as someone using a wordprocessing or data entry package, the software may simply â??crash,â?• or a function may fail: the search function wonâ??t highlight all the instances of a search term. Other bugs are more serious, resulting in data loss.

Thanks to online monitoring and harvesting user feedback, software companies develop and distribute regular â??patchesâ?• that fix or replace malfunctioning software components, as long as the software company finds out about the bug before it causes serious damage, including reputational damage to the companyâ??s products.

Vulnerabilities

Some malfunctions have implications that extend beyond inconvenience for a single end user. These are vulnerabilities that provide portals into software and systems for the spread of malware that cripples the digital functions not just for the individual using the software, but networks of users, organisations, countries, nations and global systems. Thatâ??s like breaking into the house of a neighbour that contains the keys to all the other houses in the street, or that houses a switch to the local power supply.

In the digital world itâ??s safe to assume there are â??badâ?• actors, operating alone or in groups dispersed across networks, or who operate as rogue employees of government, or who belong to benign or rogue states. All are bent on detecting and exploiting vulnerabilities in software. Itâ??s wild out there.

We can also assume that there are many points of entry to any networked system. Software, such as the Microsoft operating system, is widely distributed. Not all individuals and organisations run the latest versions of the software or institute all patches and fixes when available. Vulnerabilities gather at the least secure nodes in such networks

Malware

An ACM article by Stephen Wicker outlines some of these software vulnerabilities. Some enable a rogue agent to see or compromise confidential data. The most intriguing and worst kind of exploitation is that which distributes, loads and runs *malware*. As we are dealing with networks of computer systems, within and across organisations, the risks caused by malware can be consequential.

How are these vulnerabiities found? There are many methods. One obvious route for a hacker is to steal the source code somehow and analyse it for potential failure in the event of bad input data, e.g. that it canâ??t trap and that causes memory overflows. The most common method of attack is â??fuzzing,â?•

> â??a brute force approach in which the attacker provides overly large or otherwise unanticipated inputs to a program and then monitors the responseâ?• (99).

Exploits

Once the hackers know what makes the software system fail then they can hold the software to ransom for the users who depend it. This was the tactic in the case of the malware breach of the UKâ??s National Health Service in 2017. The so-called *WannaCry* ransomware attack exploited vulnerabilities in the Microsoft operating system. The US National Security Agency (NSA) had already detected the vulnerability and kept that knowledge secret. They might need the information as part of their cyber security defence and attack weaponry.

But other hackers had also come across the vulnerability, perhaps through a leak from the NSA or by some other means, and exploited it by disabling crucial systems and demanding payment in exchange for restoring the data. Software users were angry at the hackers of course, but the NSA could have reported the vulnerability to Microsoft who would then have patched it, and spared the NHS and other organisations the ransomware scam. The NSA had nicknamed the Microsoft vulnerability â??EternalBlue.â?• Wicker explains the problem.

> â??Having learned of (or discovered) EternalBlue, the WannaCry perpetrators used the vulnerability to put target machines in the desired vulnerable state, and then issued a â??request dataâ?? command that caused an encrypted viral payload to be loaded onto the target
> machines. The payload included ransomware as well as software that
> searched for other machines that had the same vulnerability. The ransomware rapidly propagated across the Internet, infecting machines that shared the EternalBlue vulnerabilityâ?• (99).

Shadow brokers

Knowledge of vulnerabilities is a marketable commodity. Users, systems operators or hackers may happen upon these vulnerabilities by serendipity, or may actively seek them out. Once detected, it would be appropriate to report any vulnerabilities to the software supplier, who would then write code to obviate the problem and distribute that as patches or upgrades. That may take a few days or weeks, during which time the security of the software and the systems that depend on it are compromised. There are *zero days* to fix the problem.

Cyber security advocates could argue that the security of citizens is well served by government agencies able to exploit vulnerabilities in the software and systems of hostile foreign actors. The information has value to the NSA who can exploit the vulnerabilities in disabling enemy systems.

As well as the software developers and suppliers, nefarious 3rd parties are interested in knowing about the vulnerabilities. An online article by Matt Suiche in 2016 revealed that a group of hackers known as the *Shadow Brokers* detected the EternalBlue vulnerability and offered the information for online auction. I donâ??t yet know if they sold that information to the NSA (or if the NSA was the source).

The infrastructures of entire cities have been compromised through ransomware exploits. When government security agencies learn of such a vulnerability should they disclose this to the vendor, or keep it to themselves? See post Hacking the City of the future.

The exploitation of urban vulnerabilities

Apply some of these software security issues to cities. The metaphor of *urban vulnerability as commodity* offers an interesting lens through which to think about city challenges. The nefarious exploitation of information about vulnerabilities conjures up scenarios of protection rackets, black markets, insurance scams, urban fear mongering, and the promotion of walled security enclaves.

Failure by a player is worth something to bad actors who rig games and sports that involve betting. Traders in stock markets can benefit from insider knowledge about impending failure.

As another nefarious practice, political parties and interests can put forward â??spoilerâ• candidates whose policies accord with some opposition voters and effectively splits them away from their support for the mainstream opposition. These candidates are encouraged to run and installed to fail and to dilute the vote.

There are brokers who trade portfolios of failed urban enterprises, failed retail outlets, unprofitable property investments, etc. These provide a way of avoiding corporate taxes in some countries. Bad competitive practices may induce failure in competing enterprises to facilitate takeovers.

Once written off, products that fail in terms of profitability can be used to bulk up product offerings, and serve as lures within bargain offerings.

Under this lens, wealth disparity, homelessness, uneven access to technical and social infrastructures also constitute urban vulnerabilities readily exploited by consumerism, negative and disruptive political forces â?? not to mention pandemics.

References

- Buchanan, Ben. 2016. The Life Cycles of Cyber Threats. *Survival*, (58) 1, 39-58.
- Suiche, Matt. 2016. Shadow Brokers: NSA Exploits of the Week. *comae*, 15 August. Available online: https://blog.comae.io/shadow-brokers-nsa-exploits-of-the-week-3f7e17bdc216 (accessed 14 April 2021).
- Wicker, Stephen B. 2021. The Ethics of Zero-Day Exploitsâ?? The NSA Meets the Trolley Car. *Communications of the ACM*, (64) 1, 97-103.

Note

- Picture of the open door is a painting on a door at a tea dealership in Leith, Scotland, 2004.

Category

1. Architecture

Tags

1. hacking
2. security
3. urban vulnerability

Date Created
May 1, 2021
Author
rcoyne99