**How protective
software is
kept safe from
compromise**

2019

Securing your security

kaspersky

Can Your Security Software Get Hacked?

All commercial software has some vulnerabilities which can be exploited as a means of introducing malware. The nature of today's sophisticated software, and the sheer quantity of code involved, means that it is effectively impossible to guarantee that a software application of any significant level of complexity does not include vulnerabilities.

The word 'vulnerability' has a specific meaning in this context – it is any weakness in the software code which can be exploited by a threat actor in order to introduce malware into the system running that software, or otherwise put it under threat. Finding these weaknesses in commercial software is an underworld business in itself – specific details of vulnerabilities in popular applications and operating systems are purchased by malefactors on the dark web for considerable sums. The other arm of this criminal industry is the manufacture of 'exploits' – pieces of code designed to exploit these individual vulnerabilities and use them to inject malware into your systems.

Exploiting vulnerabilities in commercial software is how the vast majority of corporate cyber-attacks, even the most complex and sophisticated, originate.

> **The most used exploit**
>
> The most frequent detection verdict that Kaspersky Lab customers have seen last year was Win32.CVE-2017-11882.an exploit for well-known vulnerability in MSOffice software
>
> Securelist, Spam & Phishing in 2018

From the moment a new version of a popular application, or operating system, is launched, its originators are in a race with the cybercriminals to see who can find its vulnerabilities first. Once a vulnerability is identified, the original manufacturer will create a new fragment of code eliminating the vulnerability, which is delivered to the end user within an updated version of the software, or in a 'update' or 'patch' which, when installed, eradicates the vulnerability. Obviously, that's why it's important to keep your computer software updated

That's all very well with a single computer. But constantly patching all the software used in a corporate IT environment is a big job. In fact, it's an almost impossible one, in view of the number of updates an IT department may receive each week, and the difficulty of applying them all without significant disruption to the business.

Some security software vendors offer solutions to this issue. Organizations using Kaspersky Endpoint Security Advanced or Kaspersky Total Security for Business, for example, benefit from vulnerability & patch management functionality, which helps identify which software requires patching, and what updates are most urgent and critical. Automated patching can be set to apply the prioritized patches at the most appropriate and convenient times -making IT administrators' lives much easier.

So – all good. Your security software ensures that vulnerabilities in the software you use are patched in a timely and efficient manner.

Hold on though....

If all commercial software inevitably contains vulnerabilities, does that include the security software itself?
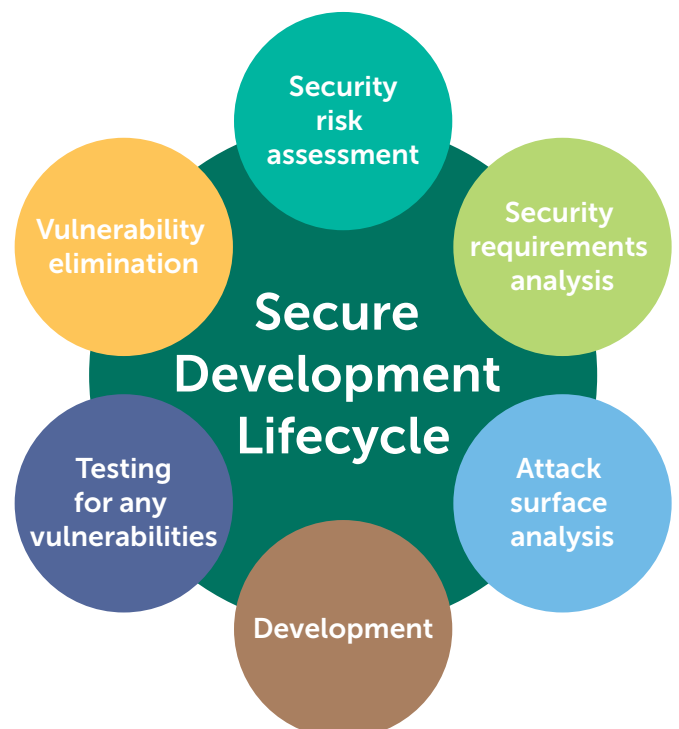
Unfortunately, it does.

Most software manufacturers make serious efforts to integrate Secure Development Lifecycle (SDL) practices into their R&D processes, in order to ensure that their products are secure. And security systems manufacturers, by definition, do this better than most. But no software vendor can ever claim 'It's impossible for cybercriminals to detect any weaknesses in our products '. The best we can do it to get as close to that ideal as is possible. The alternative – no vendor to launch any new software until they can 100% guarantee that it contains not a single weakness that could be exploited by attackers – would mean in practice that no software would ever get launched at all.

The Road to Making Products Secure

How can we make our products as secure as possible? By bringing together a number of approaches.

Security from the ground up

A secure product architecture, and a development process which incorporates risk assessment and analysis at the earliest stages, both go a long way towards minimizing the potential for vulnerabilities in the finished product, right from the start. Testing and elimination during the latter stages of development, and ongoing post-development vulnerability identification and eradication, involve R&D, product management and support all working together to complete the security cycle. At Kaspersky Lab, our SDL is further supported by our Product Security Champions and input from our Anti-Malware Research team.

Secure Development Lifecycle

- Security risk assessment
- Security requirements analysis
- Attack surface analysis
- Development
- Testing for any vulnerabilities
- Vulnerability elimination

Eliminating vulnerabilities

At Kaspersky Lab, we have a number of ways of finding those elusive vulnerabilities which somehow manage to survive our SDL processes, so that we can close them off.

These include:

- Our internal security audits, which include actually attempting to hijack our own applications and trying to penetrate our own systems.
- The HackerOne platform. Here, independent researchers, including 'white hat' hackers, can earn a 'bug bounty' by providing data about any vulnerability they can find in our products.
- Our dedicated vulnerability reporting mailbox and web page - some reporters may prefer to use these to alert us to possible vulnerabilities.

White Hat Hackers

These are highly skilled free-lance security researchers who enjoy the excitement and challenge of tracking down software vulnerabilities, but who work ethically to protect users, rather than marketing their findings on the dark web. We pay a bug bounty for any genuine vulnerability in our software that white hat hackers can find and report to us.

Sources of vulnerability intelligence

Responding to vulnerability reports is a whole separate process, Our Product Security Team (PST) works to confirm a vulnerability and to assess its severity, while product teams work to fix the vulnerability within rigorous timescales., correcting the code that allowed it to exist in the first place . If, as happens, the vulnerability turns out to be a weak spot inside an operating system rather than in one of our own products, we take a different approach – alerting the OS vendor and working with them to resolve the issue.

Safe from the inside

While exploiting vulnerabilities is by the far the most common way to attack any system, there are other ways in which attacks can be launched via popular applications, which includes security solutions.

There have, for example, been cases where 'weaponized' packages of commercially available security software, adapted by cybercriminals to enable an attack on the user, have been offered as downloads. These have generally originated from dubious download sites, so can easily be avoided by downloading only from the vendor's own resources. An attacker could perhaps manage to corrupt the stored version on the actual vendor's site – but this would involve using fake or stolen certificates that would not allow such operations to last for long.

ASUS - Attacked from within

In early 2019, Kaspersky Lab researchers discovered that a live software update tool, downloaded from the official site of computer manufacturer ASUS, had been hacked to install a malicious backdoor onto customers' computers. The attackers had managed to hijack the ASUS server – the hacked software even featured legitimate ASUS digital certificates!
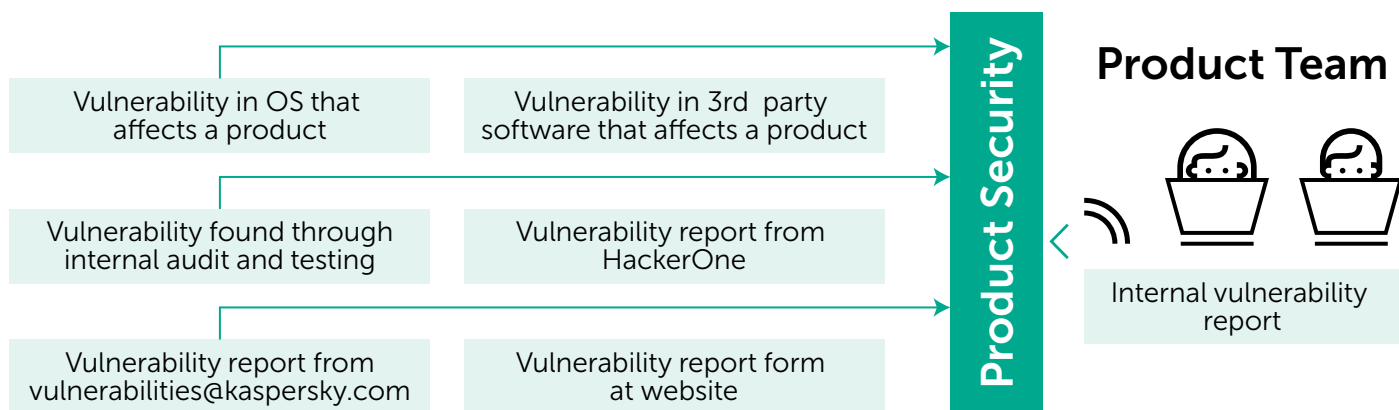
Source: Securelist.com

The real danger lies in a cyber-attacker getting access to any developers' own building factory, where they could theoretically introduce portions of malicious code into legitimate software right from the outset, creating a perfectly disguised attack tool.

But this can't happen if security is built into all aspects of the development process, as is the case with Kaspersky Lab. A securely built environment incorporating code integrity monitoring and the automated testing of all components as they are developed – plus regular security assessments of the apps and the infrastructure itself, and a multitude of other precautions we'd rather keep to ourselves - all this makes it virtually impossible for any insider attack to get to first base, let alone to succeed.

Vulnerability in OS that affects a product	Vulnerability in 3rd party software that affects a product
Vulnerability found through internal audit and testing	Vulnerability report from HackerOne
Vulnerability report from vulnerabilities@kaspersky.com	Vulnerability report form at website

Product Security

**Product Team**

Internal vulnerability report

Never stopping learning

As we integrate secure practices into our software development lifecycle, we ensure that every developer and architect is trained in all aspects of product security. This way, we know that security will automatically be at the forefront of every design decision.

DevSecOps: substance under every syllable

In line with our principles of built-in, continuous and adaptive security, Kaspersky Lab has been implementing the principle of Development and Operations connected through Security (currently known as 'DevSecOps') since well before this buzzword had become known. This integrated approach to security means that that even in a case of compromise – and as we have said, nobody, even a security vendor, is 100% bulletproof – our customers would remain as safe as ever. That's the foundation of the world's most tested and most awarded security.

Areas of security training for Kaspersky Lab developers and architects.
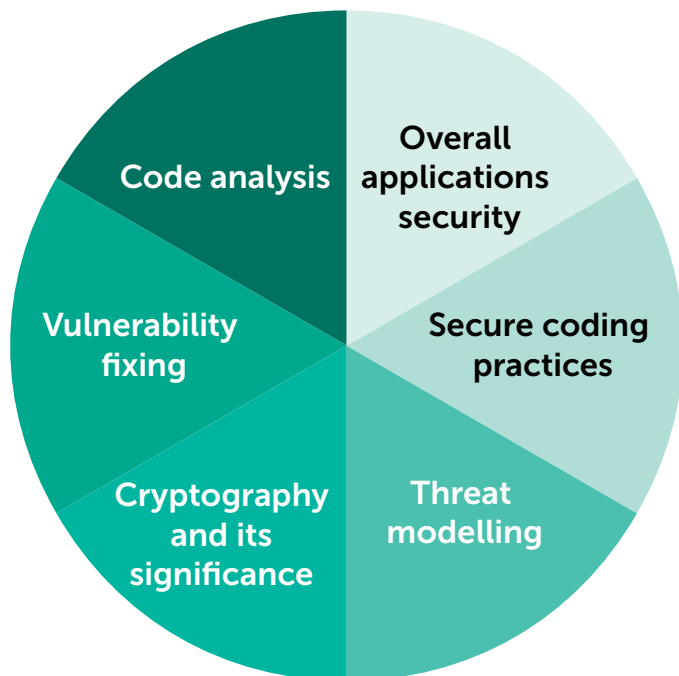
For more of the latest thinking on corporate cybersecurity and threat developments, visit Kaspersky Lab's **business blog**.

We are proven. We are independent. We are transparent. We are committed to building a safer world, where technology improves our lives. Which is why we secure it, so everyone everywhere has the endless opportunities it brings. Bring on cybersecurity for a safer tomorrow.

Known more at kaspersky.com/transparency

Proven.
Transparent.
Independent.