opentext™ | Cybersecurity

Position Paper

Guide to Protecting Source Code

Table of Contents

In 2011, technologist and venture capitalist Marc Andressen—who co-wrote the Mosaic browser and co-founded Netscape in the early 1990s—famously gave scope to the rising importance of software, writing, "software is eating the world."[1] At the time, about 1.3 million people in the United States worked as software developers (540,000 workers), systems developers (390,000 people), computer programmers (320,000 people), or Web developers (under 100,000 people).[2]

A decade later, software *has* eaten the world and become the basis for global commerce. Unsurprisingly, the role of developers has taken off—with the number software developers in 2021 estimated between 4.5 million[3] and 6.0 million[4] workers—a range that represents a significant increase over a decade ago. Those developers are estimated to produce 70 billion to 90 billion lines of new code every year.[5]

For companies across the world, this intellectual property represents a major security issue. Source code only remains valuable as long as developers have access to the codebase and can improve, analyze, and modify the application every day. However, the accessibility of the code means that tracking who has access to it and whether they are abusing their access is difficult.

As a result, source code is notoriously hard to protect. Case studies in this report include employees walking out the door with source code after being hired by another company, foreign nationals copying source code to bring back to their home country, and external hackers stealing—and then selling or simply publicly posting—source code from a wide variety of companies, such as game developers and security firms.

This paper will explain the best practices in protecting source code, how machine learning can tackle the problem of the complex inter-relationship between developers and an organization's codebase, and why augmenting current security controls around source code with user and entity behavioral analytics (UEBA) can improve a business's visibility into its use and misuse.

The (Security) Problem with Source Code

Case 01: Goldman Sachs Programmer (2009)
A programmer for Goldman Sachs received an eight-year prison sentence in 2011 as a result of being found guilty of copying source code from the investment house after taking a new position at rival company Teza Technologies. Sergey Aleynikov, 40 at the time, claims to have wanted to recover open source code from the program files and argued that only 32 megabytes of about 1.2 gigabytes of code were proprietary.[6]

1. Andressen, Marc. "Why Software Is Eating The World." *The Wall Street Journal*. Opinion Column. 20 August 2011.
2. Kow, Nicole. "The Evolution of Developer Salaries: Looking Back 20 Years." CodeSubmit. News Analysis. 11 February 2021.
3. In 2019, Evan Data Corp. estimated that there were 4.4 million software developers, while the US Bureau of Labor Statistics estimated a 24% growth rate for the next decade, which is a 2.2% CAGR. **Note:** The BLS estimate for the number of "Software Developers, Quality Assurance Analysts, and Testers" is 1.9 million.
4. McEnergy, Sage. "How much computer code has been written?" Modern Stack. Web article. 18 July 2020.
5. McEnergy. "How much computer code has been written?"
6. Lewis, Michael. "Did Goldman Sachs Overstep in Criminally Charging its Ex-Programmer?" *Vanity Fair*. Article. 1 August 2013.

Goldman Sachs reportedly uncovered the data theft only after the company began monitoring HTTPS transfers, discovering that a large volume of data had been sent to a website hosted in Germany. Aleynikov had used a Subversion repository to back up his files, but Goldman Sachs alleged he also copied, compressed, encrypted, and renamed the files—suggesting that his intent exceeded merely recovering open source files. Also suspicious? He allegedly deleted the encryption program when the task was complete and attempted to delete any log files that would have shown signs of his activities.[7]

Historically, the primary way to protect source code has been laws and legal agreements. Yet, despite more than five decades of legal cases and attorney opinions, questions abound about the legal status protecting source code. Usually, source code is protected by copyright and, in some cases, the mechanisms implemented in the source code are patented. The software does not have to be registered to be protected, but to take legal action for infringement, formal registration is required.[8]

Whether employees understand the legal protections afforded to their employers for source code created by individual developers remains a question. Often, employees will feel a sense of ownership over the code that they worked on and helped create. In addition, employees who work on their own projects outside the office—or outside of work hours, for remote workers—can add a wrinkle, as can open source projects, whose licenses could pollute the ownership of code. These issues make relying on legal mechanisms for source code protection impractical.

This paper, however, is about using technical tools to protect source code, which poses two general problems.

Friction Slows Development

Security controls around source-code access can slow critical software development time. Code needs to be very accessible, especially for developers. While a local server or internal source code repository might have suited software development groups in the past, remote work and collaboration means that source code often moves outside the firewall, with some companies completely relying on software-as-a-service (SaaS) offerings such as GitHub or Atlassian's Bitbucket. As the shift to remote work becomes more established, companies will have to protect source code no matter where it resides, whether in an on-premises server, in a private cloud development environment, or on a public cloud service.

To be productive, developers need to be able to read, modify, and compile source code, making repositories focused more on accessibility and source code management than on limiting access and security controls. Companies benefit from reducing any friction to developer productivity, which often means minimizing the number of security controls surrounding the source code on which developers work. Every access alert can cause delays in development. Security that mistakenly prevents access to needed source code can stop development cold.

7. Zetter, Kim. "Goldman Sachs Programmer Found Guilty of Stealing Code." *Wired*. Online Article. 10 December 2010.
8. Villacreses, David Chang. "Protecting Your Software Ideas: to Copyright or to Patent." Office of Technology and Commercialization Blog. Duke University. Article. 4 November 2020.

Companies Focus on the Wrong Threats

To protect against the most common types of source code leaks, companies should focus their efforts on the most common source of those leaks. Insiders often make errors, leading to security breaches and data leaks, but developers can also sometimes turn malicious and steal code.

Careless, or Entitled, Insiders

In some cases, the work a programmer does engenders a feeling of ownership of the source code and they feel entitled to take it with them. Because technology companies have a high turnover rate, with nearly 25–30% of all developers leaving in their first year and the average developer leaving in 1–3 years,[9] such feelings of ownership can lead to source-code leaks.

Employees—even technically minded developers—also make security mistakes, allowing their credentials to be phished by an external attacker or failing to correctly configure certain files and exposing sensitive code or secrets in public repositories. In its 2021 breach, development tools maker Codecov had a utility modified by malicious attackers after incorrectly configuring a Docker container used in the development process.[10]

Using legal tools—or the action of last resort, a lawsuit—is usually the worst case and can hurt employee morale. Education about work-for-hire relationships and the organization's expectations can limit source code leakage from misperceptions of ownership. Technical controls can act both as a way of monitoring potential harmful conduct and oversharing and as a reminder of the company's intellectual property boundaries. In addition, security training can remind developers that storing code on public repositories and working on code in a public place allows attackers to potentially gain information on the code or the code itself.

Malicious Insiders

While most employees who copy or otherwise abuse source code access are not malicious, bad actors do exist. As demonstrated by the case studies in this brief, dozens of developers and programmers have knowingly taken source code with them when they have left one company and moved on to another. Most often, the former employee argues that the copying or transfer of source code was accidental. Detecting such activity is critical to preventing a company's intellectual property from being stolen.

External Attackers and Ransomware

Finally, ransomware groups and other cybercriminals have historically found source code enticing and valuable. Intruders that gain access to a victim's network will usually look for the most valuable data, such as source code repositories, as well as the credentials to access development systems. While companies already have technical controls for detecting intrusions, focusing specifically on employee behavior surrounding source code can detect when unauthorized actions or access need to be investigated.

9. Gitential. "Why Software Developers Leave." Gitential Blog. Online Article. 1 June 2021.
10. Engelberg, Jerrod. "Bash Uploader Security Update." Codecov. Online advisory. 15 April 2021.

In just the past decade, hackers have stolen source code for A-list games from Electronic Arts and CD Projekt Red[11]; stole the codebase for enterprise security products from Symantec[12] and Mimecast[13]; and used a misconfigured source code management tool to steal data from US government agencies.[14] In addition, attackers are increasingly using access to source code to modify legitimate programs in order to take malicious actions. The massive compromise of government agencies and large enterprises via an update from SolarWinds started in January 2019, when attackers linked to Russia gained access to that company's source code and inserted code into an update for its network management system.[15] Similarly, Russian attackers compromised the development and update infrastructure of Ukraine accounting software firm MeDoc, distributing malware within an update for the firm's accounting package.[16]

Traditional Security Tools Are Not Enough

Case 02: IBM Programmer Stole Code (2018)
In 2018, a former IBM programmer received a five-year prison sentence after pleading guilty to charges that he had stolen proprietary IBM code to create his own business service. The FBI began investigating Jiaqiang Xu, 32, after receiving a report in 2014 that someone in China had claimed to be using IBM source code for their own business.[17] IBM did not detect the theft. Instead, the FBI recorded a conversation with Xu in December 2015 in a hotel, where he admitted to stealing the source code.

Current data-leak prevention (DLP) technology tends to focus on specific types of information, such as Social Security numbers, financial records, or personnel files. Usually, this type of data can be protected by using access controls and pattern-matching techniques that prevent information from being inadvertently or maliciously copied or transmitted. Only a small group of employees need to access the information and only in specific circumstances that are known beforehand.

Too Little Protection or Too Much Friction

Developers, however, need access to code all the time and need to be able to change and build software using code from a central or cloud-based repository. Security that gets in the way of the developer—causing friction—will lead to dissatisfaction and potential attempts to circumvent technical controls, not to mention slowing development velocity. Many traditional data security tools slow down or block the transfer of data, which is not a problem when speed of access is not critical. However, in development, every false positive alert slows down the production of code by programmers who are overwhelmingly responsible actors. This makes false positives as much of a threat to a secure development lifecycle as theft.

Yet, relaxing security too much makes technical control ineffectual. Security technologies, such as firewalls and encryption, are not sufficient to protect source code because they cannot discern whether a specific developer is handling source code in an appropriate way. Developers have regular access to source code and regular access to a variety of ways to exfiltrate source code.

11. Lemos, Robert. "How Gaming Attack Data Aids Defenders Across Industries." Dark Reading. News Article. 19 July 2021.
12. Robertson, Adi. "Symantec warns users to disable pcAnywhere in wake of source code theft." *The Verge*. News Article. 25 January 2012.
13. Osbourne, Charlie. "Mimecast reveals source code theft in SolarWinds hack." ZDNet. News Article. 18 March 2021.
14. Cimpanu, Catalin. "FBI: Hackers stole source code from US government agencies and private companies." ZDNet. News Article. 7 November 2020.
15. Vijayan, Jai. "SolarWinds CEO: Attack Began Much Earlier Than Previously Thought." Dark Reading. News Article. 19 May 2021.
16. Vijayan, Jai. "3 Years After NotPetya, Many Organizations Still in Danger of Similar Attacks." Dark Reading. New Article. 30 June 2020.
17. Pierson, Brendan. "Ex-IBM employee from China gets five years prison for stealing code." Reuters. News Article. 29 January 2018.

Cybercriminals that gain access to source code most often do so by gaining access to a developer's account or system, allowing them to pose as that developer. In either case, encryption and limiting access by application will not prevent the theft of source code.

A History of Weak Approaches to Source Code Protection

In the past, companies have maintained internal source code repositories to keep the data centralized and to be able to monitor developers. As remote work and collaborative development has become more common, cloud-based repositories have become more popular. More than 96 million developers have accounts on GitHub (56 million),[18] Gitlab (30 million),[19] and Atlassian's Bitbucket (10 million).[20]

Most recommended security measures focus on preventing unauthorized users from accessing source code, preventing secrets from leaking in repository files, and preventing malicious changes to the code. The United Kingdom's National Cyber Security Centre, for example, recommends choosing trusted repositories, protecting access credentials, revoking access as soon as possible, and reviewing all code changes.[21] Yet, protecting code under active development is like trying to protect an office building with a dozen entrances, and where the vast majority of people are workers authorized to be there.

No wonder, then, that companies most often only catch source code theft when employees lacking knowledge in covert theft move massive amounts of source code outside the company firewall. In 2009, Goldman Sachs caught a programmer, Sergey Aleynikov, allegedly copying a significant amount of source code to a Subversion repository. The developer maintained that the process was a standard practice.[22] The alternative is that the company fails to catch the theft at all, as happened to IBM in the incident discovered in 2014.

Source Code Management Requires Behavioral Analytics

Case 03: Tesla Lawsuits (2018, 2021)

Tesla has sued multiple employees for alleged theft of source code. In 2018, the company sued its employee, Guangzhi Cao, for allegedly copying Tesla's Autopilot source code. While the engineer later admitted that he uploaded the source code, he maintained that he attempted to delete all files before leaving Tesla. Tesla settled with Cao in early 2021 for undisclosed terms.[23]

In 2021, the company also sued Alex Khatilov, who worked there for less than a week before allegedly copying the code for the company's WARP Drive backend system to Dropbox.[24] The company detected the threat when Khatilov began copying a significant amount of data outside the company's network.

18. "The 2020 State of the Octoverse." GitHub. Web Report. September 2020.
19. "About Us." GitLab. Web Page. n.d.
20. Yap, Kevin. "Celebrating 10 million Bitbucket Cloud registered users." Bitbucket Blog. Web Article. 17 April 2019.
21. National Cyber Security Centre. "Secure development and deployment guidance." Protect Your Code Repository page. Web page. 20 February 2019.
22. Lewis, Michael. "Did Goldman Sachs Overstep in Criminally Charging its Ex-Programmer?"
23. Cranz, Alex. "Tesla settles with ex-engineer accused of stealing Autopilot source code." The Verge. News Article. 15 April 2021.
24. Kolodny, Lora. "Tesla sues former employee for allegedly stealing software code." CNBC. News Article. 22 January 2021.

Attempting to defend source code using rules that must be created, vetted, and deployed by a human analyst leads to overwhelming workloads. When the novel coronavirus surged in the United States in early 2020, companies instructed employees to work from home. The sudden change in work location resulted in security controls needing to be adjusted and rules rewritten. On a smaller scale, these types of behavioral anomalies happen every time a worker moves to a new office, development teams adopt new applications or infrastructure, or a sprint causes employees to work overtime.

The activity of a single employee can create hundreds[25] to thousands[26] of events that are logged every single day. Without a way to eliminate the run-of-the-mill events and focus on the most critical issues, organizations' security operations teams will be overrun. At best, overworked analysts will respond slowly to critical threats. At worst, the analysts will fail to spot an attack or the theft of source code.

Automating the SIEM's ability to learn normal behavioral patterns and using additional machine learning techniques enable companies to create behavioral profiles of different roles in the development process—minimizing human work and automatically capturing changes to behavior. The resulting security controls are more accurate and more flexible. This type of security software is often referred to as user and entity behavioral analytics, or UEBA.

Here are the three steps needed to secure source code with automation and machine learning.

Collect the Right Data

The first step is having the right data. Companies have a wide variety of data sources on which to create a baseline of activity and identify anomalies. Server-based and cloud-based repository data provides granular access and code-change events that can give security operations insights into how code is being used. Information from GitHub, GitLab, and Bitbucket can create a solid foundation of event data. A user connecting from a country that had previously not been encountered, for example, should raise red flags. In addition, a user moving a large amount of data into an untrusted site or cloud server would be a strong indicator of a potential source code leak.

Yet, relying on these systems will often miss events that provide additional insight into the behavior of a user. A developer copying data to a USB drive or encrypting source code data should trigger further investigation. For that reason, a behavioral analytics system can benefit from additional data, such as information from the telemetry of endpoint detection and response (EDR) systems. Even though the data is not specifically about source code, EDR telemetry provides rich information on local user activity—information not seen in data collected from the source code repository. However, a user that moves a large amount of data to USB drive, for example, will leave traces in the EDR telemetry.

25. Hale, Brad. "Estimating Log Generation for Security Information Event and Log Management." SolarWinds. Whitepaper. 22 January 2013.
26. Zscaler estimated "2,000 to 5,000 transactions per day" in 2017.

Watch for Anomalous Behaviors

Detecting an anomalous event requires that the system build a baseline profile of what is normal. Most behavioral analytics systems will have to spend a period of time training the machine learning models to learn the baseline of the organization and its users to be able to determine if behavior is normal or anomalous.

Once normal behavior is defined, the system can watch for any anomalous behavior, such as an employee accessing source code at an unusual time of day or from an unusual IP address. Suspicious access could also be tied to potentially risky activities, such as a developer checking out a large number of files or copying those files to offline storage. In addition, applications that the developer might not have previously used, such as PowerShell, could also be a red flag. Finally, anomalous behavior can be related to the content, such as a developer looking at code that they have not worked on before.

Creating a similar system using rules instead of machine learning can be tedious, overwhelmingly complex, and will inevitably miss something. The automated construction, verification, and application of behavioral models can provide more and consistent visibility into the status of malicious behavior in an o

A MOVING MODEL OF WHAT IS 'NORMAL'

The sudden move to remote work during the pandemic caused havoc with human-curated rules. Logging into a subset of systems from the office between the hours of 7 am and 6 pm might be normal for some employees, but because of work-from-home mandates following the coronavirus pandemic, profiles inevitably changed. A rule-based system that did not adapt to the post-pandemic business model of developers working from home likely inundated analysts with warnings based on the detection of a new location.

Companies that approach the problem of source code security using an automated collection of threat data and behavioral patterns will benefit through greater adaptability. Not only will such systems be able to recognize team behaviors changed by circumstances, but will also recognize when an individual is not acting in the same way as their team. More advanced machine learning models, called unsupervised machine learning models, can cluster developers working on the same projects into groups and detect when one developer is attempting to access a project in a different group *(see Figure 1)*.

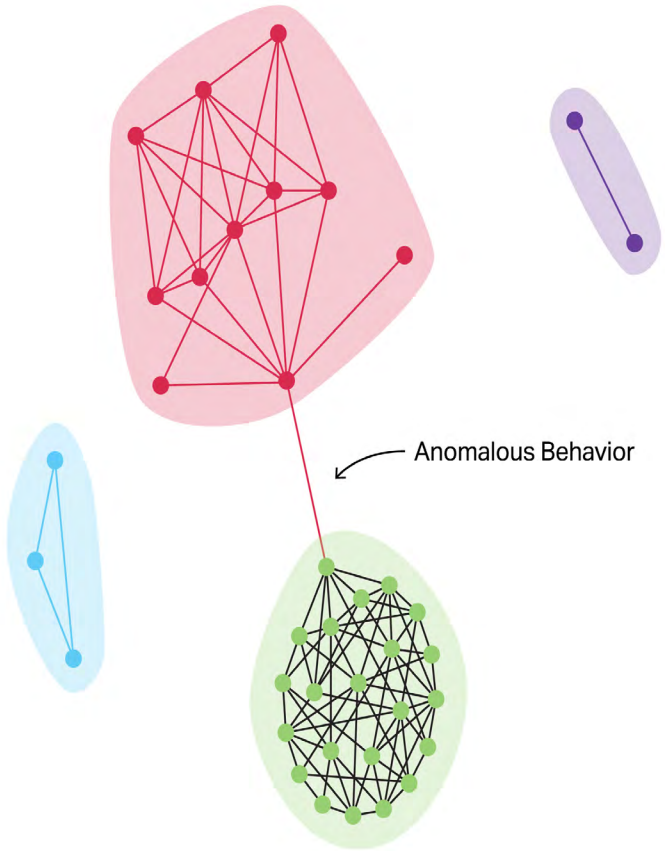

Figure 1. "Spring into Action: Protect Source Code with Behavioral Analytics," Stephan Jou, OpenText

What constitutes an anomaly?
A behavioral model typically will create a baseline of normal behavior, classifying the activity of each user by the characteristics that the model is designed to watch—whether it is the time at which they usually start work, the location from which they connect, the files on which they normally work, or the programs that they typically use to do their work. Anomalies are activities whose characteristics lie outside those expected behaviors.

Applied to source code ecosystems, relevant anomalies can be divided into four different categories: covert activity such as encrypting data, collection activity such as copying files, access activity such as connecting to services outside their role, or asymmetric access to resources when users check out more files than they check in. Yet, other activity will also be considered, expanding the model and using other relevant data.

Hunt for Threats Using the Model
With behavioral analytics automating much of the threat detection, analysts are freed up to do deeper investigation and response to higher quality threat leads. If an automated intelligence system uses a variety of data points and extracted features to determine that a particular user is acting anomalously, an analyst could follow up by identifying other users that come from the same IP address space or at the same time of day, leading to more potential threat candidates.

Source Code Incidents Cross Industries

Case 04: Programmer from Chinese Drone Maker DJI (2019)
An unnamed former staff member of Chinese drone maker DJI was fined $29,000 and given a six-month prison term for posting the company's code to GitHub, essentially putting it in the public domain. The former employee worked as a programmer, developing code for the DJI's agricultural drone management platform and the embedded system for agricultural machinery. The programmer created an account on GitHub and uploaded the code to a public repository. When the company learned of the leak, the employee came forward.[27]

Why is security education important?
Companies who are focused on protecting source code should not consider developers to be the enemy. Instead, they should educate the development staff as to the legal status of source code, what activities are allowed, and the specific organizational policies that protect source code to head off problems before they arise. Programmers who have been sued by companies or prosecuted by authorities for source code theft have claimed in their defense that they did not believe they were doing anything wrong. Security education and informed employees can prevent any misunderstandings that could lead to an incident.

27. Osborne, Charlie. "DJI employee who leaked source code sent behind bars." ZDNet. News Article. 30 April 2019.

Augmented threat detection with behavioral analytics will give your security operations teams an edge against attackers and enable them to focus on finding source code leaks before the damage is done. Using the two technologies together can help reduce the deluge of alerts that affect security operations, improving time to detection and reducing the mind-numbing efforts at triage that can reduce the effectiveness of security analysts.

Organizations that adopt user entity and behavioral analytics (UEBA) will have greater visibility into the activity that puts their source code and intellectual property at risk. With nearly every industry relying on software for a competitive advantage, visibility into activities that threaten that competitiveness is critical.

In the past decade, a variety of organizations have faced a leak that exposed their source code to criminals or rivals.

Gaming and Entertainment Firms

In February 2021, cybercriminals stole source code from CD Projekt Red, a small Polish games developer known for the Witcher series. The source code theft came after the company had struggled to tame bugs that made its latest game, Cyberpunk 2077, an Internet meme. The ransomware gang who claimed to have stolen the source code auctioned off the intellectual property within a few days. The price could have been between $1 million and $7 million and the terms required that the gang not resell the data.[28]

Four months later, video gaming giant Electronic Arts suffered a similar attack when a ransomware gang claimed to have stolen 780 gigabytes of the company's source code. The attackers offered the data for sale online, but apparently no one wanted to take the risk of buying it. A month later, the attackers attempted to extort money from Electronic Arts to not release the source code.[29]

Security Firms

Security firms are not invulnerable to hackers. In 2017, system management software maker Piriform—freshly acquired by security provider Avast—announced that attackers with access to the company's source code had modified a software update to provide a back door into users' systems after installation. This was not only a source code attack, but also a major attack on the supply chain, with the code modification affecting more than 2.27 million users.[30]

In 2019, network management vendor SolarWinds suffered an intrusion that led to the company's source code being compromised. The first inklings of the massive attack came after security firm FireEye discovered that attackers had used access to the SolarWinds Orion source code to create a patch that bypassed security measures and infected SolarWinds' customers, among them FireEye. The attacker, thought to be linked to Russia, stole the source code for the company's products.[31]

28. Abrams, Lawrence. "CD Projekt's stolen source code allegedly sold by ransomware gang." Bleeping Computer. News Article. 13 February 2021.
29. McNulty, Thomas. "EA Hackers Are Releasing Stolen Data To Extort Ransom Pay." Screenrant. News Article. 14 July 2021.
30. Lemos, Robert. "CCleaner Attack Shows Need to Bolster Software Development Security." *eWEEK*. News Article. 20 September 2017.
31. Sanger, David E. and Perlroth, Nicole. "FireEye, a Top Cybersecurity Firm, Says It Was Hacked by a Nation-State." *The New York Times*. New Article. 8 December 2020. Updated: 6 February 2021.

Technology Firms

Industrial espionage by other nations frequently targets source code. An unnamed manufacturer of locomotives based in Chicago, Illinois had its source code stolen by an engineer in 2014, two weeks after he started working there. The intellectual property theft involved not just source code for the locomotive operating system, but also specifications outlining how to use the code and the system. The theft continued for six months, until the engineer was fired for unrelated causes, according to a federal indictment unsealed in December 2017.[32]

One of the most significant attacks involved the theft of Google's source code and infiltration into the networks of more than 20 companies by Chinese hackers, as part of what became known as Operation Aurora. The attack, which gained access to the code for Google's search engine, also claimed a database that had information on US surveillance targets. At the time, China already had a reputation for stealing "massive volumes of data from companies in sectors including defense, technology, aerospace, and oil and gas."[33]

Conclusion

The steps an attacker takes to reconnoiter your network and users tailor attacks to bypass your defenses, create payloads specifically designed for your environment, and then exfiltrate data form an attack chain. While some defensive techniques focus on specific parts of the chain (such as firewalls preventing untrusted connections or anti-malware assessing executable programs), behavioral analytics detects any anomalous activity and thus delivers defense in depth against every step in the attack chain.

32. Reuters Staff. "U.S. charges ex-Illinois software engineer with taking stolen trade secrets to China." Reuters. News Article. 11 July 2019.
33. Nakashima, Ellen. "Chinese hackers who breached Google gained access to sensitive data, U.S. officials say." *The Washington Post*. News Article. 20 May 2013.

The mistakes made by an inattentive insider are likewise not isolated events. Insiders who misconfigure software or infrastructure have likely made similar errors in the past. Behavioral analytics systems can detect those issues because the anomalous behavior will trigger warnings.

Applying defenses against the attacker—or double-checking insider activity—at every step is important because an intruder who is after source code might be an authorized user or an unauthorized external attacker. In the past, almost every instance of source code theft detected by the organization happened when an insider or attacker moved the code outside the network. In most cases, the culprit was not caught until days or weeks after the source code had been copied. By focusing on the user's actions at every potential stage of the attack chain, organizations have a better chance of catching threats that target source code and stopping the transfer of data outside the organization's control.

As every company undergoes its own digital transformation, security operations teams need to protect the company's core business assets. Because every business has become a tech company with valuable source code, it is even more imperative that they protect their software to retain their strategic advantages.

Ready to Secure Your Source Code?

Want to find out how OpenText™ handles source code protection? Visit **cyberres.com/arcsight-intelligence** to learn more or **cyberres.com/ai-demo** to request a demo of our capabilities.

Connect with Us

www.opentext.com

opentext™ | Cybersecurity

OpenText Cybersecurity provides comprehensive security solutions for companies and partners of all sizes. From prevention, detection and response to recovery, investigation and compliance, our unified end-to-end platform helps customers build cyber resilience via a holistic security portfolio. Powered by actionable insights from our real-time and contextual threat intelligence, OpenText Cybersecurity customers benefit from high efficacy products, a compliant experience and simplified security to help manage business risk.