hackerone | Veterans United Home Loans

CASE STUDY

The Power of HackerOne's API Automation: A Q+A With Veterans United

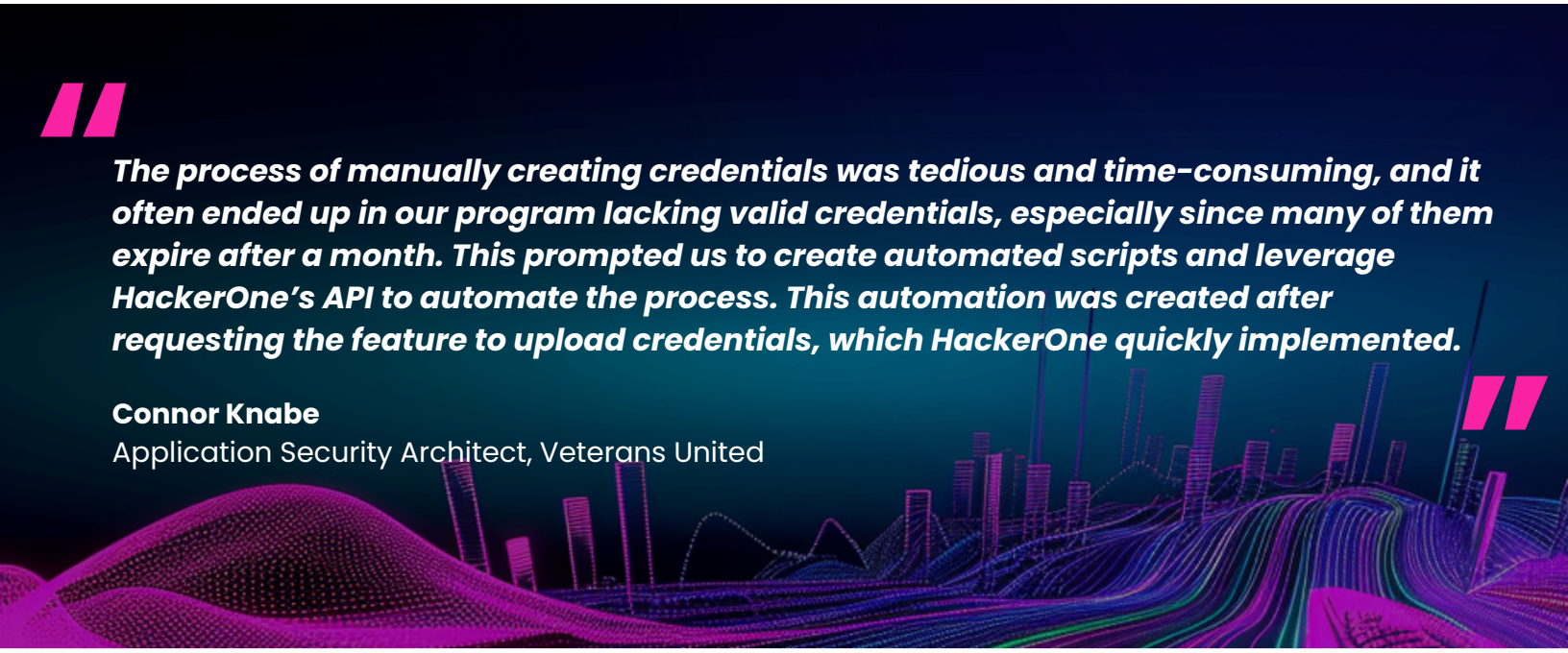
Every business sector faces sophisticated security threats—but in the highly regulated financial services industry, the stakes are especially high. Mortgage company Veterans United is taking a forward-thinking approach by automating key aspects of its security testing program. We spoke with Veterans United application security team member Connor Knabe to learn about the organization's journey toward automated credential management using the HackerOne API—and how this process has boosted security researcher engagement with the Veterans United bug bounty program.

Read this Q&A to learn how this innovative approach has transformed their security testing workflow, significantly improving efficiency and accuracy while reducing manual overhead.

Q: Please introduce yourself. Tell us what you do at Veterans United and why cybersecurity is important to Veterans United.
A: My name is Connor Knabe. I work on the application security team at Veterans United, a mortgage company that helps veteran heroes with the home-buying process. Our application security team works with our software engineers to help build secure applications that protect our customers' data.

Q: What types of security testing do you focus on?
A: One of our biggest focuses on the application security team is security testing internally developed public-facing applications. We work on discovering vulnerabilities that put our customer's data at risk. Pentesting and bug bounty are two of the primary forms of security testing that we do. However, other security tools, such as dynamic application security testing, code reviews, and SBOM, are shifted to the left.

Q: How are you currently managing credentials for authenticated testing? What were you using before that made you want to do something more automated?
A: For almost all of our assets, we have automated scripts written in Postman that create the credentials and stage them properly by calling various internally built APIs and then uploading them to HackerOne. Before that, we would have to work with QAs on various teams, who would then create and stage accounts and get that data back to us. The process of manually creating credentials was tedious and time-consuming, and it often ended up in our program lacking valid credentials, especially since many of them expire after a month. This prompted us to create automated scripts and leverage HackerOne's API to automate the process. This automation was created after requesting the feature to upload credentials, which HackerOne quickly implemented.



Q: Can you describe your automated credential management process?

A: We have tested a few public-facing assets using HackerOne. MyVU is our most significant web application, and it is a portal for borrowers to upload documents, complete tasks, and interact with their loan team throughout the loan process. This application is dynamic, and different features are exposed at various times throughout the loan process. Initially, we collect information to prequalify the borrower; later, we have them upload documents, and later, they might pay for their loan. To test these various features, we needed automation to call our internal APIs and stage one specific account for a hacker to test a feature such as disclosure signing. We have API calls written in Postman to set up various parts of the loan. The Postman script goes out to various APIs in order and sets up data on the account. It has various features such as the number of accounts per HackerOne "credential," revoking old creds after a certain length of time, the number of accounts to create in HackerOne, and JSON with the configuration for easy modification.

Q: What benefits have you seen from automating this process?

A: One of the main benefits we have partially attributed to this process is more findings. We think this results from having credentials available that allow hackers to access more of our website's features, which before were time-limited due to credentials being so "costly" to create. Another benefit would be more hacker engagement, which we think results from needing to spend less time creating and coordinating credential creation, allowing us to interact with the hackers more frequently and spend more time responding to their findings.

> ***One of the main benefits of this process is more findings. This results from having credentials available that allow hackers to access more of our website's features, which before were time-limited due to credentials being so "costly" to create.***
>
> **Connor Knabe**
> Application Security Architect, Veterans United

Q: How does this approach compare to manual credential management?

A: Manual credential management is tedious and creates gaps in testing due to creds not always being in HackerOne. Keeping the creds current was nearly impossible when we had a manual process. There were a few reasons for this, one of which was that many parties needed to handle the manual creation, including the business, namely QA, and getting buy-in from the business to allocate time for these endeavors. Since creating credentials manually is very time-consuming, we believe this left gaps in possible vulnerabilities that could have been found if we had always had fresh credentials. This was especially apparent when new researchers were joining our program and actively testing. If the credentials aren't there when the hacker tests, it's less likely they will test an asset when the credentials are added later.

Q: What advice would you give other organizations considering API automation for credential management?

A: API automation for HackerOne testing is worth the up-front investment and maintenance. It's worth noting that this may require heavy collaboration with software engineering teams since there may not be APIs already created to stage accounts, and you may need to duct tape some things together and think outside the box to create automation in ways that might be brittle and need some level of maintenance.

Despite implementing automations mostly on our own with minimal assistance from software engineering teams, we are grateful and happy to maintain them when things break (i.e., an API changes or stops working) due to the value they add to our HackerOne program. A side benefit of these automations is they can help keep your systems resilient as they may be leveraging APIs in ways that the software engineering teams haven't, and when they break, the teams will need to be informed.

We'd recommend testing against a UAT (user acceptance test) environment and not prod since automations can break and might cause issues. You don't want to ruin trust tokens with the engineering teams, especially if you don't already have a strong partnership with them. Having someone on the security team with an engineering background is beneficial for building these automations. It's worth noting that these automations were built on the security side of the house and were not a request of software engineering.

Q: How has this automation impacted your overall security testing program?

A: It has made it easier and more efficient and helped build confidence in HackerOne's bug bounty program. Before the automation, we weren't extremely happy with the number of findings we were getting via bug bounty, but now we wouldn't imagine going without a bounty program.

Q: Have you faced any challenges implementing this automated process?

A: One of the challenges that wasn't expected is that we blew through our bounty budget the first year we implemented some of these automations. This was a good problem to have, but it did result in our program needing to be paused for some of the year until our contract was renewed. When we resumed, the program kicked back up without issues and has continued producing value for us.

Another challenge related to implementing the automations by needing to understand the business processes and how things were connected programmatically, which did require collaboration with the software engineering team.

"API automation for HackerOne testing is worth the up-front investment and maintenance. Before the automation, we weren't extremely happy with the number of findings we were getting via bug bounty, but now we wouldn't imagine going without a bounty program."

Connor Knabe
Application Security Architect, Veterans United

hackerone | Veterans United Home Loans

With over 2,000 customer programs, more companies trust HackerOne than any other vendor

gm · Lufthansa · ZEBRA · zoom · Spotify · citrix · PayPal · Uber · HYATT · U.S. Department of Defense · Google · reddit · Nintendo · Adobe · A.S. Watson Group · coinbase · yahoo! · priceline · shopify · slack · yelp · salesforce · TOYOTA

Contact us

About HackerOne

HackerOne pinpoints the most critical security flaws across an organization's attack surface with continual adversarial testing to outmatch cybercriminals. HackerOne's Attack Resistance Platform blends the security expertise of ethical hackers with asset discovery, continuous assessment, and process enhancement to reduce threat exposure and empower organizations to transform their businesses with confidence. Customers include Citrix, Coinbase, Costa Coffee, General Motors, GitHub, Goldman Sachs, Hyatt, Microsoft, PayPal, Singapore's Ministry of Defense, Slack, the U.S. Department of Defense, and Yahoo. In 2023, HackerOne was named a Best Workplace for Innovators by Fast Company.