The Best Write-ups that 2019 Brought Us

BY INTIGRITI · DECEMBER 30, 2019 · LAST UPDATED ON MARCH 6, 2025

Want to read the latest version? Visit the link below:
https://www.intigriti.com/researchers/blog/hacking-tools/the-best-write-ups-that-2019-brought-us

With 2020 just a days away, it is time to look back and appreciate the good stuff last year brought us. So in case you're stuck on a boring Holiday party: now is the time to sneak out and take a moment and revisit the top ten best write-ups of 2019.

10) Hacking GitHub with Unicode's dotless 'i'.

Link: https://eng.getwisdom.io/hacking-github-with-unicode-dotless-i/
Author: @wisdom_devs

Your application can support as many languages you like: in the end, your webserver will only process 1's and 0's. Normalization is everywhere and can attribute to situational and unique security issues, which makes them so interesting. In this example, the author was able to gain partial account takeover in GitHub by using a single character. Even though GitHub's outgoing mail server partially mitigated a broader exploitation of this attack, it is a novel and creative entry that indicates we will see more unicode normalization issues in the future.

9) Abusing autoresponders and email bounces

Link: https://medium.com/intigriti/abusing-autoresponders-and-email-bounces-9b1995eb53c2
Author: @securinti

In 2019, we've seen a couple of new attack vectors, but we never expected someone to come up with a new technique in a technology that's been around for 50 years: e-mail!

Revising old concepts in newer implementations is a great strategy for finding a lot of valid bugs, but it requires new thinking and discovering something that few people might have noticed. So it is nice to read about @securinti's thought process!

This article encompasses a lot of information like:

- How to find valid target email addresses without spamming them
- Examples of how to exploit them for many different attacks (blind XSS, arbitrary file upload, Ticket Trick, abusing printers...)
- How to abuse autoresponder and bounce emails to obtain sensitive information (like someone's real email address behind a generic one)
- Two examples of such bugs found on Google and Intigriti

What was interesting about this write-up is that, whilst the initial attack surface was limited, it lead to some great follow-up discoveries on private programs throughout 2019, sometimes even leading to account takeover without user interaction.

8) Facebook Messenger server random memory exposure through corrupted GIF image

Link: https://www.vulnano.com/2019/03/facebook-messenger-server-random-memory.html
Author: @xdzmitry

This was a weird bug in Facebook Messenger for Android: @vulnano uploaded a corrupted GIF file with missing content body. The image displayed back contained data from previously used memory buffers. It was leaking data from memory! He noticed it because the image display had white noise, while it was supposed to be blank.

Also, when the images were uploaded with Facebook Messenger for Android, nothing happened. The weird images were only visible from the Facebook Web app.

Another takeaway is to not rely on tools without understanding what they do and how to do the same job yourself. @vulnano first generated corrupted images with Gifoeb, but they caused the app to crash. So he studied the GIF image format and generated his own images.

7) Cracking my windshield and earning $10,000 on the Tesla Bug Bounty Program

Link: https://samcurry.net/cracking-my-windshield-and-earning-10000-on-the-tesla-bug-bounty-program/
Author: @samwcyo

Some bugs came in unexpected ways. In 2019 @samwcyo was unfortunate enough to crack his windshield, but then accidentally triggered a blind XSS in the repair process, yielding him a $10,000 bounty.

Takeways we'll carry with us in 2020:

- – Put blind XSS payloads everywhere
- – Own a Tesla one day
- – Then damage it intentionally to find new bugs

6) A Tale of Exploitation in Spreadsheet File Conversions

Link: https://buer.haus/2019/10/18/a-tale-of-exploitation-in-spreadsheet-file-conversions/
Authors: @bbuerhaus, @daeken, @erbbysam, @smiegles

Do you remember this awesome video snippet with @daeken where he was clapping because obviously some kind of exploit or bug worked? It turns out that he was working on a Ghostscript payload in LibreOffice, in collaboration with @bbuerhaus, @smiegles, and @erbbysam.

It did work, and this is the writeup of the whole research that led to that bug. It touches on many topics: Ghostscript, fingerprinting LibreOffice, LFD, SSRF... This is worth reading and a great example of research in Web app security.

5) Abusing HTTP hop-by-hop request headers

Link: https://nathandavison.com/blog/abusing-http-hop-by-hop-request-headers
Author: @nj_dav

This was some cool research on hop-by-hop headers. These are headers that are used by proxies and not forwarded to the end server.

@nj_dav discovered a way to abuse them and basically remove other request headers. This can have unexpected results like authentication bypass, Cache poisoning DoS, etc.

The premise is simple to understand, but it would be interesting to practice this attack and take the research further by testing on common WAFs.

4) Bypassing GitHub's OAuth flow

Link: https://blog.teddykatz.com/2019/11/05/github-oauth-bypass.html
Author: @not_aardvark

Who would have thought that playing with HTTP methods could bypass OAuth on GitHub and yield a $25,000 bounty?!

The bug exists because the same controller handles both GET & POST requests, and using a HEAD request instead is unexpected.

The controller relies on the HTTP method to determine whether it will grant access to the app or serve an OAuth authorization page. @not_aardvark used the HEAD method. It was routed as GET (Rails behavior) and at the same time, the controller treated it as an authenticated POST request, bypassing authorization.

3) Filling in the Blanks: Exploiting Null Byte Buffer Overflow for a $40,000 Bounty

Link: https://samcurry.net/filling-in-the-blanks-exploiting-null-byte-buffer-overflow-for-a-40000-bounty

Author: @samwyco

Just in time for the new year, @samwyco treated us with a write-up that blew our mind! A lot of security researchers try to set themselves goals during bug hunting and Sam was no exception: he decided to focus on re-registering existing usernames. He tried adding special characters (like null byte, CRLF characters, spaces, Unicode…) hoping that they would be removed during the registration process.

The vulnerability is that each null byte inserted was replaced with random data, e.g.:

- Request: *POST /register?username=victim%00@domain.com*
- Response: *username victimIdL@domain.com*

So, injecting multiple null bytes (*victim%00%00%00@domain.com*) made the server return chunks of memory that contained very sensitive data (SSH keys, passwords, usernames, etc).

2) Owning The Clout Through SSRF

Link: https://www.youtube.com/watch?v=o-tL9ULF0KI
Author: @nahamsec, @daeken

Just when we thought we'd seen every single SSRF technique, @NahamSec and @Daeken reinvented this attack vector by throwing in DNS rebinding, causing a wave of hotfixes and bug bounty submissions in popular PDF generators that were previously deemed secure.

1) HTTP Desync Attacks: Request Smuggling Reborn

Link: https://portswigger.net/research/http-desync-attacks-request-smuggling-reborn
Author: @albinowax

James Kettle, better known as @albinowax, breaks the internet nearly every year and 2019 was no exception. Last summer, James gave one of the most bespoken presentations at Black Hat and Defcon, discussing a HTTP Desync Attacks, a technique that could allow attackers to harvest plaintext passwords. We're happy that people like James use their skills to do good, and think the more than $70K in bounties he was able to collect with this attack were more than well-deserved. No pressure or anything, but we look forward to what next year will bring, @albinowax!

The blog articles referenced above were previously curated by @pentesterland for our weekly #BugBytes newsletter.

REQUEST A DEMO

intigriti.com/demo

VISIT THE WEBSITE

intigriti.com

GET IN TOUCH

hello@intigriti.com