Imagery

I ran a quick Nmap and found a web app on port 8000. The site had registration/login pages, image upload/transform features, and a bug-report form. I chained a stored XSS in the bug report to steal an admin session, used that access to hit an admin log downloader that was vulnerable to LFI, pulled the app database (db.json), cracked the hashes to own a web user, and then exploited an unsafe ImageMagick invocation to get RCE as the web user. From there I found an encrypted backup, cracked it offline to recover another account, and finally abused an automated job system to get root.

Steps I took

Recon
Did a service scan and basic app enumeration. Found the image upload/transform flow and a bug-report feature that accepts HTML.

Stored XSS → admin cookie steal
I crafted a small stored XSS payload in the bug form that exfiltrated cookies back to my listener. That snagged an admin session cookie and gave me access to the admin interface.

Admin LFI → db.json
Using the admin session, I abused the log-download parameter to read arbitrary files. I pulled db.json and got user records and password hashes — and a view into the code that called ImageMagick unsafely.

Cracked hashes → web user
I extracted the hashes and cracked them offline (rockyou). That let me log in as a normal web user.

ImageMagick injection → RCE
The image transform endpoint built shell commands from user input and ran them directly. I injected a payload into the ImageMagick arguments to spawn a shell and got a reverse connection as the web user.

Backup recovery → new creds
I found an encrypted backup on the host, downloaded it, and cracked it offline. That revealed another account I could use to pivot.

Pivot to mark and escalate
With the recovered credentials I moved to the next user, grabbed the user flag, and kept enumerating. I discovered an automated job manager that accepted commands and ran them with elevated privileges. I used the job system's features to get an interactive shell as root.

Root

Once I had root via the job system, the final flag was trivial:

cat /root/root.txt

Key takeaways

- Don’t ignore UI features: bug forms and image endpoints are common attack vectors.
- If the app spawns system tools (ImageMagick, ffmpeg) from user input, assume command injection is possible unless you’ve reviewed the code.
- Backups on the server are gold — grab them and check for secrets.
- Admin panels that expose file download or log viewing often lead to LFI if parameters aren’t sanitized.
- Automated job systems tend to run with high privileges — they’re great for escalation if you can control inputs.

— MalwareMusashi