|  |  |
| --- | --- |
| Machine Description | VIkings : 1 is a Linux boot-to-root CTF-style vulnerable VM released via VulnHub. It is intended for enumeration, exploitation, privilege escalation via containers (LXD) and misconfigurations. The path leads from web enumeration to user shell, and eventually a container breakout to root. |
| Target IP | 10.136.108.240 |
| Vulnerability Name | LXD Container Escape via Unrestricted Container Launch (mapped to a container breakout vulnerability) |
| Service / Version | TCP / 80 HTTP |
| CVSS Score / Severity | 9.8 Critical |
| Attack Vector | AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H |
| Proof of Concept | STEP 1 – Discover the Target IP using Netdiscover tooll    STEP 2 – Scan Target for open ports and services running using Nmap    STEP 3 – Enumerating HTTP Port 80     * We have site lets look what’s inside      * Let’s look for the directories and files on server      * We got war.txt here      * We have another subdirectory here      * Lets verify what’s there      * We have random text here which is base64 encoded lets save in file and decode it      * Looks some kind of data lets save in file and check format of data      * Let’s unzip it      * Needs to unzip the file and it has password and username     STEP 4 – The file password is ragnarok123 lets unzip it online     * We have one image file lets download it and get more info with exif tool      * We got one more hash value      * Hash-ID say’s its SHA-256      * Lets try binwalk and here we got some data      * Here is file user lets read the content      * We got valid creds here * User = floki * Pass = f@m0usboatbuilde7   STEP 5 – Let’s authenticate with SSH using creds we got    STEP 6 – Local enumeration  We can see that the user floki belongs to the group lxd and this directly gives us access to the root user.     * Here are two files to find path to another user Ragnar     From the file boat, we see that we have to find printable characters from collatz conjecture of a number. This is a very interesting conjecture and its domain is positive numbers. Since this hasn’t been proved or disproved (for all positive numbers), it is an unsolvable problem. Anyway, this works till the number 2 to the power 68. According to the conjecture, for a positive odd number n, we have to perform the next calculation as n = 3n + 1. Similarly, for a positive even number, it should be n = n / 2. Doing this iteration, we will reach number 1 and we can stop.   * So, here, we have to find the collatz conjecture of the 29th prime number i.e. 109. For this, I wrote a simple python script.      * Let’s run the script      * Decode it on Cyber Chef      * Got password : mR)|>^/Gky[gz=\.F#j5P(   STEP 7 – Authenticate using ssh     * Got flag here      * Turning into proper shell     STEP 8 – lets look for running process using netstat     * a misconfigured **RPyC Classic server** running as **root**, then executed Python code inside the root process to implant your SSH key and gain full root access.      * Ragnar writes a Python function that copies his key into /root/.ssh * Ragnar connects to root’s RPyC service and teleports the function      * We have created a key and also passphrase for authentication * Lets see the key and copy it on our attacker machine      * Copy the key to local system * Login with root user * Use key file for auth and * Enter passphrase as : yashroot     GOT ROOT ON TARGET |
| Impact | * Attacker obtains valid user credentials via hidden zipped archive & steganography embedded in the web app. * Using those credentials they log in as a regular user and gain shell access. * The user is part of the “lxd” group enabling container creation/execution. * The attacker launches a malicious container, escapes into host root, and completely compromises the system including root flag. |
| Remediation | * Remove weak or hard-coded credentials, disable unnecessary directory listings and zipped archives on publicly accessible web directories. * Enforce strong authentication, implement multi-factor where possible, restrict user group memberships (especially for virtualization/container tools). * Limit or disable LXD (or similar container systems) on production hosts unless strictly required; enforce container isolation, patch runc/container runtimes. * Regularly audit sudoers, group memberships, container privileges, and restrict root-level container capabilities. |
| Reference | <https://nvd.nist.gov/vuln/detail/CVE-2019-5736> |