Lecture 10 : Confused Deputy

Alexandre Bartel

2019

Previously...

Previously... in Lecture 1 (Introduction)

- Software Development Life-cycle
- Vulnerability Life-cycle
- Vulnerability Disclosure

Previously... in Lecture 2 (Buffer Overflow)

- A buffer on the stack
- Return address on the stack
- Overwrite return address
- Jump to shellcode on the stack

Previously... in Lecture 3 (ROP)

- NX bit (stack non-executable)
- Gadgets in already loaded code
- Chain gadgets (addresses of gadgets and data on the stack)
- Only data on the stack

Previously... in Lecture 4 (ASLR)

- Randomize code segment at program start
- Breaks gadget chains
- Bypass with information leak (e.g, vulnerability)

Previously... in Lecture 6 (CFI)

- Mecanism to allow only "intended" paths
- Binary instrumentation to add IDs
- Indirect jumps, call, returns check if ID of "destination" is correct
- Pure software implementation have 20% overhead

Previously... in Lecture 7 (Heap-Overflow)

- How a heap-overflow can be attacked depends on the heap management implementation
- The "unlink" attack present in early versions of glibc provides a "write anywhere" primitive to the attacker
- Recent implementations performs more check to prevent "unlink" based attacks

Previously... in Lecture 8 (Type Confusion)

- What is it? Manipulation of an object through another object.
- Consequences? Undefined behavior, hijack control flow.
- Why it works? No verification at runtime (otherwise runtime and/or memory overhead).

Previously... in Lecture 9 (SQL Injection)

- Code injection attacks enables bypass of authorization checks and/or execution of arbitrary code on the server
- Consequences: attacker gets access to privileged environment and/or can dump databases
- Protection include sanitization of the input and/or well defining what is code and what is data

Confused Deputy

Confused Deputy

- Confused: being unable to think with clarity or act with understanding and intelligence
- Deputy: An entity (person / software) empowered to act for another
- (specific type of privilege escalation vulnerability)

The vulnerability

- 1: Deputy software provides a service
- 2: Deputy software is not well configured/programmed
- 3: Attacker asks deputy software to use the service to its advantage
- 4: Deputy software says, "Sure, no problem. Let's do it."
- 5: Deputy software should have replied "No, sorry."

- ex 1: return GPS coordinates
- ex 2: do not check that caller has GPS permission
- ex 3: give me GPS coordinates
- ex 4: here are the GPS coordinates, thank you for using my service
- ex 5: Sorry, I cannot give you the GPS coordinates since you do not have the GPS permission

Illustration 1: "The Confused Deputy" [1] (1988)

- Compiler FORT installed in directory SYSX
- User would call (SYSX)FORT to run the compiler on source code to generate a binary
- User could specific target file to receive debugging information
- Compiler engineers designed it to output statistics information in (SYSX)STATS
- This requires the OS to know that FORT has the right to write in its home directory (SYSX)
- Thus FORT has been given the right to its home directory (SYSX)

[1] Hardy, Norm. "The Confused Deputy:(or why capabilities might have been invented)." ACM SIGOPS Operating Systems Review 22.4 (1988): 36-38.

Illustration 1: "The Confused Deputy" (1988)

- Billing information also saved to (SYSX)BILL...
- What could go wrong? $\rightarrow$ User can specify BILL as the output for debugging information...
- Problem: compiler runs with authority from 2 sources: the user and the compiler (can write to his home)
- Solution: new system call to "switch hats"
- The confused deputy was the compiler with the privilege to write to directories where normal users cannot

Illustration 2: Cross-Site Request Forgery (CSRF) [1]

- Tricks the victim into submitting a malicious request.
- Synonyms: XSRF, "Sea Surf", Session Riding, Cross-Site Reference Forgery and Hostile Linking.

[1] https://www.owasp.org/index.php/Cross-Site_Request_Forgery_(CSRF)

Illustration 2: Cross-Site Request Forgery (CSRF)

- User browses on his banking site
- Banking site accepts requests such as "https://bank.lu/requests.html?amount=10&to=Dave"
- Without closing the banking session, user visits other websites
- One of them, "https://www.malicious.lu" sends an html page with the following link "https://bank.lu/requests.html?amount=10&to=Attacker"
- If the user clicks the request is sent.
- The confused deputy is the web-browser which has the privilege to send requests to the bank

Illustration 2: Cross-Site Request Forgery (CSRF)

- Secret value in cookie?
- Only accepting POST requests?
- HTTPS?
- Add a hash to all forms?

Illustration 3: Cross-Site Scripting (XSS) [1]

- Store code on web-server side, "www.compromised.lu"
- For instance, if servers allows users to add comments but do not filer comments properly
- Malory could send "$< script >$ alert('hi there!'); $< /script >$" as a comment to page "www.compromised.lu/blog.html"
- This comment is stored in the database of web-server "www.compromised.lu"
- This comment is present in the html page generated by web-server when a user visits "www.compromised.lu/blog.html"
- Thus, all users visiting "www.compromised.lu/blog.html" will execute the javascript code of Malory: "$< script >$ alert('hi there!'); $< /script >$"

[1] https://www.owasp.org/index.php/Cross-Site_Request_Forgery_(CSRF)

Illustration 3: Cross-Site Scripting (XSS)

- The confused deputy is the web-server which has the privilege to generate html pages to clients
- Solution?
- Sanitize input/output

Illustration 4: Android Application as Confused Deputies

- A developer can program multiple Android applications A1, A2, A3
- A1, A2, A3 all have GPS permission
- A1 has code to retrieve GPS coordinates and share this information to A2 and A3
- A1 is used to share GPS coordinates to A2 and A3 through a service component
- Malory could write application MA (with no GPS permisison) and ask A1 for GPS permissions"

Illustration 4: Android Application as Confused Deputies

- The confused deputy is the Android application which has the privilege to retrieve GPS coordinates
- Solution?
- Restrict access to clients

Conclusion

- Confused Deputy Attack are a type of privilege escalation vulnerability
- Attacker exploits the associated vulnerability (misconfiguration, logic error) to have more privilege
- Protection include sanitization of the input and/or changing the configuration and/or patching the code logic

Question?

Projets

- Groups of 2
- Suggested topics:
  1. Heap exploitation on Debian 3.1
  2. Patch for CVE-2018-20343 (Ricardo, Alex)
  3. Complete exploit for CVE-2018-20343
  4. Study and PoC for CVE-2013-0912 (Chrome type confusion) (Adriano)
  5. Stable code injection through `/proc/self/mem`
  6. Explanation of a recent exploit targeting webbrowsers (Chrome, Firefox, etc.) (Yurii, Ervin)
  7. Exploitation of a PoC type confusion in C++ (Ihor, Artem)
  8. Break wordpress authentication mechanism.
- Deliverables: Presentation + Code (PoC)

Projet: Heap exploitation on Debian 3.1

- Explain the differences in the heap management code from debian 2.2 (lab 7) and debian 3.1
- Explain and develop a proof-of-concept to exploit a heap overflow on debian 3.1

Projet: Patch for CVE-2018-20343

- Understand CVE-2018-20343, a buffer overflow vulnerability
- You have to identify all instances of buffer overflow in the code (the code is not very big)
- You have to patch the vulnerable code

Projet: Complete exploit for CVE-2018-20343

- The current proof-of-concept only changes the value of EIP.
- You have to improve the PoC to enable an attacker to execute arbitrary code

Projet: Study and PoC for CVE-2013-0912 (Chrome type confusion)

- Reproduce the SVG code for the exploit based on information you find on the internet
- You should create a VM with a distribution from 2013 and have the vulnerable version of Chrome

Projet: Stable code injection through `/proc/self/mem`

- DosBox enables untrusted code to mount the host filesystem in the guest
- Thus untrusted code can write to `/proc/self/mem`
- You develop code to inject a shellcode into the virtual process of dosbox to execute arbitrary code
- You do this by writing to `/proc/self/mem`

Projet: Explanation of a recent exploit targeting webbrowsers (Chrome, Firefox, etc.)

- Contact me when you have found a CVE you want to explain.

Projet: Exploitation of a PoC type confusion in C++

- Write a proof-of-concept showing how to exploit a type confusion in C++ in a x86_64 architecture (latest debian)