

Unpredictable IDs and How to Use Them to Secure Your Application

Unpredictable IDs and How to Use Them to Secure Your Application

Stela Trencheva

Fontys University of Applied Sciences

Class: S3-CB01

Student number: 4095200

Table of Contents

<i>Table of Contents</i>	2
Problem definition	3
How BudgetSimple application number IDs can be easily used for hacker attacks and how to replace them with unpredictable IDs for a better security of information?	4
What is IDOR vulnerability and how hackers use it?	4
How to prevent IDOR vulnerability?	7
What are the best solutions for creation of unpredictable IDs?	10
Conclusion	12
References	13
Tables	14

Problem definition

This research is conducted in order to find a solution for the currently used insecure IDs in the URLs of the application BudgetSimple. The IDs that the application generates and uses at the moment are auto-incremented and can be easily guessed. That makes the application and information of users/wallets/etc. prompt to hacker attacks. In this paper, I am going to look at possible ways to create and use unpredictable IDs and apply them in the project so that the information in the application is secured from hackers and customers will be safe using the application.

Research questions

- How BudgetSimple application number IDs can be easily used for hacker attacks and how to replace them with unpredictable IDs for a better security of information?
 - What is IDOR vulnerability and how hackers use it?
 - How to prevent IDOR vulnerability?
 - What are the best solutions for creation of unpredictable IDs?

The three levels of the Development Oriented Triangulation Framework are used for defining the main questions of this research.

How BudgetSimple application number IDs can be easily used for hacker attacks and how to replace them with unpredictable IDs for a better security of information?

What is IDOR vulnerability and how hackers use it?

The insecure direct object reference (IDOR) simply represents the flaws in the system design without the full protection mechanism for the sensitive system resources or data. It basically occurs when the web application developer provides direct access to objects in accordance with the user input. So any attacker can exploit this web vulnerability and gain access to privileged information by bypassing the authorization. In other words, it involves replacing the entity name with a different value without the user's authorization. As a result, users will be directed to links, pages, or sites other than the ones they intended to visit. IDOR may not be a direct security threat, but it creates a conducive environment for hackers to access unauthorized data. The Open Web Application Security Project (OWASP) has a list of top 10 web application vulnerabilities and IDOR is one of them.

IDOR has been regarded as a serious web application vulnerability. Unlike other vulnerabilities such as SQL-Injection, identifying IDOR vulnerability is a bit challenging using automated tools. This is because to successfully attack using this flaw, we need to distinguish the flawed interface as well as the pattern to spot an insecure object. In order to identify an interface that provides access to sensitive contents, code review and website walk-through must be done ^[1].

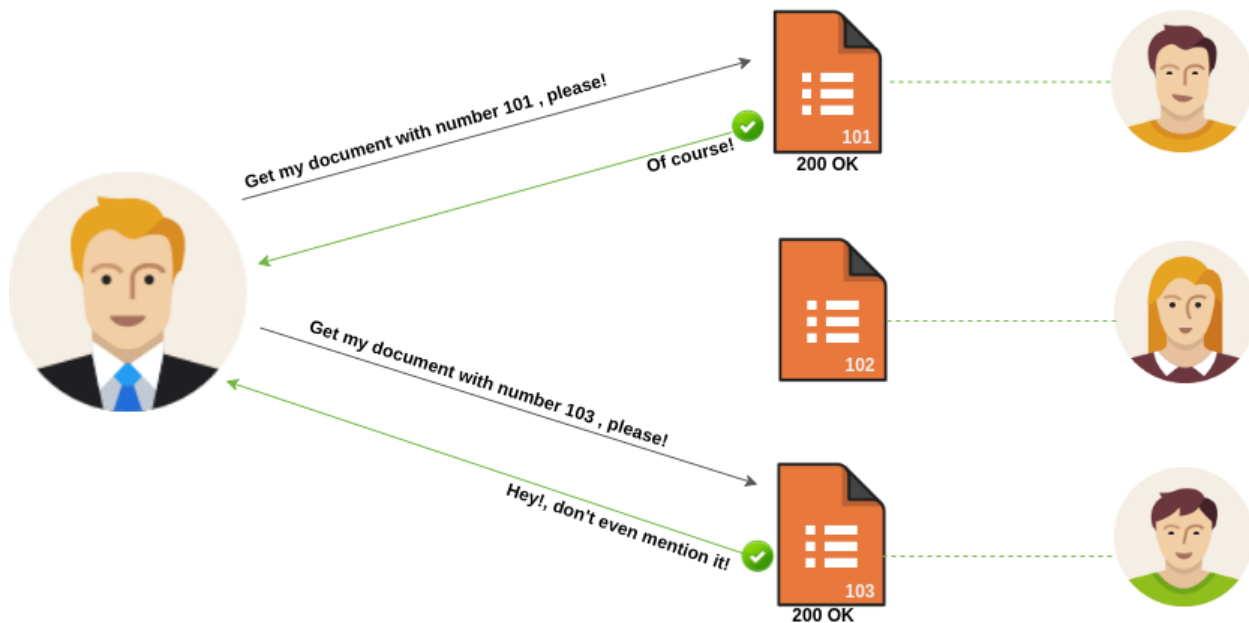


Figure 1: Example of IDOR vulnerability

IDOR has a major role in large security breaches worldwide. The following are only a few cases:

- US Security Researcher discovered IDOR case regarding Samsung ^[2].
- More than 1.5 Million records from Yahoo database were deleted by one Egyptian hacker using IDOR ^[3].
- Turkish security researcher hacked Apple Developer's site using IDOR and revealed 275000 registered third-party developers ^[4].

The most common attack vectors for IDOR vulnerability are:

- Missing access level control
 - In OWASP Top 10 2017, Missing Function Level Access Control and IDOR are merged together and called Broken Access Control^[5]. Missing function level access control is much of an authorization issue where the application just checks for the user being authorized or not.

- Parameter modification
 - This attack is based on the manipulation of parameters exchanged between client and server in order to modify application data, such as user credentials and permissions. Usually, this information is stored in cookies, hidden form fields. Most of the IDOR attacks are Parameter modification attacks.
- SessionID prediction
 - The session prediction is defined as an attack which focuses on predicting SessionID values that permit an attacker to bypass the authentication schema of an application. By analyzing and understanding the SessionID generation process, an attacker can predict a valid SessionID value and get access to the application

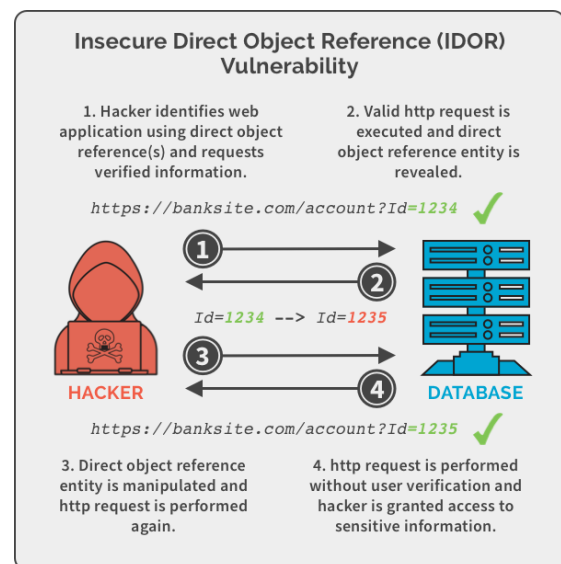


Figure 2: ID prediction

[6]. If the SessionID is predictable, a user uses different techniques to gain unauthorized access or to bypass authentication. This is the case in the application of BudgetSimple. Currently, it is using such predictable IDs and hackers can easily perform an IDOR ID prediction attack.

How to prevent IDOR vulnerability?

There are many best practices techniques to prevent IDOR vulnerability but few of the most important and popular are:

- The use of indirect object reference helps to prevent from getting access to unauthorized resources. An Indirect Reference Map is an alternative design method to 'Direct Object Reference' that helps businesses avoid IDOR vulnerabilities. It replaces the actual references (such as user IDs, names, keys, etc.) with alternate IDs that map to the original values. The mapping between the alternate IDs and actual references are maintained safely on the servers^[7].
- For each requested record, make sure that the user has the right privilege. Servers fail to identify tampered URLs because there are no access checks in place at the data-object level. Data layer access controls should be enforced only when the server verifies whether the current user owns or has access permissions to the requested data. At the very least, the application should perform a Syntactic Validation to verify suspicious inputs. The application should establish criteria for incoming input, and if it doesn't meet expectations, reject the value^[7].
- Encrypt legitimate values to make it more difficult for an attacker to predict the pattern and find other possible keys.

Using an automated tool to do these security things might be difficult. However, there are some free security scanning tools which are used to scan for web application security vulnerabilities.

- Sonarqube

SonarQube is developed by SonarSource and it is used as an open source platform for continuous inspection of code quality to perform automatic reviews with static analysis of

code to detect bugs, code smells, and security vulnerabilities on more than twenty programming languages^[8]

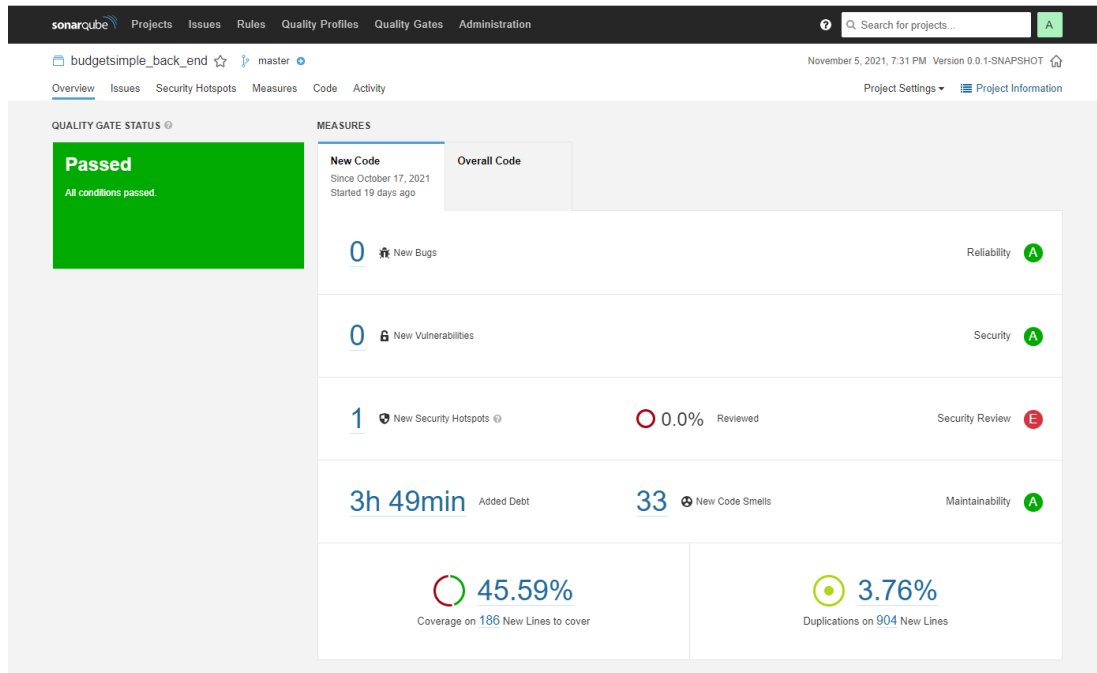
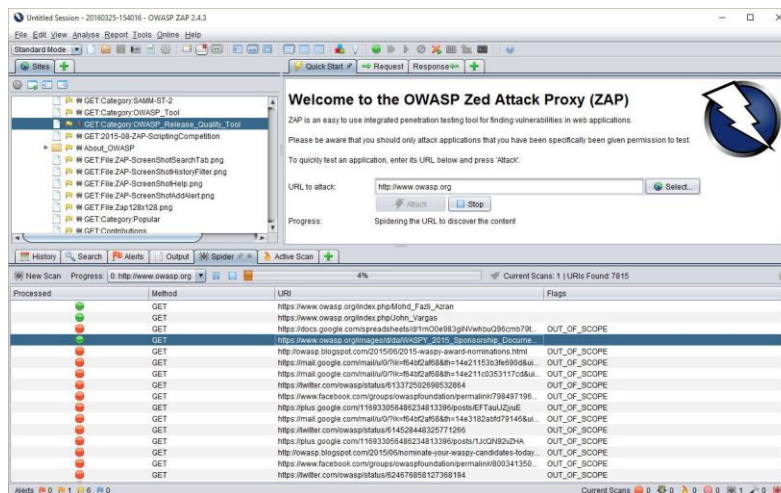


Figure 3: Sonarqube

- OWASP ZAP

Zed attack proxy (ZAP) is “an easy to use integrated penetration testing tool for finding vulnerabilities in web applications. It is designed to be used by people with a wide range



of security experience and as such is ideal for developers and functional testers who are new to penetration testing”^[9]. In OWASP ZAP spider scanning is used to scan

Figure 4: OWASP ZAP

the whole system automatically.

- Checkmarx

Checkmarx is “a provider of state-of-the-art application security solutions: static code analysis software, seamlessly integrated into development process”^[10]. This tool is one of the most expensive tool which costs approximately \$15,000 a year per license^[10].



Figure 5: Checkmarx

What are the best solutions for creation of unpredictable IDs?

In order to prevent Insecure Direct Object References, it is very essential to minimize user ability to predict object IDs/Names by using the hard-to-guess numbers.

Universal Unique Identifiers is a well-known concept that's been used in software for years. A UUID is a 128-bit number that, when generated in a controlled and standardised way, provides an extremely large keyspace, virtually eliminating the possibility of collision. A UUID is a synthetic ID comprising of several distinct parts, such as time, the node's MAC address, or an MD5 hashed namespace.^[11]

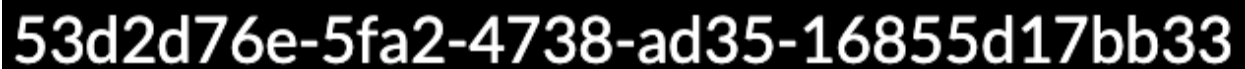


Figure 6: UUID

Snowflake IDs were developed by Twitter in June 2010 and are later used, with modifications, by companies like Discord or Instagram^[12]. Twitter was looking for a minimum of 10000 IDs per second per process with a response rate of <2ms. ID servers should require no network coordination at all between them and generated IDs should be, roughly, time-ordered. Finally, to keep storage to a minimum, IDs had to be compact. The generated IDs in the project are composed of:

- Time — 41 bits (millisecond precision)
- Configured machine ID — 10 bits
- Sequence number — 12 bits — rolls over every 4096 per machine

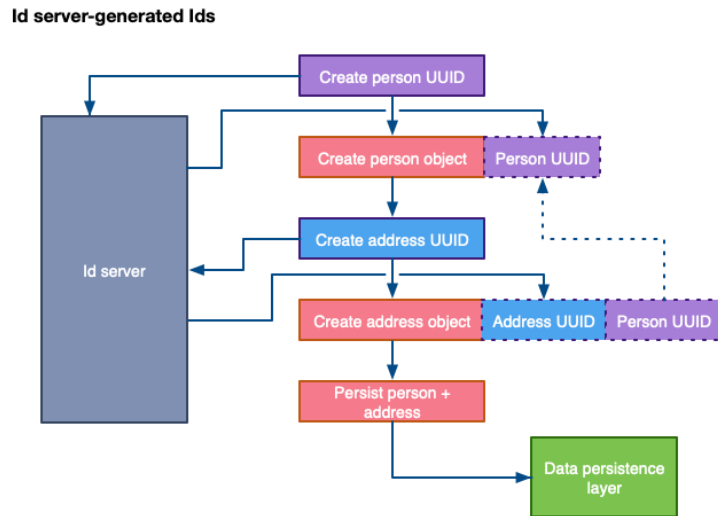


Figure 7: Server IDs

Since then Twitter upgraded their Snowflake project and they developed TwitterServer. An ID server takes care of generating unique IDs for your distributed infrastructure. According to the implementation of your ID server, it can be a single server creating IDs or a cluster of servers creating high numbers of IDs per second.^[11]

There is also an open source mapping interface called “**AccessReferenceMap**” class in OWASP ESAPI. The object of this class is used to create and store the dictionary in the session, and enables to add or delete the items from the dictionary, and to translate the IDs into the GUID and vice-versa. Since the object ID may be anything like file name or an internal user ID, Indirect Reference Map always maintains a non-sequential & random identifier for a server side resource. Therefore the end user may only see the alternate ID (GUIDs) but not the actual object’s ID. This sort of mapping can either be stored temporarily in server cache or permanently in a mapping table. What basically happens in this approach is that when the user tries to retrieve the information from the IDs, the HTTP Post, for example in the AJAX call, contains the alternate ID (indirect reference GUID), which is mapped to the ID value before retrieving any data and return it to the caller. If the call was an attack then the service might not be able to access the mapped value from the AccessReferenceMap. Thus the genuine calls and the attacks can be distinguished.^[13]

Conclusion

Insecure direct object references vulnerability is one of the owasp Top 10 vulnerability lists since 2004. Although IDOR is not a new vulnerability, the impact of this vulnerability is significant when it occurs. Existing automated security tools are not complete solutions to find IDOR vulnerability. IDOR vulnerability exist in different web application unless the systems are manually tested. The researched methods to prevent IDOR vulnerability and unpredictable IDs will be applied to the application BudgetSimple so that it is more secured and safe for customers to use it.

References

- [1] - <https://www.researchgate.net/publication/275043190>
- [2] - <https://hackernoon.com/samsung-leaking-customerinformation-b7e2dcb006d#.vvmbm3mkf>
- [3] - <http://securityaffairs.co/wordpress/22687/hacking/critical-flawyahoo.html>
- [4] - <https://www.theguardian.com/technology/2013/jul/22/apple-developer-site-hacked>
- [5] - <https://www.netsparker.com/blog/web-security/owasp-top-10/>
- [6] - https://www.owasp.org/index.php/Session_Prediction
- [7] - <https://spanning.com/blog/insecure-direct-object-reference-web-based-application-security-part-6/>
- [8] - www.sonarqube.org/about/
- [9] - https://www.owasp.org/index.php/Appendix_A:_Testing_Tools
- [10] - www.checkmarx.com/products/static-application-security-testing/
- [11] - <https://betterprogramming.pub/uuid-generation-snowflake-identifiers-unique-2aed8b1771bc>
- [12] - https://en.wikipedia.org/wiki/Snowflake_ID

Tables

Acronyms

Acronym	Definition
ID	Identification
URL	Uniform Resource Locator
IDOR	Insecure Direct Object Reference
OWASP	Open Web Application Security Project
SQL	Structured query language
ZAP	Zed attack proxy
ESAPI	The OWASP Enterprise Security API
UUID	Universal Unique Identifiers
GUID	Globally Unique Identifier
AJAX	Asynchronous JavaScript and XML
HTTP	Hypertext Transfer Protocol
MAC	Media Access Control
MD5	Message Digest 5