XXE Injection

Overview

An XML external entity (XXE) injection is a type of website security vulnerability that allows an attacker to interfere with how a web application processes XML data. Through an XXE injection, an attacker is potentially able to gain access to an application's server and files on its back-end system and carry out a number of exploits; from executing a remote request from the server to carrying out a Denial-Of-Service (DOS) attack.

XML (eXtensible Markup Language) is primarily used to store and transport data. XML stores data in a plain text format allowing for a software- and hardware-independent means of storing, transporting and sharing data. It is often used by web applications to format data prior to transport between the server and the browser. XML is similar to HTML, but unlike HTML, XML has no predefined tags - the programmer needs to define both the tags and the document structure that describe the data. The XML document type definition (DTD) outlines this document structure and contains declarations which define the structure of the particular XML document. An XML external entity (XXE) is a type of custom XML entity whose definition is located outside of the DTD. XXEs are the primary instruments for carrying out XXE attacks.

XXE Attack Via File Upload

Some websites allow users to upload their own files or images which in turn become processed by the server. Many image formats may be supported by the server, depending on the image processing library being used. An attacker can take advantage of this vulnerability by uploading a malicious XML file to a web application, such as an image file in SVG format. An attacker can use these files to extract information from the server, such as the server's hostname or password file.

XXE Attack Via Modified Content Type

When a web application receives a POST request they usually expect the content of the request to be of a certain type, normally 'application/x-www-form-urlencoded'. However, if a website is configured to accept other types of content such as XML, this XML will be accepted and parsed by the website, ultimately leaving the website vulnerable to injection by malicious XML.

XInclude Attack

There are numerous web applications which, after receiving client-submitted data, embed this data server-side into an XML document for subsequent parsing. Although this makes it difficult to carry out one of the more typical XXE attacks, since a would-be attacker would be unable to control the entire XML document, they can still attempt an XInclude attack.

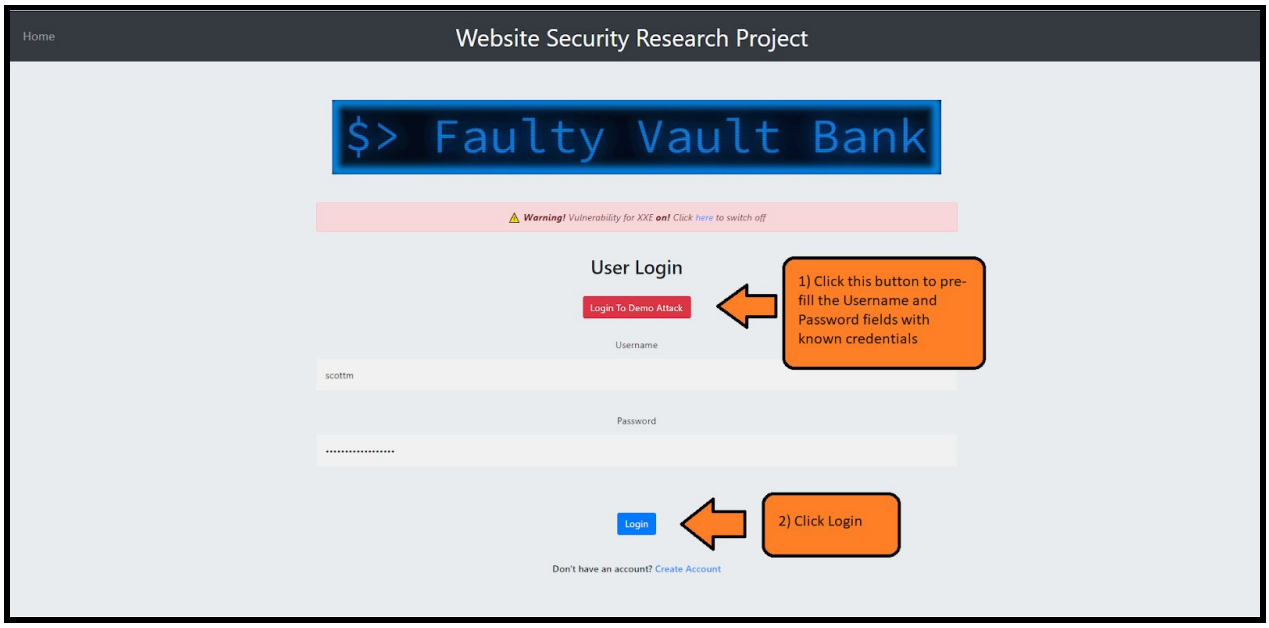
XInclude, defined in the XML specification, allows for an XML document to be built from a number of different sub-documents. This gives rise to a situation where an attacker, using XInclude, would only need to inject malicious XML code into one of these sub-documents prior to being incorporated into the server-side XML document in order to carry out an XXE attack.

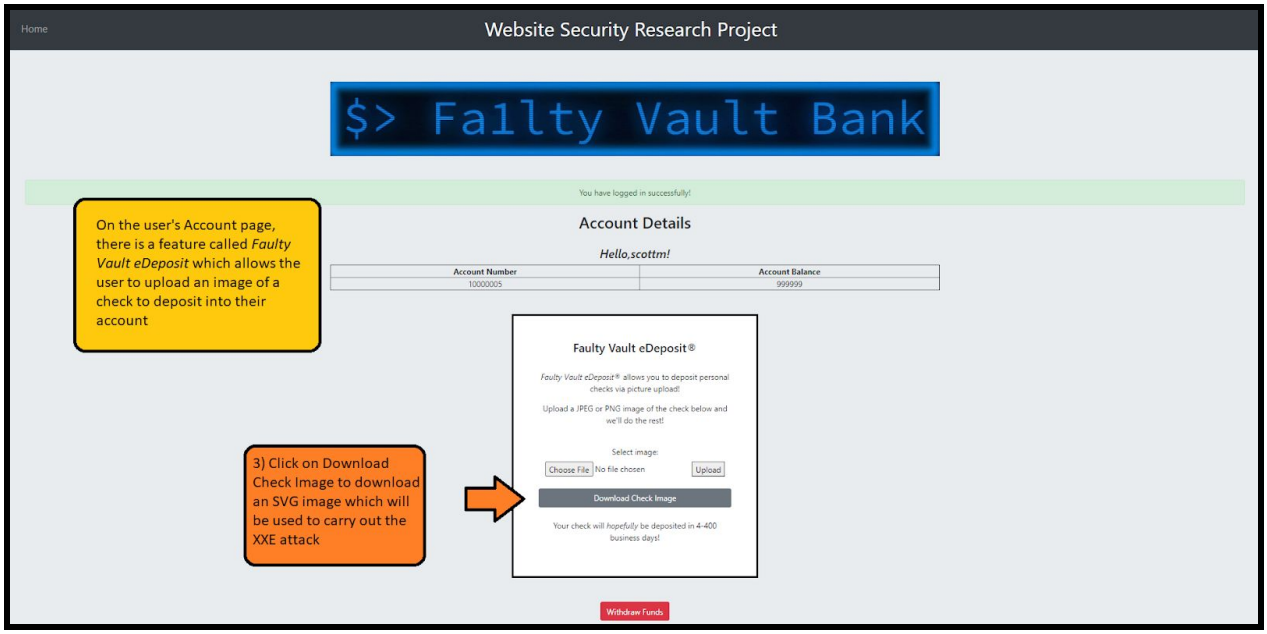
As you can see, there are numerous types of XXE attack strategies. In this project, we will be carrying out an XXE attack via file upload to our web application.

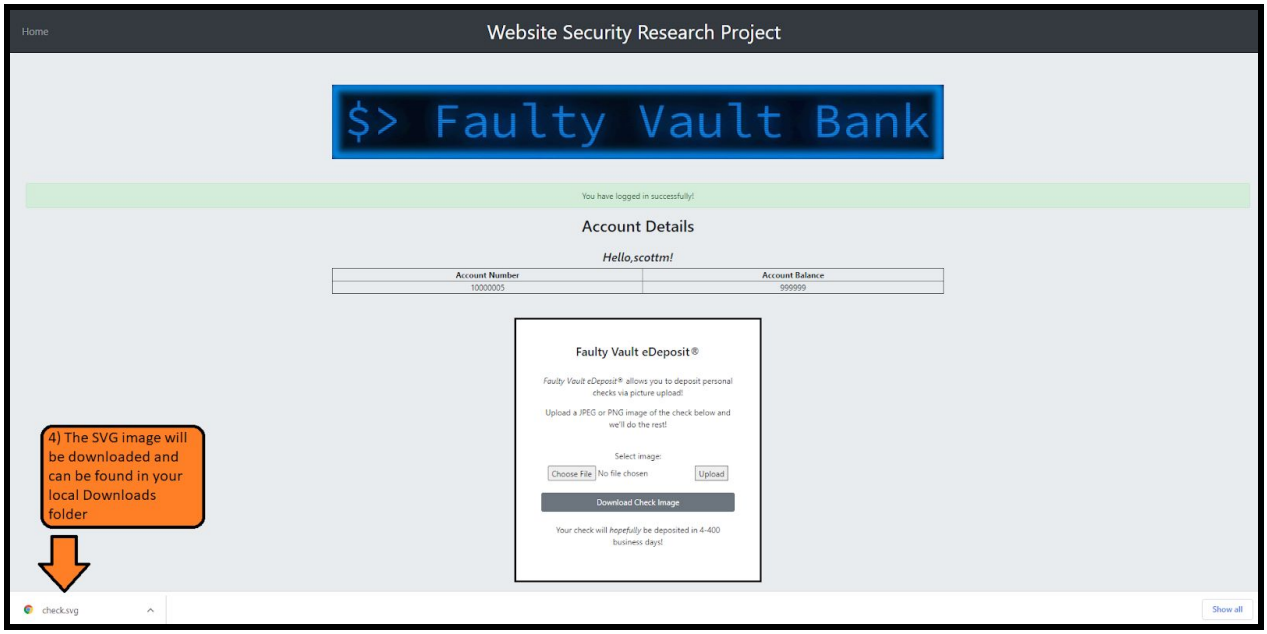
Attack Procedures

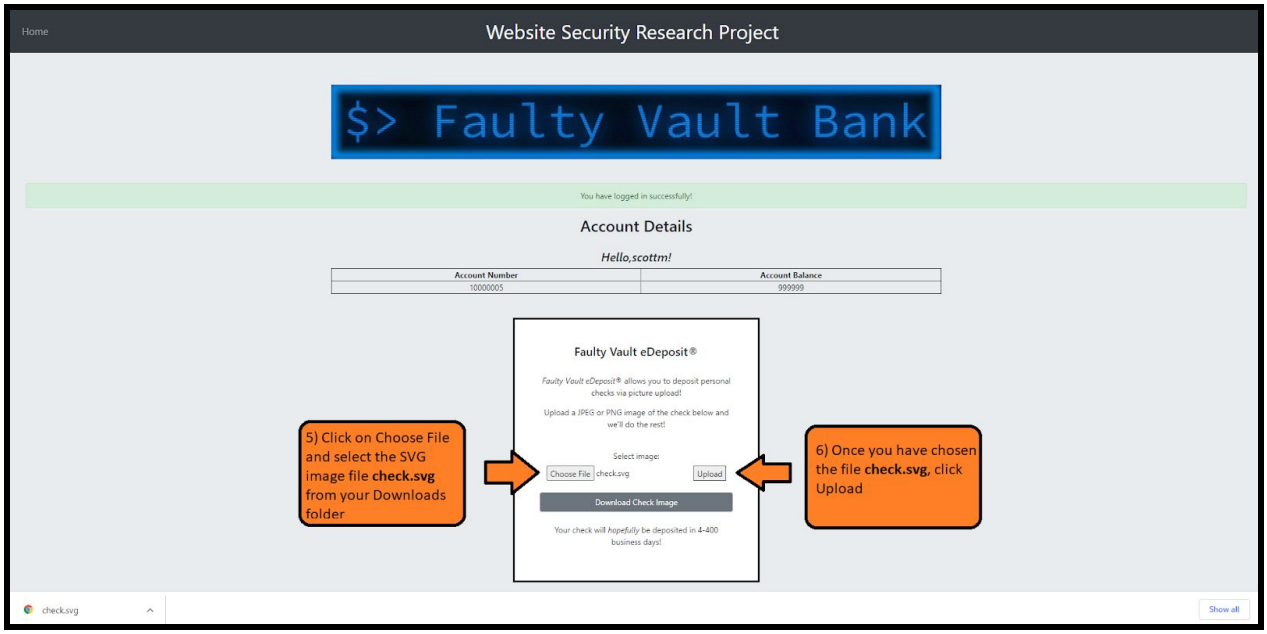
XXE Attack Via File Upload

Please use Google Chrome or Mozilla Firefox browser to carry out attack





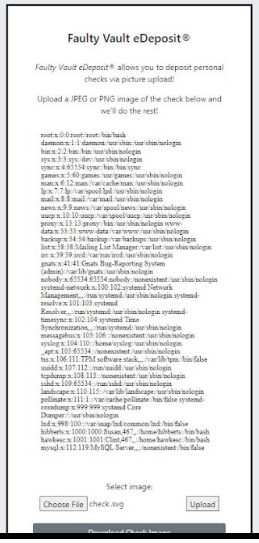

Home
Website Security Research Project
$> Faulty Vault Bank
You have logged in successfully!
Account Details
Hello,scottm!
Account Number
Account Balance
10000005
999999
Faulty Vault eDeposit®
Faulty Vault eDeposit® allows you to deposit personal checks via picture upload!
Upload a JPEG or PNG image of the check below and we'll do the rest!
Select image:
Choose File
No file chosen
Upload
Download Check Image
Your check will hopefully be deposited in 4-400 business days!
4) The SVG image will be downloaded and can be found in your local Downloads folder
check.svg
Show all


Home
Website Security Research Project
$> Faulty Vault Bank
You have logged in successfully!
Account Details
Hello,scottm!
Account Number
Account Balance
10000005
999999
Faulty Vault eDeposit®
Faulty Vault eDeposit® allows you to deposit personal checks via picture upload!
Upload a JPEG or PNG image of the check below and we'll do the rest!
Select image:
Choose File
check.svg
Upload
Download Check Image
Your check will hopefully be deposited in 4-400 business days!
5) Click on Choose File and select the SVG image file **check.svg** from your Downloads folder
6) Once you have chosen the file **check.svg**, click Upload
check.svg
Show all

Hello,scottm!

Account Number	Account Balance
10000005	999999

7) The contents of the server's etc/passwd file has been dumped on screen after the SVG image upload. This file stores all the user names and accounts on the server, allowing an attacker to carry out reconnaissance on the system



Faulty Vault eDeposit®

Faulty Vault eDeposit® allows you to deposit personal checks via picture upload!

Upload a JPEG or PNG image of the check below and we'll do the rest!

root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System
(admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-network:x:100:102:systemd Network
Management,,,:/run/systemd:/usr/sbin/nologin systemd-
resolve:x:101:103:systemd
Resolver,,,:/run/systemd:/usr/sbin/nologin systemd-
timesync:x:102:104:systemd Time
Synchronization,,,:/run/systemd:/usr/sbin/nologin
messagebus:x:103:106::/nonexistent:/usr/sbin/nologin
syslog:x:104:110::/home/syslog:/usr/sbin/nologin
_apt:x:105:65534::/nonexistent:/usr/sbin/nologin
tss:x:106:111:TPM software stack,,,:/var/lib/tpm:/bin/false
uuidd:x:107:112::/run/uuidd:/usr/sbin/nologin
tcpdump:x:108:113::/nonexistent:/usr/sbin/nologin
sshd:x:109:65534::/run/sshd:/usr/sbin/nologin
landscape:x:110:115::/var/lib/landscape:/usr/sbin/nologin
pollinate:x:111:1::/var/cache/pollinate:/bin/false systemd-
coredump:x:999:999:systemd Core
Dumper:/:/usr/sbin/nologin
lxd:x:998:100::/var/snap/lxd/common/lxd:/bin/false
hibberts:x:1000:1000:Susan,467,,:/home/hibberts:/bin/bash
hawkeac:x:1001:1001:Clint,467,,:/home/hawkeac:/bin/bash
mysql:x:112:119:MySQL Server,,,:/nonexistent:/bin/false

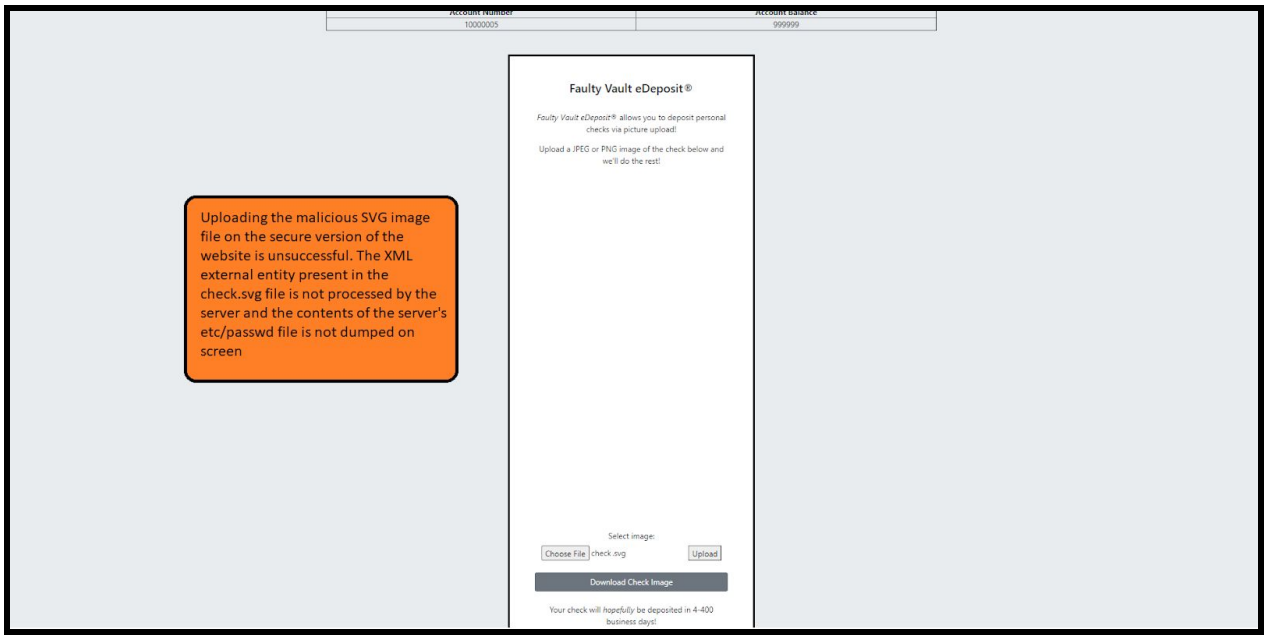
Select image:

Choose File check.svg Upload

Website Hardening

XXE vulnerabilities are primarily the result of the XML specification containing a number of potentially dangerous and exploitable features which standard XML parsers support, even if these features are not used by an application. The simplest and most effective way to protect against an XXE attack, therefore, is to disable these susceptible features in the first place. This is usually achieved by disabling all processing of external entities and disabling support for XInclude.

Attempt at XXE Attack Via File Upload After Website Hardening



Resources

https://owasp.org/www-project-top-ten/2017/A4_2017-XML_External_Entities_(XXE)

https://portswigger.net/web-security/xxe

https://www.w3schools.com/xml/xml_whatis.asp

https://owasp.org/www-project-top-ten/2017/A4_2017-XML_External_Entities_(XXE)

https://medium.com/@onehackman/exploiting-xml-external-entity-xxe-injections-b0e3eac388f9

https://stackoverflow.com/questions/4991171/auto-line-wrapping-in-svg-text

https://security.stackexchange.com/questions/92766/what-can-hackers-do-with-ability-to-read-etc-passwd

https://www.cyberciti.biz/faq/understanding-etcpasswd-file-format/