Modern Web Security

The Art of Creating and Breaking Assertions

AppSec California 2020
John Villamil

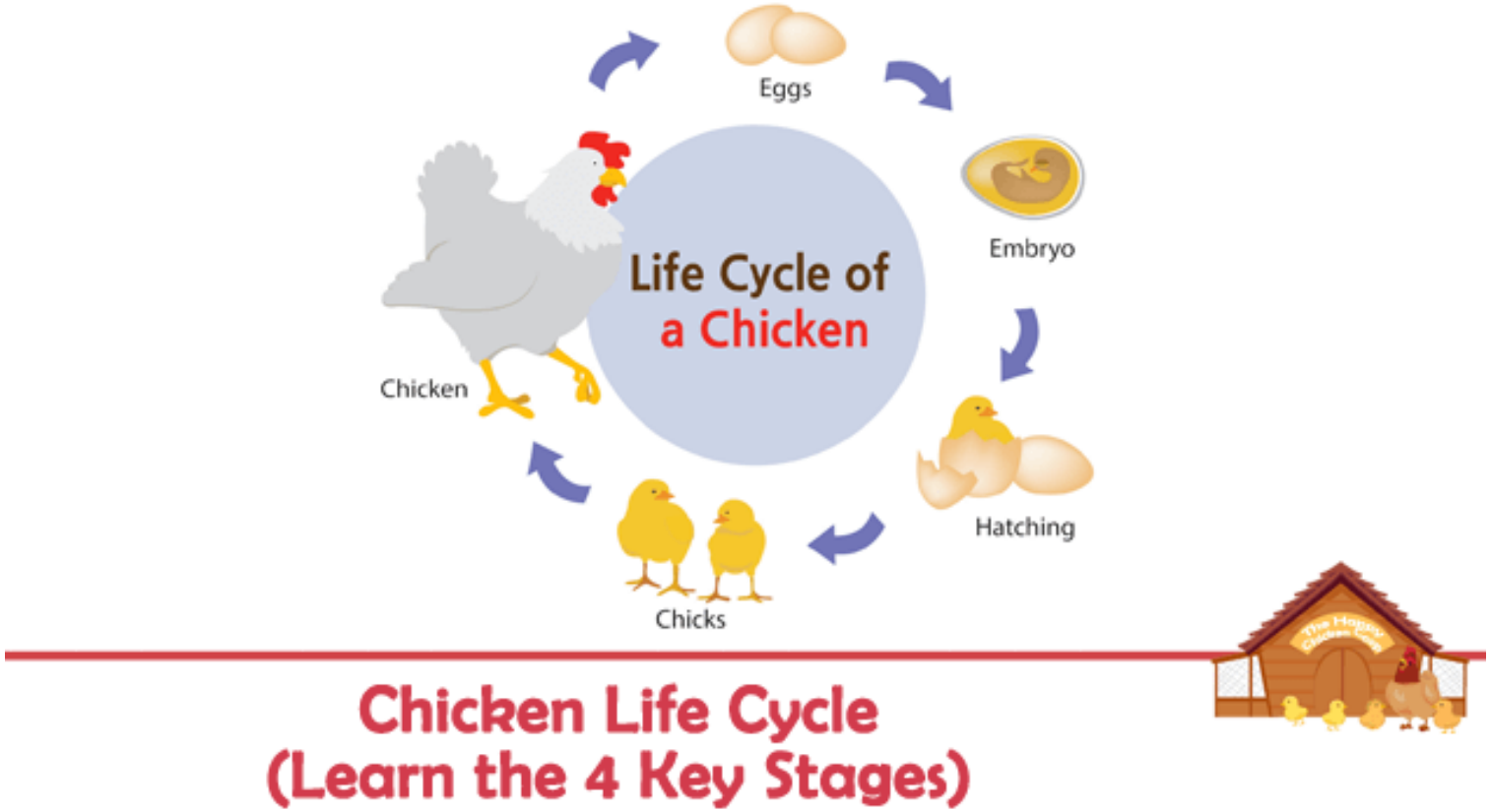
Goals

1. Communicate my opinion on how web security has evolved
2. Show a very brief history of web application security
3. Show how modern web security is now better defined by “assertions”
4. Where is it going?

Build with Security

- Small Security Consulting Firm
  - < 10 people
  - Specializes in Web Application Security
- 3 years old
- Over 80 clients

Web security over the years



https://www.thehappychickencoop.com/chicken-life-cycle/

"The OWASP Top 10 is a standard awareness document for developers and web application security. It represents a broad consensus about the most critical security risks to web applications."

https://owasp.org/www-project-top-ten/

OWASP Top 10 - 2013	→	OWASP Top 10 - 2017
A1 – Injection	→	A1:2017-Injection
A2 – Broken Authentication and Session Management	→	A2:2017-Broken Authentication
A3 – Cross-Site Scripting (XSS)	↘	A3:2017-Sensitive Data Exposure
A4 – Insecure Direct Object References [Merged+A7]	∪	A4:2017-XML External Entities (XXE) [NEW]
A5 – Security Misconfiguration	↘	A5:2017-Broken Access Control [Merged]
A6 – Sensitive Data Exposure	↗	A6:2017-Security Misconfiguration
A7 – Missing Function Level Access Contr [Merged+A4]	∪	A7:2017-Cross-Site Scripting (XSS)
A8 – Cross-Site Request Forgery (CSRF)	☒	A8:2017-Insecure Deserialization [NEW, Community]
A9 – Using Components with Known Vulnerabilities	→	A9:2017-Using Components with Known Vulnerabilities
A10 – Unvalidated Redirects and Forwards	☒	A10:2017-Insufficient Logging&Monitoring [NEW,Comm.]

https://github.com/OWASP/Top10

OWASP Top Ten	2003	2004	2007	2010	2013	2017 RC1	2017 RC2
Unvalidated Input	A1	A1 [9]	×	×	×	×	×
Buffer Overflows	A5	A5	×	×	×	×	×
Denial of Service	×	A9 [2]	×	×	×	×	×
Injection	A6	A6 [3]	A2	A1 [10]	A1	A1	A1
Cross Site Scripting (XSS)	A4	A4	A1	A2	A3	A3	A7
Broken Authentication and Session Management	A3	A3	A7	A3	A2	A2	A2
Insecure Direct Object Reference	×	A2	A4 [11]	A4	A4	A4 [20]	A5 [20]
Cross Site Request Forgery (CSRF)	×	×	A5	A5	A8	A8	×
Security Misconfiguration	A10	A10 [3][5]	×	A6	A5	A5	A6
Broken Access Control	A2	A2 [1]	A10 [13]	A8	A7 [16]	A4	A5
Insufficient Attack Protection	×	×	×	×	×	A7	×
Unvalidated Redirects and Forwards	×	×	×	A10	A10	×	×
Information Leakage and Improper Error Handling	A7	A7 [14][4]	A6	A6 [8]	×	×	×
Malicious File Execution	×	×	A3	A6 [8]	×	×	×
Sensitive Data Exposure	A8	A8 [6][5]	A8	A7	A6 [17]	A6	A3
Insecure Communications	×	A10	A9 [7]	A9	×	×	×
Remote Administration Flaws	A9	×	×	×	×	×	×
Using Known Vulnerable Components	×	×	×	×	A9 [18][19]	A9	A9
Unprotected APIs	×	×	×	×	×	A10	×
Insecure Deserialization	×	×	×	×	×	×	A8
XML External Entity (XXE)	×	×	×	×	×	×	A4
Insufficient Logging & Monitoring	×	×	×	×	×	×	A10

Pentesting under constant threat

- Client/Server Frameworks like React, Django, Rails
- Automated Scanners
- General education among developers
- Detrimental affects of security snake oil
- Bug Bounties
- New languages and frameworks
  - MVC
  - NoSQL
  - JWT Tokens
  - SSO like Okta
  - etc

http://securitysnakeoil.org/

The Need for Evolution

- Pentesting is still here
  - It must provide value

Pen Testing is Dying- Here are the Six Things that are killing

Don't all kill the messenger at once but the sad truth is there are a growing number of organizatio
'm afraid they have good reasons.

Pen Testing: Dead in 2009

Does pen testing belong in the QA department? Fortify Co-Founder and Chief Scientist Brian Chess
2009 will mark the end of pen tests as we know them. His theory is being met with resistance

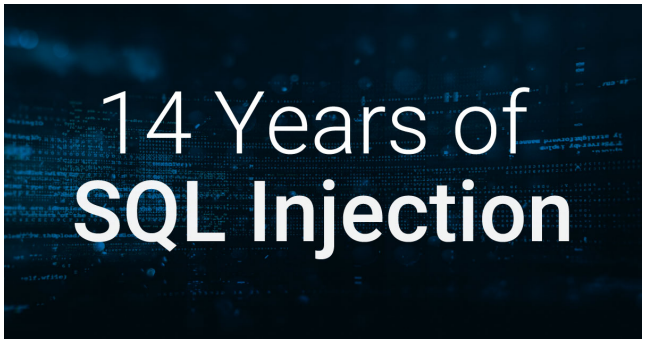
Penetrating Testing is Dead as We Now Know It.

Published on February 19, 2018

Definition of *assertion*
: the act of asserting or something that is asserted: such as

a: insistent and positive affirming, maintaining, or defending (as of a right or attribute)
//an *assertion* of ownership/innocence

b: a declaration that something is the case
//He presented no evidence to support his *assertions*.



https://www.netsparker.com/blog/web-security/sql-injection-vulnerability-history/ Aug 2013

- Injection bugs tend to be well explained
  - SQL, Command Execution, XSS, etc

- Assertion bugs are usually harder to define

https://wiki.owasp.org/index.php/Testing_for_business_logic

- “Testing for business logic flaws in a multi-functional dynamic web application requires thinking in unconventional methods.”
- “The classification of business logic flaws has been under-studied”
- “There is debate within the community about whether these problems represent particularly new concepts, or if they are variations of well-known principles.”

Where do assertions come from?

- Comments in source code
  - TODOs or statements of functionality
- RFCs
  - https://tools.ietf.org/html/rfc6749#section-4.1.1
    - ie Oauth

> **scope**
>
> OPTIONAL. The scope of the access request as described by Section 3.3.
>
> **state**
>
> RECOMMENDED. An opaque value used by the client to maintain state between the request and callback. The authorization server includes this value when redirecting the user-agent back to the client. The parameter SHOULD be used for preventing cross-site request forgery as described in Section 10.12.

- Conversations and meetings with engineers
- API docs and other public documentation
- Statements made in features or changesets
- Comments and issues in Github

Paypal 2fa Bypass 2016

https://henryhoggard.co.uk/blog/Paypal-2FA-Bypass

Step 3: Using a proxy, remove “securityQuestion0” and “securityQuestion1” from the post data.

```
selectOption=SECURITY_QUESTION&securityQuestion0=test&securityQuestion1=test&jsEnabled=1&ex
```

Step 4: Profit

#434763

Incorrect details on OAuth permissions screen allows DMs to be read without permission

Share:

State	Resolved (Closed)	Severity	Medium (4.3)
Disclosed	December 13, 2018 4:01pm -0800	Participants	
Reported To	Twitter	Visibility	Disclosed (Full)
Asset	*.twitter.com (Domain)		
Weakness	Privacy Violation		
Bounty	$2,940		

Summary:

The OAuth screen can be tricked into saying that an app cannot read Direct Messages. Despite that, DMs can be read.

Description:

The official Twitter API keys have been leaked and are in use in several popular apps.
The iPhone keys and Google TV keys (as seen on https://gist.github.com/shobotch/5160017) present an OAuth screen which says the app "Will not be able to: Access your direct messages."
This is false. The apps *can* read DMs.

#723118 [IDOR] API endpoint leaking sensitive user information

State	Resolved (Closed)	Severity	Medium (6.5)
Disclosed	January 8, 2020 7:25pm -0800	Participants	
Reported To	Razer	Visibility	Disclosed (Full)
Asset	Group 2 assets (Rewards based on Imp... (Other)		
Weakness	Improper Access Control - Generic		
Bounty	$375		

Steps To Reproduce:

1. Go to a random user's profile, say, ttps://insider.razer.com/index.php?members/kajira.714/
2. Look at all the information that can be accessed publicly.
3. Now go to https://insider.razer.com/api.php?action=getuserprofile&user_id=714, and as you can see, a lot of user metadata is getting leaked, like the email ID, FB and Twitter ID, RZR_ID, CSRF token etc.

#685909 Searching from Hacktivity returns hits for words in limited disclosure reports that are not visible

State: Resolved (Closed)
Severity: Medium (4.4)

#502593 Attacker is able to access commit title and team member comments which are supposed to be private

State: Resolved (Closed)
Severity: High (7 ~ 8.9)

#719631 [Partial] SSN & [PII] exposed through iPERMs Presentation Slide.

State: Resolved (Closed)
Severity: Critical (9 ~ 10)
Disclosed: December 2, 2019 12:03pm -0800
Participants
Reported To: U.S. Dept Of Defense
Visibility: Disclosed (Full)
Weakness: Information Disclosure

https://hackerone.com/reports/689314
https://gitlab.com/gitlab-org/gitlab-foss/issues/67109

#689314 **Project Template functionality can be used to copy private project data, such as repository, confidential issues, snippets, and merge requests**

State ● Resolved (Closed) Severity Critical (9 ~ 10)

jobert submitted a report to GitLab. Sep 5th (5 months ago)

I've found a three minor vulnerabilities which, when combined, allow an attacker to copy private repositories, confidential issues, private snippets, and then some. I'll go through the code path to explain the vulnerabilities and how they are combined. See the **Proof of Concept** section if you want to reproduce it immediately.

Let's start at the `ProjectsController` of EE, which is prepended to `app/controllers/projects_controller.rb` in an EE instance.

HOW TO SHOP FOR FREE ONLINE – SECURITY ANALYSIS OF CASHIER-AS-A-SERVICE BASED WEB STORES

Rui Wang (Indiana Univ.)

Joint work with
Shuo Chen (MSR), XiaoFeng Wang (Indiana Univ.), Shaz Qadeer (MSR)

https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/caas-oakland-final.pdf
http://www.cs.columbia.edu/~suman/secure_sw_devel/Semantic-bugs-shopfree.pdf

Summary of the 9 logic flaws

Merchant	CaaS	Flaw	Result	Specific to	Who fixed it
NopCommerce	PayPal Standard	Insufficient check of payment total	Pay arbitrary price	Merchant	Merchant
NopCommerce	Amazon Simple Pay	Insufficient protection against a shopper with a malicious merchant	Shop for free	Payment method	CaaS
Interspire	Amazon Simple Pay	Incorrect use of signature	Shop for free	Merchant	Merchant
Interspire	PayPal Express	Insufficient protection against a shopper with two shopping sessions	Pay arbitrary price	Merchant	Merchant
Interspire	PayPal Standard	Payment notification can be replayed under certain condition	Pay arbitrary price	Merchant	Merchant
Interspire	Google Checkout	Can add items to cart after payment total is fixed	Pay arbitrary price	Merchant	Merchant
JR.com	Checkout By Amazon	Insufficient protection against a shopper with a malicious merchant	Pay arbitrary price	Merchant	Merchant
Buy.com	PayPal Express	Paypal token allowed to be reused	Pay arbitrary price	Merchant	Merchant
Web stores using Amazon SDKs	Amazon Flexible Payments	Insufficient signature validation	Shop for free	CaaS	CaaS

Different Mentality

- Auditing for injection vulnerabilities
  - Attempt to ensure those vulnerabilities don’t exist
- Auditing for assertions
  - Attempt to ensure well defined, controlled, and expected behavior in an application

Future of Web Sec?

Pierre Bourdon
@delroth_

Zelda OOT speedruns in 2020: "yeah we just manipulate the heap by lifting rocks to exploit a use-after-free and rewrite a function pointer to jump into a multi stage payload involving Link's name and the buttons pressed on controllers 1 and 3"

RTA Viable Credits Warp in Kokiri Forest Discovered in OOT!!!!!!!!
Posted in r/speedrun by u/Lobsterzelda • 868 points and 150 comments
reddit.com

5:53 PM · Jan 15, 2020 · Twitter Web App

109 Retweets **281** Likes

yoshi314 @wengo314 · Jan 15
Replying to @delroth_
we've already had people reprogram castlevania via controller on psx (non-tas), so i guess sky's the limit.
2

mudlord @opcode_raeg · Jan 19
Replying to @delroth_
Is that even a speedrun then, if all you do is code execution without playing the game at all.
1 1

Race Conditions

- Interesting bug class
  - Blends both mentalities
  - Very under represented
  - “Hacking Starbucks for unlimited coffee” 2015 http://sakurity.com/blog/2015/05/21/starbucks.html

- Burp Turbo intruder
  - https://github.com/PortSwigger/turbo-intruder
  - Released and improved upon in 2019

- Will we see this bug class rise higher on the OWASP Top10?

Goals Recap

1. Communicate my opinion on how web security has evolved
2. Show a very brief history of web application security
3. Show how modern web security is now better defined by “assertions”
4. Where is it going?

Thank you

- https://www.doyensec.com
- @day6reak
- info@doyensec.com