



Andrew van der
Stock



TOP10

Brian Glas
Andrew van der Stock
Torsten Gigler
Neil Smithline

owasp.org/Top10



- Top 10 Risks – not Top 10 impacts, likelihoods, or vulnerabilities
- First released in 2003, 2021 is the 7th update
- Audience
 - Developers, lead developers, architects
 - Framework developers (but they really should be using ASVS)
 - AppSec program management (CISOs, CTOs, and so on)
 - AppSec professionals: consultancies, tools, vendors, trainers



- Collaborative - all of us do all the things
- Goals: conceptual integrity and to include our community

	What they primarily do
Brian Glas	Co-lead, author, data scientist, data analysis, risk rankings, interface to data sources, and more
Torsten Gigler	Co-lead, author, data analysis, risk rankings, document template, website, English editor, German translation, and more
Neil Smithline	Co-lead, author, data analysis, risk rankings, and valuable counsel and advice, and more
Andrew van der Stock	Co-lead, author, data analysis, risk rankings, persnickety grammar person, often gets interviews and media outlet requests, and more



- After nearly 20 years, injection is no longer A1
 - Even with XSS combined into injection
 - Shorter by design
 - Mobile-first, PDF and wall poster available (soon)
 - New look after a decade
-
- How to adopt as a (pseudo-)standard and basic appsec program
 - A1..A10 titles are root causes, not symptoms
 - Four new categories, including Insecure Design and SSRF
 - And lastly, next steps



Data Collection



Data Analysis



Review



Responsive Web and Mobile Version



Industry Survey



Write Up



Translations



PDF and Developer Poster





- Want to learn how the Top 10 was made?
- Please attend Brian Glas' talk at 1700 US EDT / 2100 UTC

[The making of the OWASP Top 10 and beyond](#)

Risky Click of the Day
It's not a Rick Roll. Promise





Broken Access Control

TOP10

- Bypass access control checks
- Unauthorized access to accounts
- Unauthorized creation, reading, updating and deletion of data
- Elevation of privilege
- Privacy and regulatory impacts
- The biggest breaches and largest costs



34 CWEs
19k CVEs

Found in 3.8% apps
Occurred 318k times

Weighted Exploit: 6.9
Weighted Impact: 5.9



Cryptographic Failures

- Covers
 - Some facets of “Sensitive Data Exposure”
 - Missing or ineffective data at rest controls
 - Missing or ineffective TLS
 - Missing or ineffective configuration
- Includes CWEs for hard coded passwords
- Mostly found during code reviews or static code analysis

29 CWEs
3075 CVEs

Found in 4.5% apps
Occurred 234k times

Weighted Exploit: 7.3
Weighted Impact: 6.8



Injection

- Moving down from A1 ... at last
- Now covers XSS and JavaScript injection due to safer view frameworks
- Easily - but now rarely - found using tools
- Still quite exploitable
- Adopt better frameworks and more secure paved roads
- Provide observability to development teams if they use less secure alternatives
- Help by providing paved roads and gold standard support for safer frameworks

33 CWEs
32k CVEs

Found in 3.4% apps
Occurred 274k times

Weighted Exploit: 7.3
Weighted Impact: 7.2



Insecure Design

TOP10

NEW

- New category obtained from data
 - Broad category, but it's NOT a catch all bucket!
- Insecure design directly impacts application security
- Insecure design is easily the costliest to fix later (up to 100x)
- **Really shift left!** Earlier integration with the development and teams
- **Threat model** Where are controls needed? Are they there? Do they work?
- **Adopt better frameworks!** Create secure paved roads **with** dev teams
- **Test, test, and test!** Create unit, integration, and other tests

40 CWEs
2691 CVEs

Found in 3.0% apps
Occurred 262k times

Weighted Exploit: 6.5
Weighted Impact: 6.8



Security Misconfiguration

TOP10

- Cloud infrastructure as code == slight jump to A5
- Covers unhardened, misconfigured, and default configurations
- Eliminate the risk: Build “paved road” pre-hardened development and production frameworks, components, and build configurations
- Surface the risk: Build tools to identify weakly or insecurely configured components and applications

20 CWEs
789 CVEs

Found in 4.5% apps
Occurred 208k times

Weighted Exploit: 8.1
Weighted Impact: 6.6



Vulnerable and Outdated Components



- Root cause of the LARGEST and MOST COSTLY breach of all time
- Covers the USG Executive Order for supply chain security
- Covers “Patching Applications” of the ASD Essential 8
- CWEs are self-referential to previous OWASP Top 10's
- Recommend using CI/CD tools to warn for outdated components
- Strongly recommend breaking the build for vulnerable components

3 CWEs
0 CVEs

Found in 8.8% apps
Occurred 30k times

Weighted Exploit: 5.0
Weighted Impact: 5.0



Identification and authentication failures

TOP10

- Replaces 2017:A2 Broken Authentication
- Includes authentication and session management issues
- CWEs cover nearly all the ASVS V2 and V3 at Level 1
- Protect against re-used, breached, and weak passwords
- Add MFA to all the things
- Use the ASVS to improve authentication of your apps
- Consider a “paved road” secured and shared authentication service

22 CWEs
3897 CVEs

Found in 2.6% apps
Occurred 132k times

Weighted Exploit: 7.4
Weighted Impact: 6.5



Software and Data Integrity Failures

TOP10

NEW

- Integrity of business or privacy critical data
- Lack of integrity of includes from content data networks
- Software updates without integrity
- CI/CD pipelines without check in or build checks, unsigned output
- Improve the integrity of the build process
- Use SBOM to identify authentic builds and updates
- Use sub-resource integrity if using CDN for web page includes
- Consider how you vet and ensure npm, maven, repos are legit

10 CWEs
1152 CVEs

Found in 2.0% apps
Occurred 47.9k times

Weighted Exploit: 6.9
Weighted Impact: 7.9



Security Logging and Monitoring Failures

TOP10

- Included by survey results for a second time
- Critical to reduce the breach window, response time, and cleanup
- Necessary if you have breach disclosure laws
- Critical if you intend to prosecute
- Interview or code review the best review technique
- Static code analysis can't find the absence
- Still difficult to dynamically test

4 CWEs

242 CVEs

Found in 6.5% apps

Occurred 53.6k times

Weighted Exploit: 6.9

Weighted Impact: 5.0



Server-Side Request Forgery (SSRF)

TOP10

NEW

- Included by survey
- Written by Orange Tsai – thank you so much!
- Everyone needs to learn how to test
 - Developers
 - AppSec Professionals
- Frameworks need to protect against SSRF by default
- IDEs (and frameworks though *doc) need to highlight potential SSRF
- Like XXE, we hope the focus in OWASP Top 10 2021 will help retire it

1 CWEs
385 CVEs

Found in 2.7% apps
Occurred 9.5k times

Weighted Exploit: 8.2
Weighted Impact: 6.7



Next Steps



- OWASP Top 10 is the MINIMUM
 - There's always something that nearly makes it in
 - Include these in any coding standard or testing
-
- **Code Quality issues**
 - **Denial of Service**
 - **Memory Management Errors**



- Frameworks helped eliminate bug classes. Please continue!
- Threat model, eliminate or reduce bug classes, and test, test, test
- Improve your appsec program or checklists using:
 - OWASP Proactive Controls for entry level developers
 - OWASP Application Security Verification Standard for all developers
 - OWASP Cheat Sheets for concrete advice
 - OWASP Web Testing Guide to learn how to test the ASVS and Top 10
 - OWASP Education and Training Committee developing a curriculum and framework for developer training



- Head over to #project-top-10 on OWASP Slack and say hi
 - Still to produce one pager and PDF version
 - Looking for translators - #top-10-translations
- Log issues at <https://github.com/OWASP/Top10>
 - Review drafts, suggest improvements by logging issues
 - We work in GitHub in Markdown
 - Fork and branch to create PRs



#20th-anniv-flagshipproject

TOP10





TOP10

