Secure Coding with OWASP ASVS

Agenda

01 Intros & Getting Started

02 ASVS and using it for Code Review

03 Your turn for Secure Code Review!

04 Present your findings to one another ☺

05 Conclusion / Q&A

What we'll be doing

- Choose a ticket and inspect its commit or changeset
- File a report for each security finding, then review another ticket or ask for a second opinion
- Patch and run the application to test your fix!

Getting started

```
git clone https://github.com/cronchie/asvs-defcon-workshop.git
cd asvs-defcon-workshop
docker compose --profile turnstile --profile herd --profile apex up
-d
```

Open:

- archive viewer, ASVS, and field report: <http://localhost:8082/>
- Turnstile storefront: <http://localhost:5173/>
- Turnstile API docs: <http://localhost:8000/docs>
- HERD servlet: sh herd/console/run-desktop.sh
- Apex console: docker compose --profile apex --profile mosaic up -d

The Application Security Verification Standard

- One of OWASP's flagship projects
- Developed in the open by contributors like you and me <3
- Designed to be:
  - Actionable
  - Forkable
  - Flexible

The Application Security Verification Standard

- A curated set of **positive** requirements that govern **properties** of web application and API code
- Can be used as:
  - a checklist for secure coding, including writing the code and code reviews
  - a reference for penetration testing
  - secure architecture guidance
  - the basis for unit / integration tests
  - topics for developer training
  - a baseline for procurement decisions

```java
@Bean
JwtDecoder jwtDecoder() {
    NimbusJwtDecoder decoder = NimbusJwtDecoder.withJwkSetUri("https://login.example.com/.well-
        known/jwks.json").build();

    JwtTimestampValidator time = new JwtTimestampValidator();
    time.setClockSkew(java.time.Duration.ofSeconds(60));

    OAuth2TokenValidator<Jwt> issuer = jwt -> {
    String iss = jwt.getIssuer() != null ? jwt.getIssuer().toString() : "";
    boolean ok = iss.equals("https://login.example.com/");
    return ok
        ? OAuth2TokenValidatorResult.success()
        : OAuth2TokenValidatorResult.failure(new OAuth2Error("invalid_token", "Untrusted issuer", null));
    };

    OAuth2TokenValidator<Jwt> alg = jwt -> {
    String algorithm = jwt.getHeaders().getAlgorithm() != null ? jwt.getHeaders().getAlgorithm().getName() : "";
    boolean ok = algorithm.equals("RS256");
    return ok
        ? OAuth2TokenValidatorResult.success()
        : OAuth2TokenValidatorResult.failure(new OAuth2Error("invalid_token", "Unexpected signing algorithm",
        null));
    };

    decoder.setJwtValidator(new DelegatingOAuth2TokenValidator<>(time, issuer, alg));
    return decoder;
}
```

V10.3: OAuth Resource Server

#	Description	Level
10.3.1	Verify that the resource server only accepts access tokens that are intended for use with that service (audience). The audience may be included in a structured access token (such as the 'aud'claim in JWT), or it can be checked using the token introspection endpoint.	2
10.3.2	Verify that the resource server enforces authorization decisions based on claims from the access token that define delegated authorization. If claims such as 'sub', 'scope', and 'authorization_details'are present, they must be part of the decision.	2
10.3.3	Verify that if an access control decision requires identifying a unique user from an access token (JWT or related token introspection response), the resource server identifies the user from claims that cannot be reassigned to other users. Typically, it means using a combination of 'iss'and 'sub'claims.	2

ASVS
APPLICATION SECURITY VERIFICATION STANDARD

```
@Bean
JwtDecoder jwtDecoder() {


        ...
        ...
        ...

        OAuth2TokenValidator<Jwt> audience = jwt -> {
        List<String> aud = jwt.getAudience();
        boolean ok = aud != null && aud.contains("api://inventory");
        return ok
                ? OAuth2TokenValidatorResult.success()
                : OAuth2TokenValidatorResult.failure(new OAuth2Error("invalid_token", "Missing required
                audience", null));
        };

        decoder.setJwtValidator(new DelegatingOAuth2TokenValidator<>(time, issuer, alg));
        return decoder;
}
```

APPLICATION SECURITY VERIFICATION STANDARD

OTP rate-limiter example

```
type Limiter struct {
    attempts map[string]*store.Entry
    ttl      time.Duration
    now      func() time.Time
}

func (l *Limiter) RecordFailure(_ context.Context, key string, limit
int,
) error {
    e := store.Ensure(l.attempts, key, l.ttl, l.now())
    if e.Count >= limit {
        return ratelimit.ErrLimitExceeded
    }
    e.Count++
    return nil
}
```

Applicable requirement: ASVS 15.4.2

```go
type Limiter struct {
    mu       sync.Mutex
    attempts map[string]*store.Entry
    ttl      time.Duration
    now      func() time.Time
}

func (l *Limiter) RecordFailure(
    _ context.Context, key string, limit int,
) error {
    l.mu.Lock()
    defer l.mu.Unlock()

    e := store.Ensure(l.attempts, key, l.ttl, l.now())
    if e.Count >= limit {       // check and act
        return ratelimit.ErrLimitExceeded
    }
    e.Count++                   // both under the lock
    return nil
}
```

> *Verify that checks on a resource's state and the actions that depend on them are performed as a single atomic operation, to prevent time-of-check to time-of-use (TOCTOU) race conditions.*

APPLICATION SECURITY VERIFICATION STANDARD

The ticketing incident

issue list

#29 Copy the affected production orders into staging

Repo:	allied/turnstile
State:	open
Labels:	incident, ops, priority-critical
Assignee:	nadia
Reporter:	priya
Opened:	2026-05-06 07:40
Milestone:	incident

Commits

- 29-prod-snapshot.diff Wed, 6 May 2026 07:55:00 -0600, Nadia Brennan

---

priya commented 2026-05-06 07:40

```
We opened the park on Monday. By Tuesday morning, guests saw all thirty dates
marked sold out, while total revenue remained below a normal Saturday. Support
has received calls throughout each day.

One account has hundreds of unpaid orders across every date. A second account
has completed bookings and used tickets, while each order records a charge of
zero. Copy these records into staging before anyone changes the production
data.
```

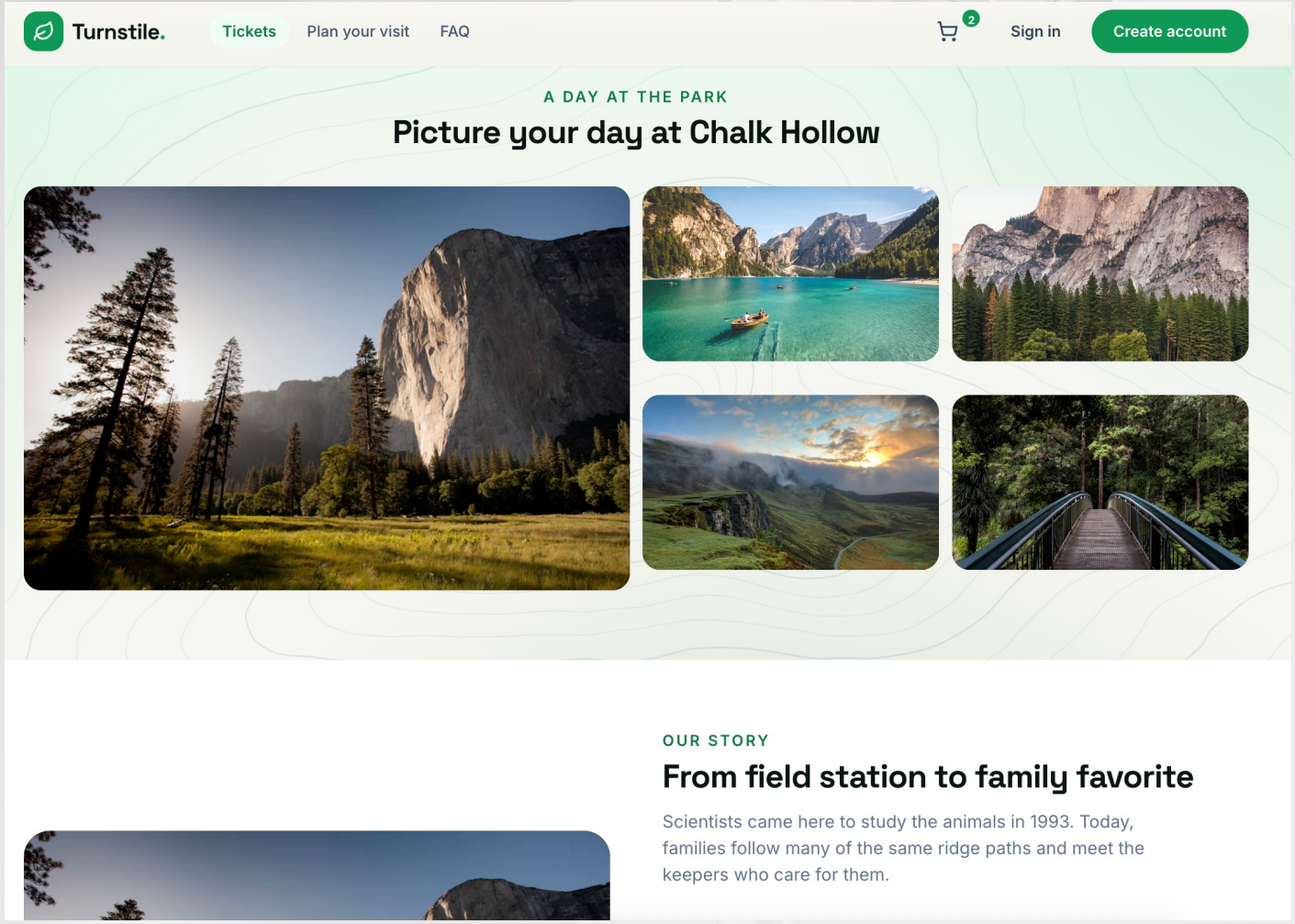
---

nadia commented 2026-05-06 07:55

```
Copied the affected accounts, orders, park dates, and ticket records into the
staging seed.
```

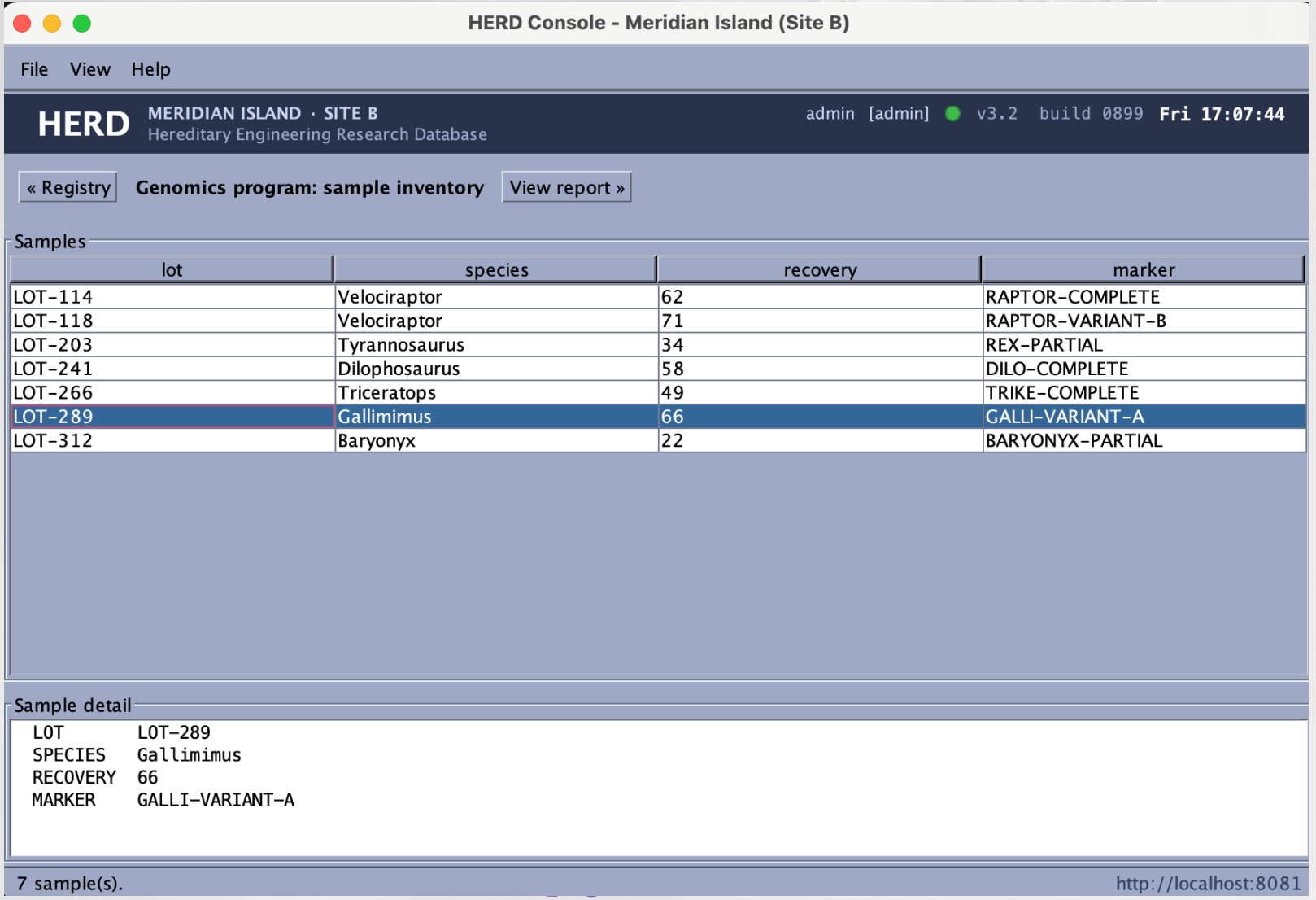
Repository contents

Turnstile, 2026
FastAPI, React, Git, and issue files



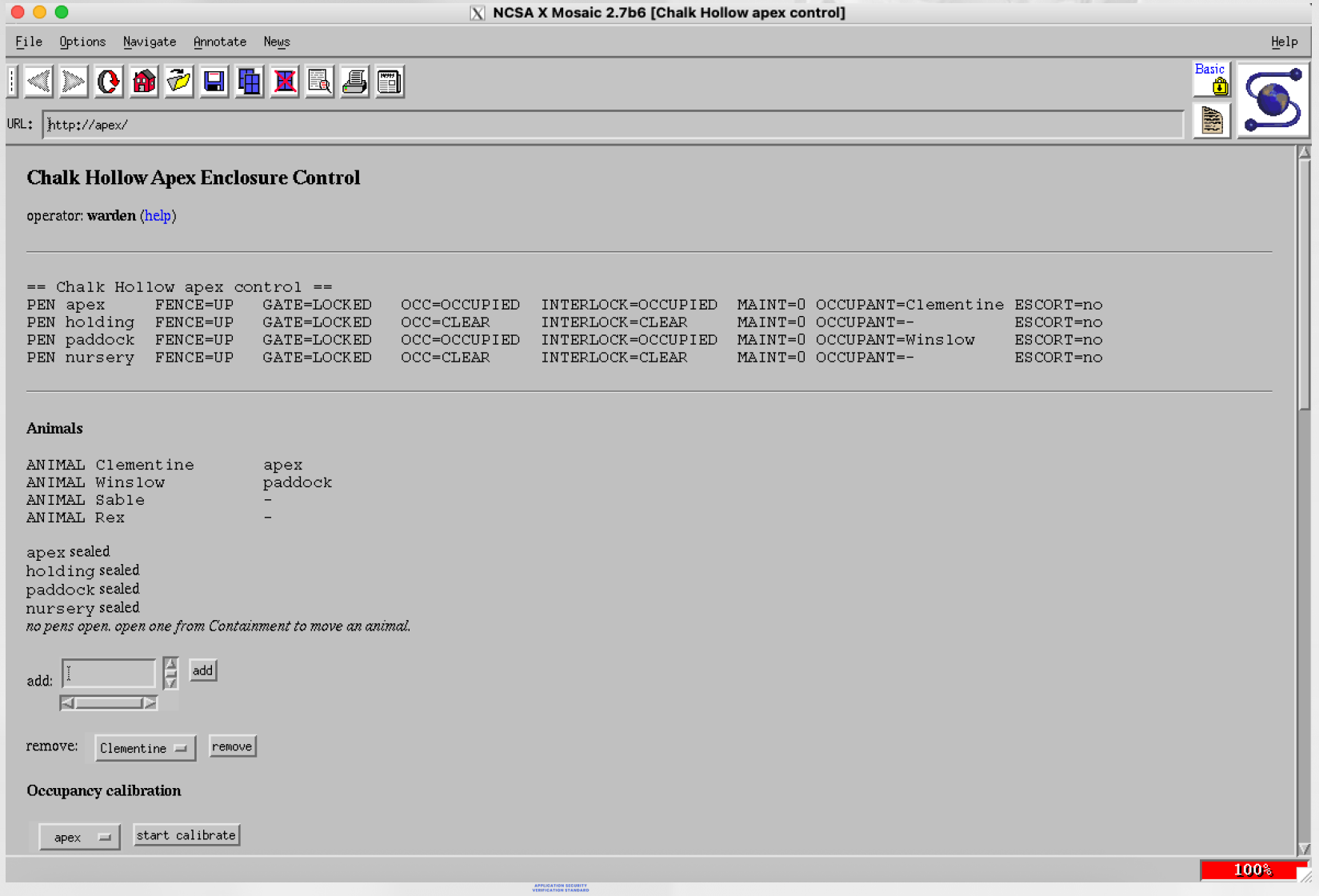
Repository contents

HERD, 1999
Java cloning and genomics registry, Bugzilla, and CVS



Repository contents

Apex, 1993
C enclosure controls,
GNATS, and CVS.



Live demonstration: review one change

1. Open a ticket.
2. Open its linked commit or changeset.
3. Read the diff and current code.
4. Select an ASVS requirement.
5. Open the field report.

Terminal: git show <commit>

engineering services

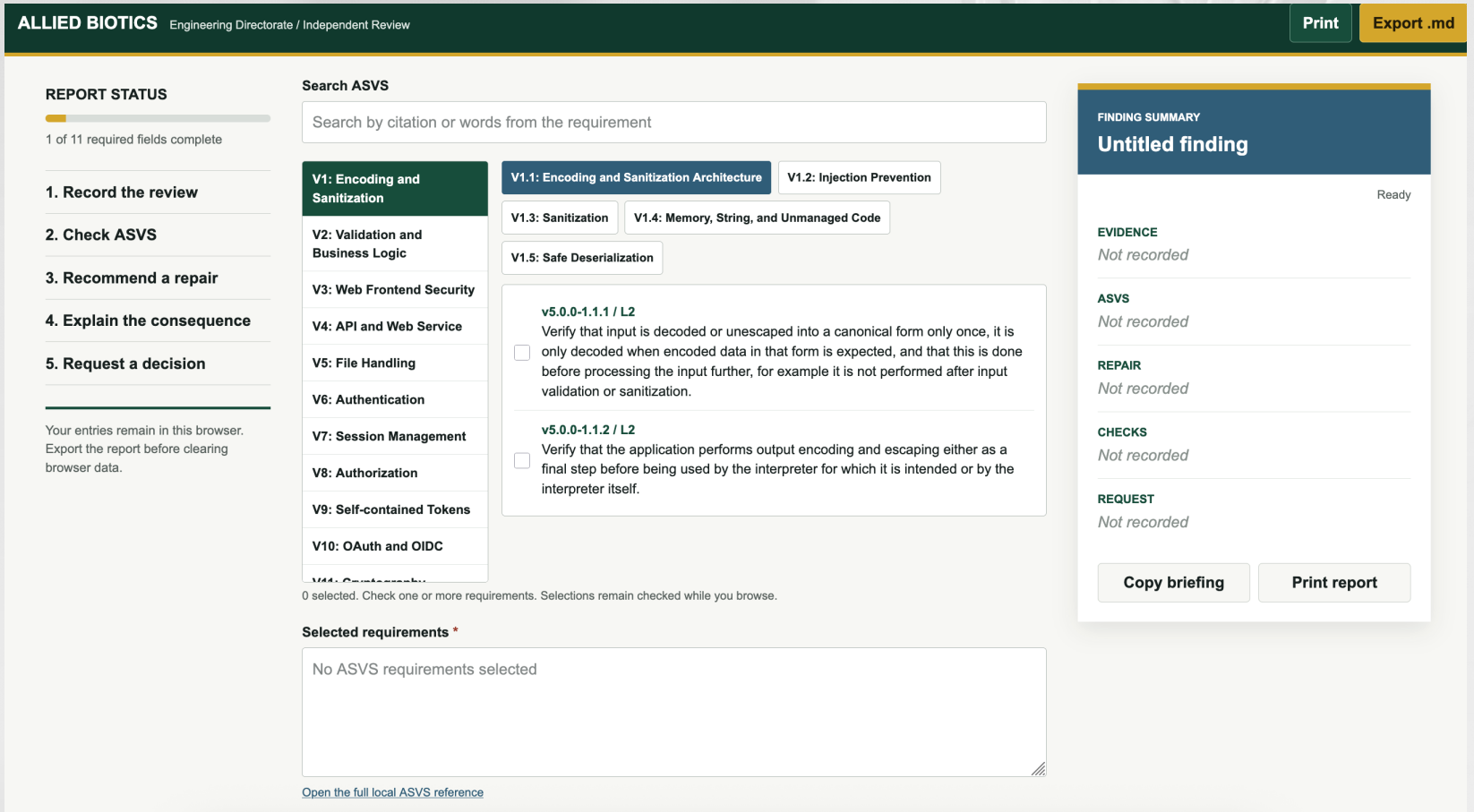
allied/turnstile

The Turnstile team filed one issue for each commit from the initial setup through the May incident review. Open an issue for the request ar

#	State	Labels	Assignee	Opened	Title
#1	closed	task, setup	nadia	2026-02-03 08:35	Set up the API and web workspaces
#2	closed	task, setup	nadia	2026-02-04 09:10	Run the database, API, and web app together
#3	closed	task, tooling	nadia	2026-02-05 13:20	Require type checks in both workspaces
#4	closed	feature, auth	nadia	2026-02-06 09:30	Store guest accounts and registration details
#5	closed	feature, auth	nadia	2026-02-09 14:10	Let returning guests sign in
#6	closed	feature, auth	nadia	2026-02-10 16:05	Return an authentication token after login
#7	closed	tech-debt, api	marcus	2026-02-12 17:40	Remove the duplicate signup model
#8	closed	feature, catalog	nadia	2026-02-16 10:15	Publish ticket types and park dates
#9	closed	feature, web	dana	2026-02-18 09:25	Show park dates and ticket prices in the storefront
#10	closed	feature, checkout	marcus	2026-02-20 11:40	Create orders from a guest's cart
#11	closed	bug, checkout, priority-high	marcus	2026-02-24 09:12	Reserve tickets while a guest completes payment
#12	closed	bug, catalog	marcus	2026-02-26 10:05	Calculate availability from active orders
#13	closed	performance, checkout	marcus	2026-03-01 14:25	Reduce the time required to create an order
#14	closed	feature, checkout, web	dana	2026-03-02 10:20	Include cart prices in order requests
#15	closed	feature, sales	marcus	2026-03-05 11:30	Allow one order for a large group
#16	closed	feature, admin	nadia	2026-03-09 09:50	Give the office a guest list and order ledger
#17	closed	performance, admin	marcus	2026-03-11 14:05	Remove the user lookup from admin authorization
#18	closed	bug, ops, blocker	sam	2026-03-30 15:50	Let the CDN-hosted storefront call the API
#19	closed	growth, auth	marcus	2026-04-01 09:20	Reduce signup failures at the password field
#20	closed	ops, auth	sam	2026-04-06 08:15	Unblock the login load test
#21	closed	feature, web, checkout	dana	2026-04-07 10:30	Complete the booking flow in the web app
#22	closed	feature, web	dana	2026-04-11 09:15	Build the public storefront homepage
#23	closed	feature, web, design	dana	2026-04-15 09:40	Add the Apex Island park map
#24	closed	feature, web, content	dana	2026-04-17 14:20	Publish visitor policies and cookie controls
#25	closed	task, web, ops	dana	2026-04-21 09:05	Prepare browser telemetry hooks
#26	closed	docs, test, launch	nadia	2026-04-23 09:30	Document launch information and test the main booking path
#27	closed	bug, web, priority-high	marcus	2026-04-27 08:50	Stop checkout when a date has sold out

What to include in a field report

- Ticket and changed code
- Unsafe behavior
- ASVS requirement
- Repair that preserves the requested feature



Let’s get started!

http://localhost:8082/

Related OWASP Projects You Should Also Care About

Awareness documents

- OWASP Top 10
- Top 10 Proactive Controls

Requirements for other domains

- MASVS
- ISVS
- SPVS
- AISVS

Testing guides

- WSTG
- MSTG

Other guidance documents

- Cheat Sheet Series
- SAMM

Mappings

- OpenCRE

OWASP Project Wayfinder

- https://owasp.org/projects/

It’s the LLMs’ world and we’re all just living in it

- Rigor is even more important for vibe coders!
- Even your favorite bot du jour needs standards to follow
- The faster we build, the more we need guardrails
- Keep an eye out for projects aiming to integrate ASVS with agents, MCP servers etc.

Contact and references

Eden Yardeni

eden.yardeni@owasp.org

eden@cronchie.dev

@cronchie

OWASP ASVS

@OWASP_ASVS

https://github.com/OWASP/ASVS

https://owasp.org/asvs

https://asvs.dev