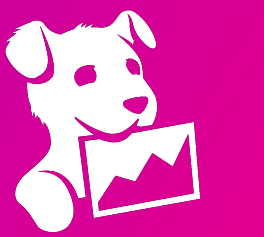
Follow the Trace:

How Traditional AppSec Tools Have Failed Us



DATADOG

Kennedy Toomey

Application Security Researcher & Advocate @ Datadog

Meet Trace

Objective

Let the journey begin…

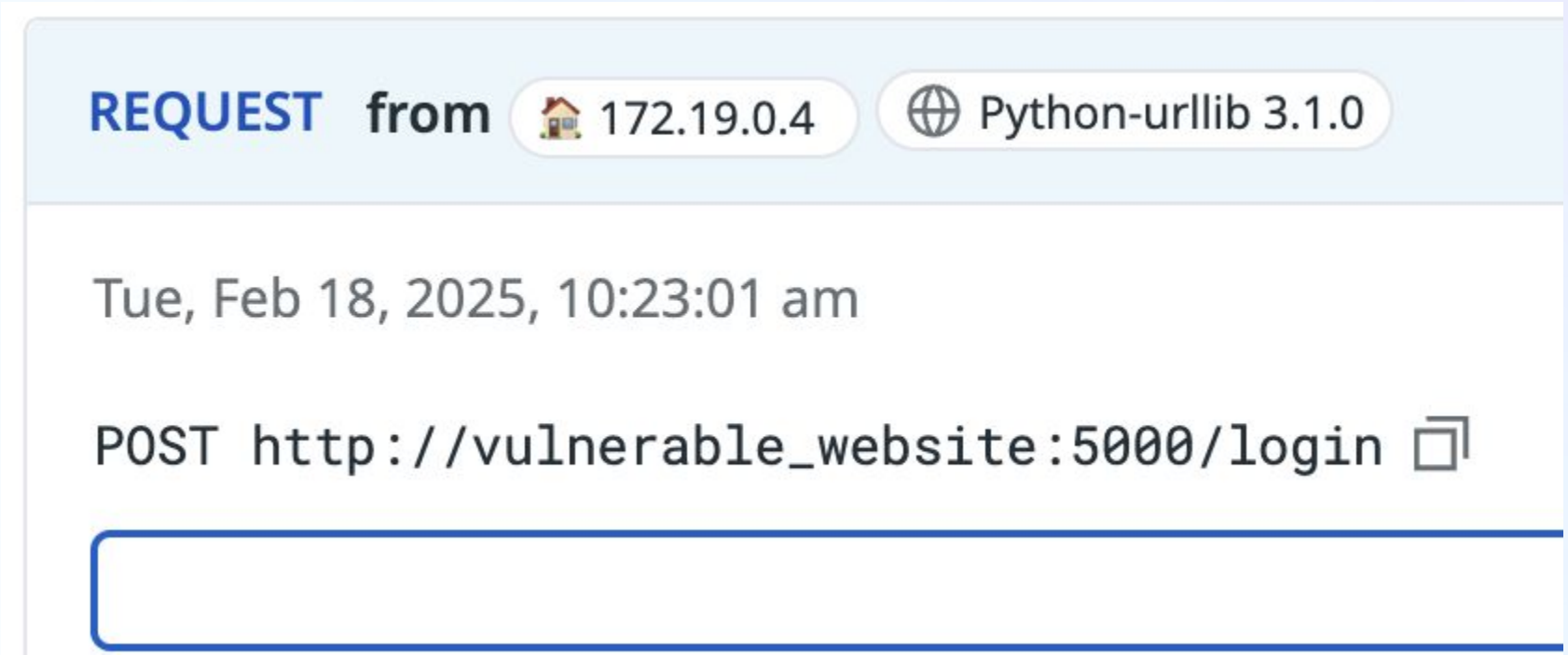
What is a trace?

Detailed record of an application's actions and events

Helps us understand the execution flow of an application

Can identify performance issues or pinpoint root cause of errors

Our First Request



Basic Login Page

Welcome Register Log In

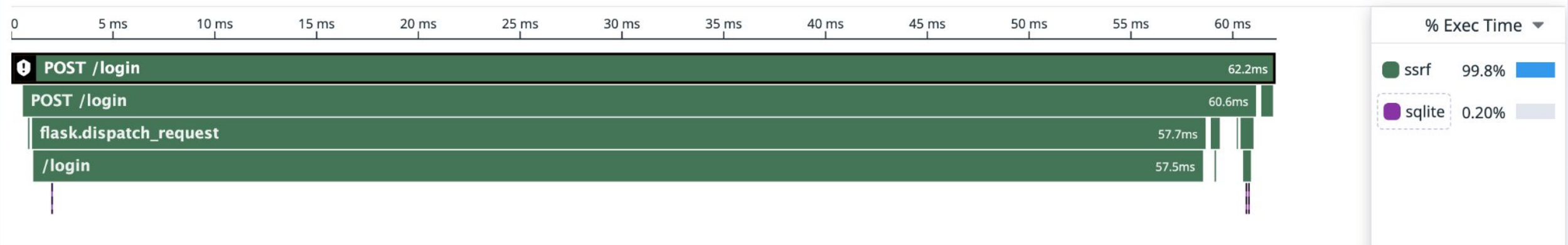
Log In

Username

Password

Log In

Login Flame Graph



Capturing Runtime Behavior & Interactions

Runtime behavior

- What your application is doing right now
- Tracks function calls and the code paths
- Latency, throughput, and error patterns

Interactions across components

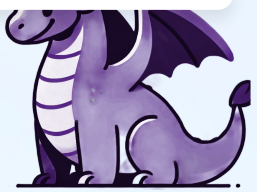
- Requests flowing through services
- Inbound & outbound traffic
- Distributed tracing

What gets captured?

- Timestamps
- Spans & traces
- Metadata

Why it matters

- Performance tuning
- Troubleshooting
- Dependency mapping
- Baseline for application behavior



Application Security Basics

Application Security Testing

SAST

S = Static

- Scans the code
  - Pattern matching

DAST

D = Dynamic

- Scans the running application
  - Simulates malicious attacks on the application

IAST

I = Interactive

- Analyses the running application's behavior
  - Follows the data flow

Application Layer Protections

WAF

Web Application Firewall

- Outside the application
  - Great: Threat Landscape Visibility
  - Good: DDoS Blocking
  - Bad: Exploit Prevention

RASP

Runtime Application Self-Protection

- Within the application
  - Great: Exploit Prevention
  - Great: Threat Landscape Visibility
  - Bad: DDoS Blocking

Runtime in Application Security

Runtime tools: IAST & RASP

Why Runtime?

Added context allow for more precise results

IAST tools provide significantly less false positives

IAST tools show the data flow which can help with reachability analysis

RASP tools can analyze traces realtime and identify attacks

Step-by-Step Adventure

Targeted Service

Welcome

Register Log In

Add New Blog

Optional: Link to a picture to add to your blog.

Test My Photo URL

Add Blog Post

Oh no…

We've encountered a threat!

The Threat 🔥

REQUEST from 172.19.0.4 Python Requests 2.32.0

Tue, Feb 18, 2025, 10:23:01 am

POST http://vulnerable_website:5000/test_picture_url

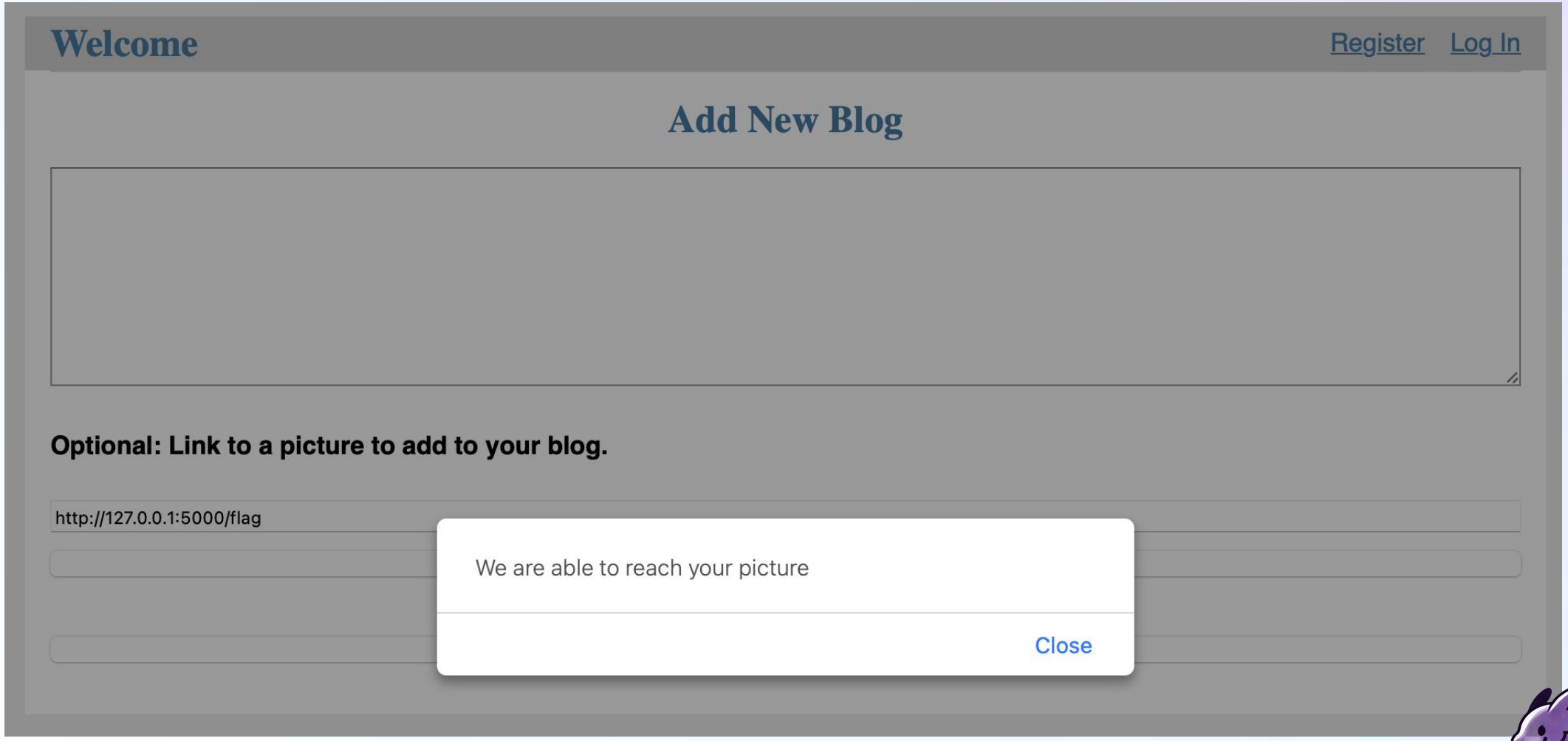
BODY

```
{
    url  http://127.0.0.1:5000/flag
}
```

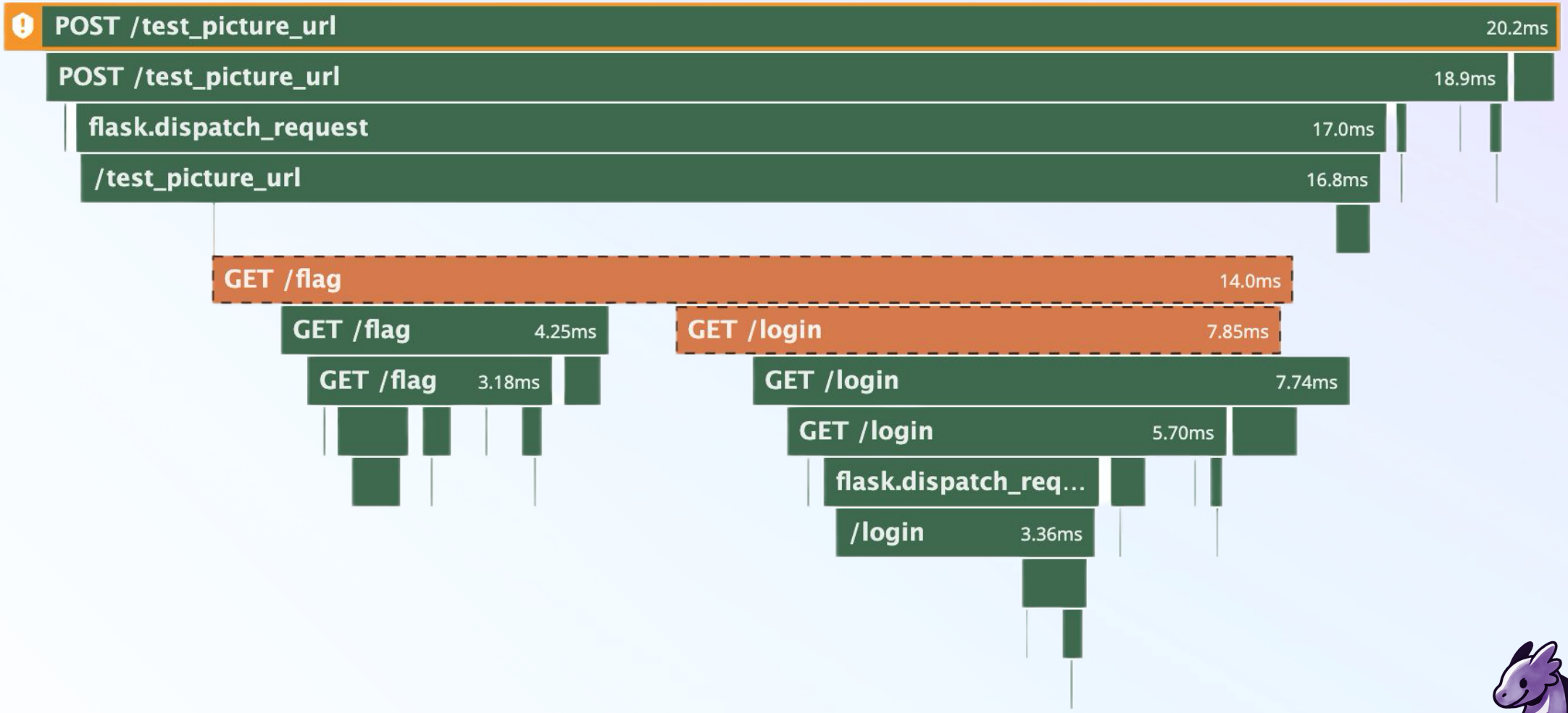
The Attack

```python
# Target URL
base_url = "http://vulnerable_website:5000/test_picture_url"

auth = HTTPBasicAuth(username, password)

payload = {
    "url": "http://127.0.0.1:5000/flag"  # valid url
}

try:
    response = requests.post(base_url, json=payload, auth = auth)
    print("Status Code:", response.status_code)
    print("Response:")
    print(response.text)
except requests.exceptions.RequestException as e:
    print("An error occurred:", e)
```

The Result


Welcome
Register
Log In
Add New Blog
Optional: Link to a picture to add to your blog.
http://127.0.0.1:5000/flag
We are able to reach your picture
Close

Flame Graph from the Attack



RASP to the Rescue!

What is a RASP looking for?

RASPs have a list of detection rules

Alerts when an incoming trace matches the pattern for a detection rule

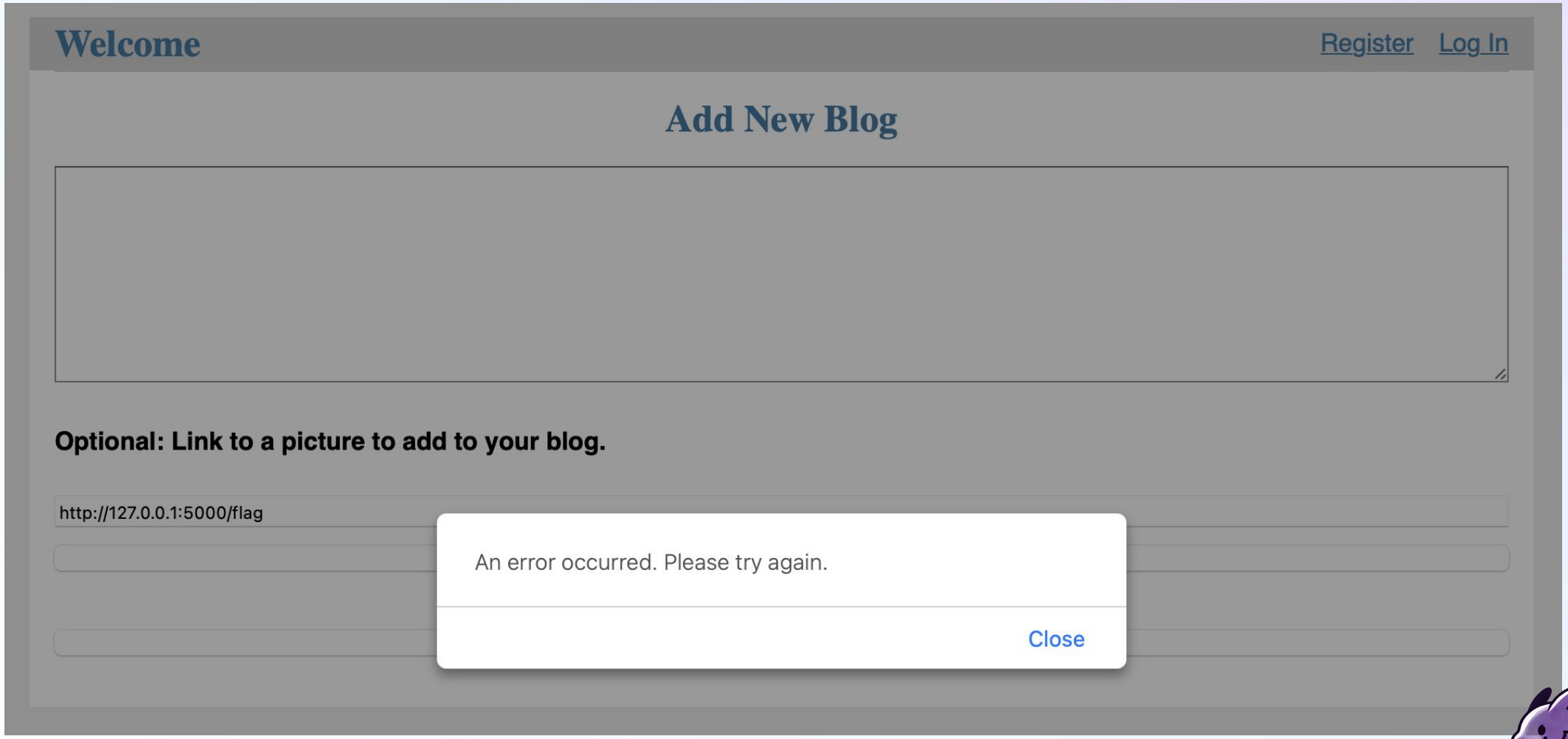
Adding a RASP

Requires configuration for best success

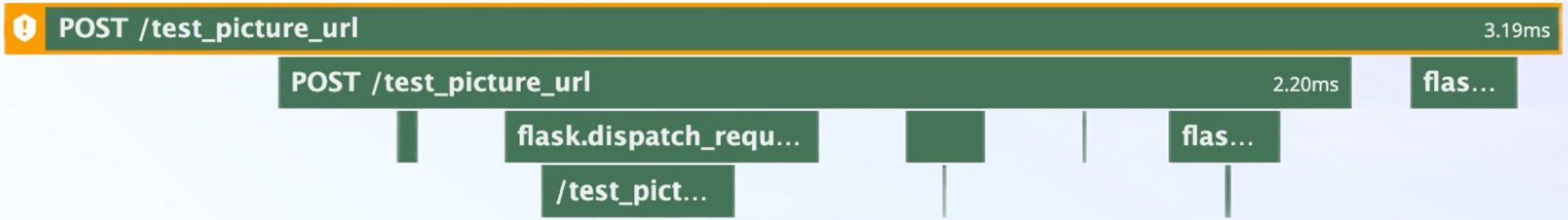
Monitor or block different vulnerabilities

For this use case, I decided to set my RASP to block all SSRF attacks and monitor all other attacks.

✋ Blocked by the RASP ✋



Flame Graph from the Blocked Attack



Security Insights

Context Makes All the Difference

Context puts less stress on developers

- Less false positives to investigate or fix

Context empowers security teams

- Provides the info need without having to rely on developers

Challenges & Opportunities

Limitations of Runtime Tools

Many tools require an agent running within the application

Setup is often more complex and continuous tweaking of rules is needed

Other tools are still required for early detection

The Evolving Runtime Security Landscape

Runtime appsec tools are becoming more available

Agentless tools are slowly emerging to allow for easier setup

Recapping our Journey

Runtime security is changing the game in application security

IAST Scans have the added advantage of runtime context

SAST scans still have a place in AppSec

Questions?

Resources

https://github.com/DataDog/application-security-reference-examples

Thank You!