Web Backend Hacking

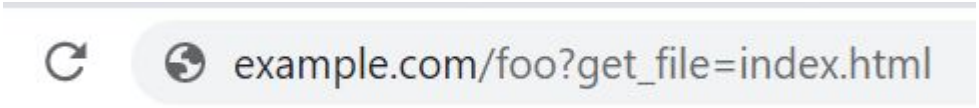
Path traversal & server-side request forgery

ankur & ian

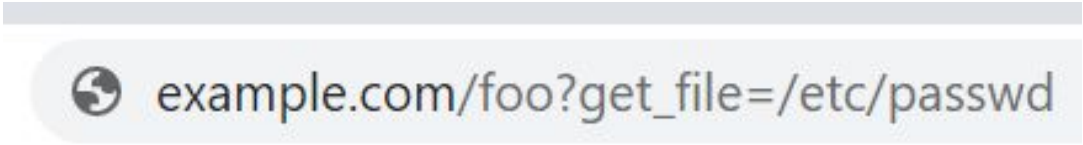
SIGPWNY

Path traversal

The idea:



The problem:



- Developers write a website with functionality to access files by path
- They forget to limit what paths you can access
- Outcome? We can read arbitrary files off of the filesystem

SIGPWNY

Challenge 1:

ssrf-1.chal.sigpwny.com

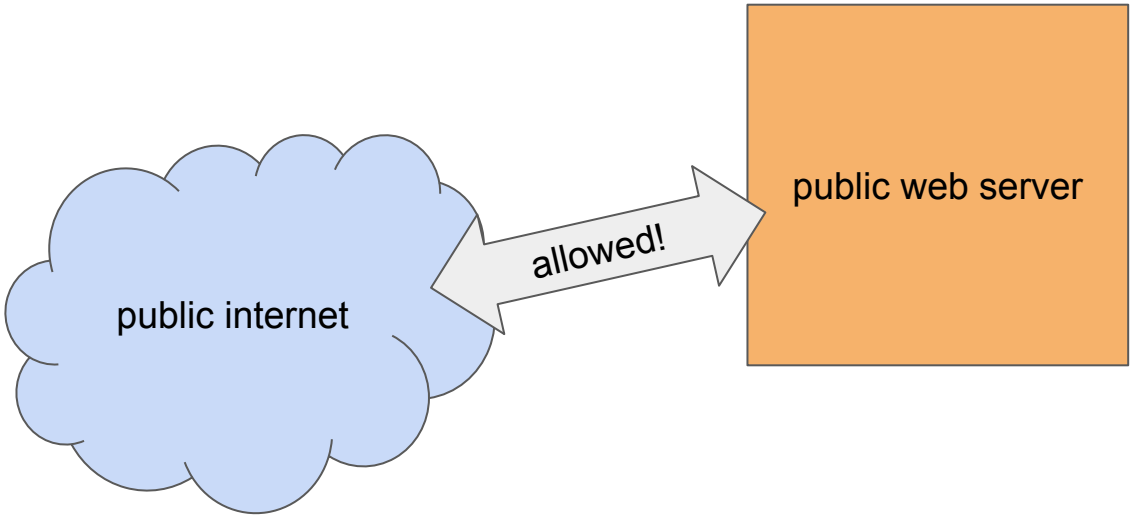
Can you get the flag? (in a file, **flag.txt**)

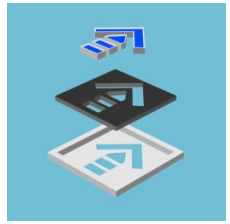
example.com/foo?get_file=/etc/passwd

2 minutes

SIGPWNY

SSRF: server side request forgery





SIGPWNY

SSRF: server side request forgery

internal network

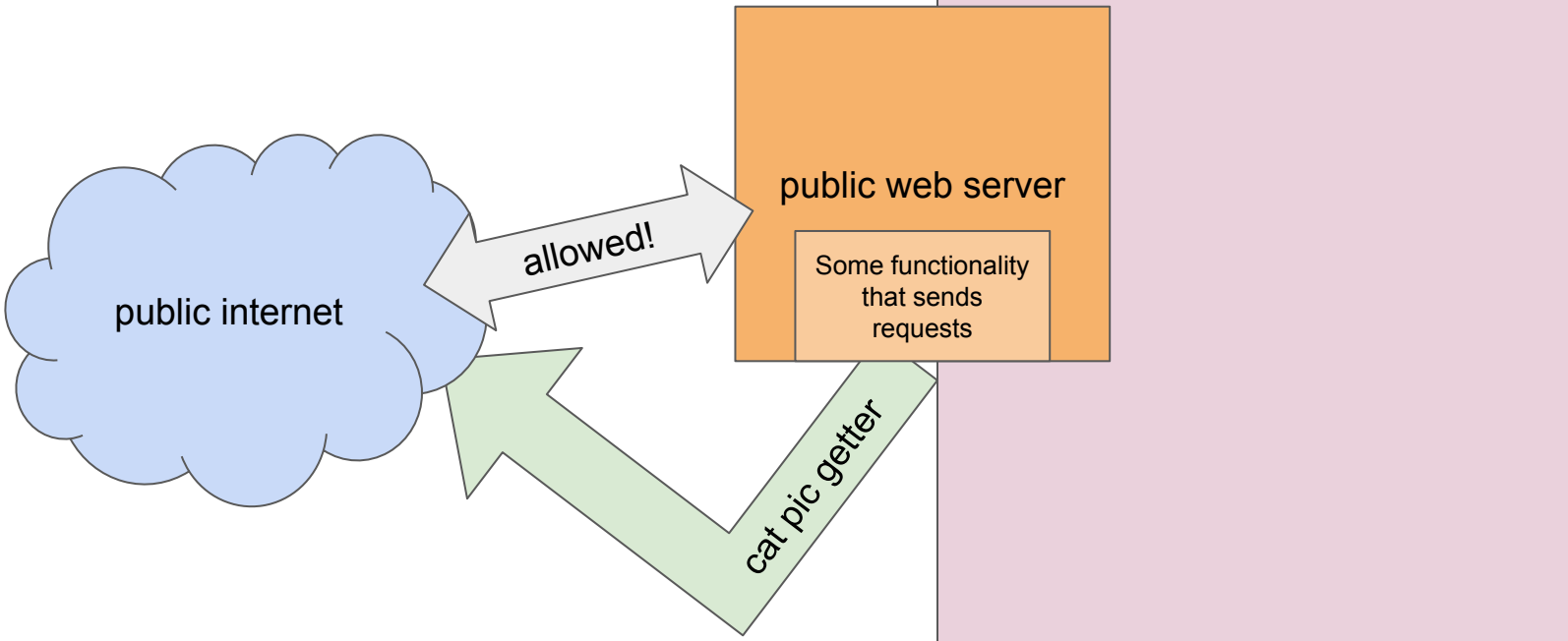
public web server

public internet

allowed!

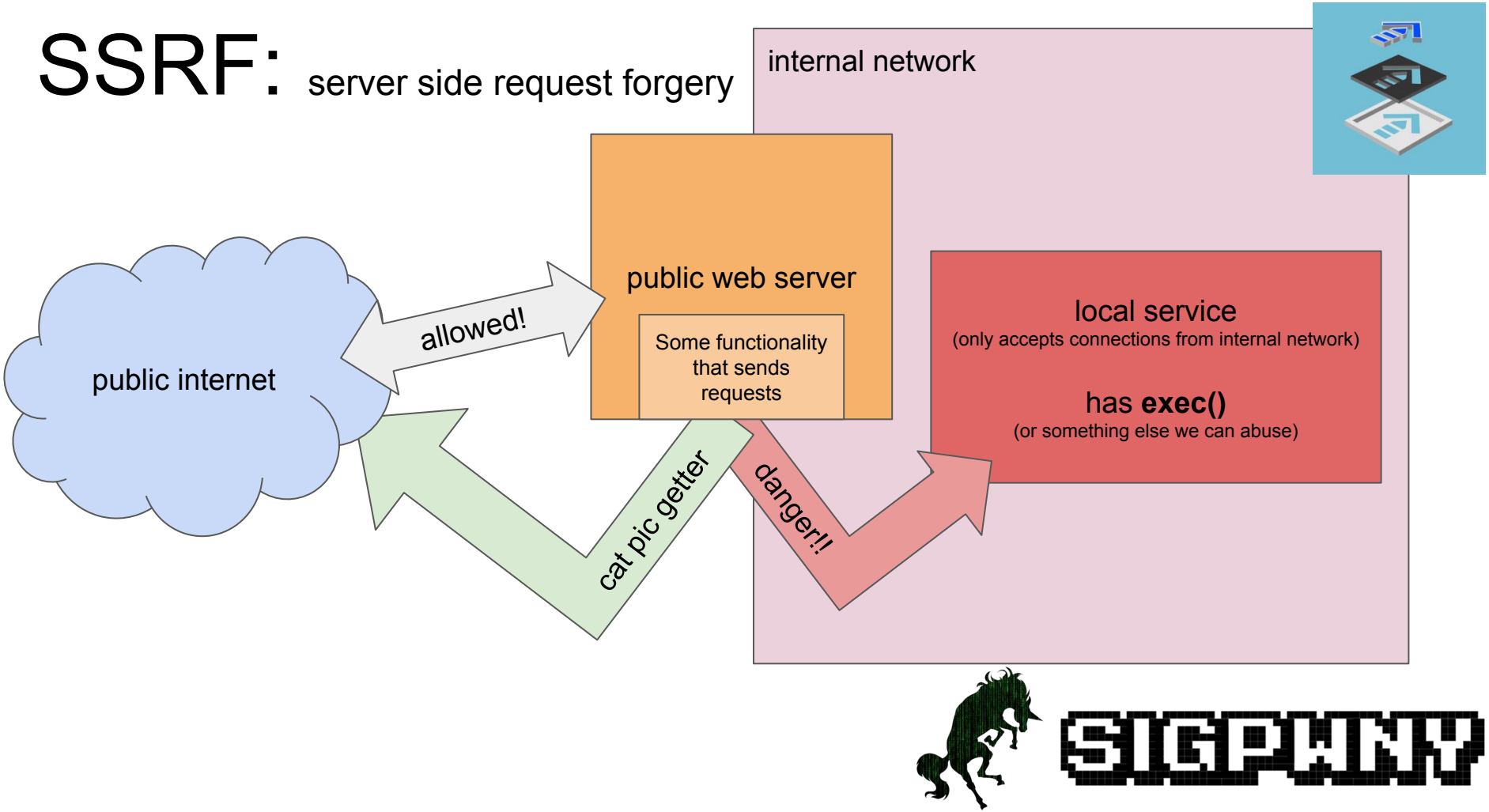
SIGPWNY

SSRF: server side request forgery



SIGPWNY

SSRF: server side request forgery



SIGPWNY

SSRF: the mistake

- Something that can make a request
  - open a socket and put bytes into it
- That functionality doesn't validate the address
- This allows access to private resources

how would a developer ever do this????? you might ask

SIGPWNY

Enter PHP: file_get_contents()

- This One Cool Trick Will Let You SSRF!
  - file_get_contents() lets you get:
    - Files
    - URLs
    - FTP
    - and more

- file:// — Accessing local filesystem
- http:// — Accessing HTTP(s) URLs
- ftp:// — Accessing FTP(s) URLs
- php:// — Accessing various I/O streams
- zlib:// — Compression Streams
- data:// — Data (RFC 2397)
- glob:// — Find pathnames matching pattern
- phar:// — PHP Archive
- ssh2:// — Secure Shell 2
- rar:// — RAR
- ogg:// — Audio streams
- expect:// — Process Interaction Streams

Functionality that causes SSRF

- Fetch image based on a URL
  - pass an internal URL -> SSRF
- Any sort of XML processing
  - XXE (XML External Entities)
- SVGs
  - can include links!
- Fetching an RSS feed

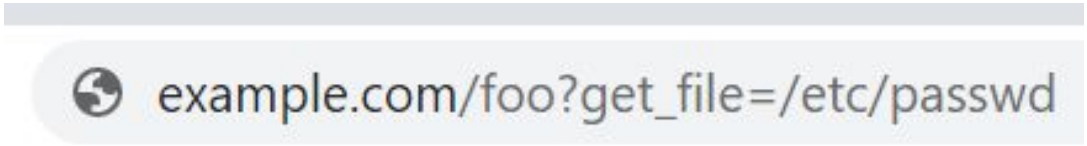
cat-fetcher.com/?file=http://cat-provider.com/cat1.jpg

SIGPWNY

Challenge 2:

ssrf-2.chal.sigpwny.com

Can you get the flag? (located at /flag.php)



5 minutes

SIGPWNY

A (naive) fix?

```
if ‘127.0.0.1’ or ‘localhost’ in request.url:
        reject_request()
```

but how could you bypass this check, hmm….?

what other ways are there to represent numbers in an IP address?

SIGPWNY

A proper fix

- Make an allowlist of URLs that are OK
  - Could denylist internal IPs, but there are problems...
- Preventing SSRF when you’re accepting arbitrary URLs:
  - complicated!
  - better to ask yourself if you *need* that functionality
  - if you *really* need it, OWASP Preventing SSRF

SIGPWNY

Got SSRF….. now what?

- Scan internal network
  - Find other services to abuse!
- get(“AWS_METADATA_ENDPOINT/secret_access_keys”)
  - haha now I can control your servers
  - How the Capital One breach happened
- Abuse other protocols that the URL handler supports
  - file:// - arbitrary file read
  - ldap:// - gather user information
  - (s)ftp:// - file read from other machines on the network
  - gopher:// - send ***arbitrary TCP packets***
- If you’re lucky, you can chain an SSRF into RCE

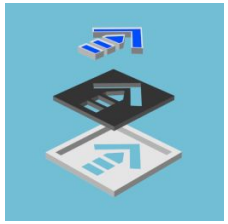
SIGPWNY

Got SSRF….. now what? (cont.)

- Elasticsearch
  - Widely used tool to index & search for documents
  - Exposes an internal HTTP API
- Redis
  - Caching technology
  - Takes RESP (REdis Serialization Protocol) over TCP
  - RESP commands are separated with \r\n… 🤔
- RCE on internal Gitlab via git:// protocol
  - https://liveoverflow.com/gitlab-11-4-7-remote-code-execution-real-world-ctf-2018/
- PHP phar://
  - Unserialize anything… (can lead to a deserialization chain)
- … and lots more!

SIGPWNY

in conclusion



- **ssrf - get a webserver to make requests on your behalf**
  - **gives you access to internal resources (stuff only accessible from localhost)**

now go do ssrf-3.chal.sigpwny.com! (and 4, and 5)

SIGPWNY

more modern web attacks if you're curious:

- read more:
  - https://portswigger.net/web-security/ssrf
  - https://book.hacktricks.xyz/pentesting-web/ssrf-server-side-request-forgery
- Some other classes of modern web bugs:
  - request smuggling - proxy & backend interpret request differently
    - serve a malicious response to an unsuspecting user
  - xs-search - infoleak through search functionality
  - none jwt, wrong jwt (RS256 vs HS256), forgeable jwt (key leaking)
  - oauth downgrade (pkce but strip the param so the server accepts)
- https://portswigger.net/daily-swig
  - Good blog, web security++

SIGPWNY