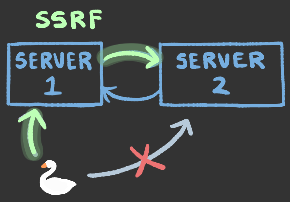
SSRF Server-Side Request Forgery

Vie

·Vulnerability that arises when multiple servers are talking to each other



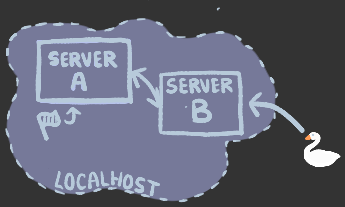
GENERAL PREMISE

·In a system w/ multiple servers in communication, and users can't talk to server 2. HOWEVER,
  -Users CAN talk to server 1
  -Server 1 CAN talk to server 2
·SSRF allows us to transitively talk to server 2, *through* server 1

·Most popular use case: making a server form an HTTP request to an arbitrary (usually private) domain

USE CASES

Localhost-only



·Server A can only accept connections from localhost
  ↳will reject all other connections
·Server B is front-facing (accepts all connections)
  ↳contains logic that makes B make connections of user's choosing
·Exploit: make B connect to A, request for flag

Plaintext Protocols

·Servers utilizing certain plaintext protocols may be good SSRF targets/gadgets
  -HTTP
  -FTP
  -SMTP
  -RESP
  -Memcached
  -Graphite
  -LDAP
  -file://
  -dict://
  -gopher

SSRF SPECIAL CASES

FTP BOUNCE

·FTP has 2 data channel modes: active (PORT) and passive (PASV)
  -Active: u specify location to establish data channel to, seperate from command channel
  -Passive: client establishes both channels
·You can use FTP as a vehicle of SSRF delivery - send arbitrary files containing commands, have FTP server yeet the file to another server.

GOPHER PROTOCOL

·old-ass protocol where you can specify the IP, port, and bytes - use this to exploit any TCP server you want.

·gopher://<ip>:<port>/<data in bytes>

SNI INJECTION (thanks Orange Tsai)

- TLS property: server name indication (sni)
- SNI is NOT encrypted in TLS, can be used to smuggle in other protocol commands

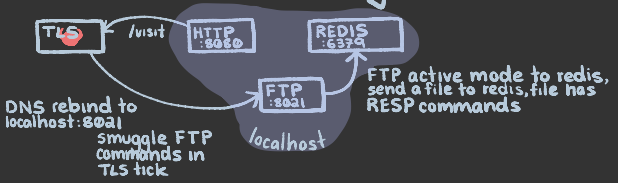
- Utilizes bugs in the HTTP server's url-parsing logic to do SNI-injection
  - ↳ nodeJS http parsing unicode chars weird
  - → Glibc NSS
- Orange Tsai: make SSRFs great again

TLS POISON (thanks Joshua Maddux)

- Fun w/ TLS part 2
- Context: session ids/tickets
  - ↳ TLS session caching: Client is given TLS session id/ticket post TLS-handshake, which maps to the client's session - if client reconnects later & presents the id/ticket, the TLS server can retrieve the session from the cache.

Combine DNS rebinding w/ sessional ids/tickets, and you can make an HTTP server send a sessional id/tick to a domain of your choosing (usually localhost).
- ↳ inject the tick w/ arbitrary commands

- Ex.) MapleCTF Art Gallery



- HTTP server connects to attacker HTTPS server, given a TLS session ticket, DNS rebind to make HTTP server connect to localhost:8021 - FTP server

AWS METADATA

- Memes w/ AWS EC2

- If a server running on AWS allows you to make arbitrary connections, you can use it to possibly connect to the AWS metadata service: 169.254.169.254