Week 11: Lecture B

Security in Practice: Tor

Thursday, November 9, 2023
Announcements

- **Project 3: WebSec** released
  - **Deadline:** tonight by 11:59PM!

---

**Project 3: Web Security**

**Deadline:** Thursday, November 9 by 11:59PM.

Before you start, review the course syllabus for the Lateness, Collaboration, and Ethical Use policies.

You may optionally work alone, or in teams of **at most two** and submit **one project per team**. If you have difficulties forming a team, post on Piazza's Search for Teammates forum. Note that the final exam will cover project material, so you and your partner should collaborate on each part.

The code and other answers your group submits must be entirely your own work, and you are bound by the University's Student Code. You may consult with other students about the conceptualization of the project and the meaning of the questions, but you may not look at any part of someone else's solution or collaborate with anyone outside your group. You may consult published references, provided that you appropriately cite them (e.g., in your code comments). **Don't risk your grade and degree by cheating!**

Complete your work in the **CS 4440 VM**—we will use this same environment for grading. You may not use any **external dependencies**. Use only default Python 3 libraries and/or modules we provide you.
Announcements

- **Project 4: NetSec** released
  - **Deadline:** Thursday, December 7th by 11:59PM

---

**Project 4: Network Security**

Deadline: Thursday, December 7 by 11:59PM.

Before you start, review the [course syllabus](#) for the Lateness, Collaboration, and Ethical Use policies.

You may optionally work alone, or in teams of at most two and submit one project per team. If you have difficulties forming a team, post on [Piazza's Search for Teammates](#) forum. Note that the final exam will cover project material, so you and your partner should collaborate on each part.

The code and other answers your group submits must be entirely your own work, and you are bound by the University's Student Code. You may consult with other students about the conceptualization of the project and the meaning of the questions, but you may not look at any part of someone else's solution or collaborate with anyone outside your group. You may consult published references, provided that you appropriately cite them (e.g., in your code comments). **Don't risk your grade and degree by cheating!**

Complete your work in the [CS 4440 VM](#) — we will use this same environment for grading. You may not use any external dependencies. Use only default Python 3 libraries and/or modules we provide you.
Announcements

See Discord for meeting info!

www.utahsec.com
Interested in automated bug-finding?

  - Everything you’d ever want to know about fuzzing for finding **security bugs**!
  - Course project: team up to fuzz a **real program** (of your choice), and find and report its bugs!
  - [https://cs.utah.edu/~snagy/courses/cs5963/](https://cs.utah.edu/~snagy/courses/cs5963/)
Last time on CS 4440...

Authentication
Multi-factor Authentication
One-time Passwords
Secure Password Storage
What is authentication?

- **What is it?**
  - That password you re-use for every website
  - An ever-changing set of rules to frustrate you
  - The most annoying thing about attending UofU

Sorry, but your password must contain an uppercase letter, a number, a hieroglyph, a feather from a hawk and the blood of a unicorn.

**WHY JUST 12 HOURS!?!?**
What is authentication?

- Goal: ???
- Problem: ???
- Challenge: ???
What is authentication?

- **Goal:** establish trust in the **identity** of another communicating party

- **Problem:** cannot directly interact with them to verify their identity

- **Challenge:** how can someone prove they are **who they say they are**?
The Three Factors of Authentication

- Something you ???
- Something you ???
- Something you ???
The Three Factors of Authentication

- **Something you have**
  - Smartphone
  - Laptop
  - Email account

- **Something you are**
  - Your fingerprint
  - Your DNA
  - Your iris, retina

- **Something you know**
  - Account password, banking PIN number
  - Nuclear strike challenge-response code
One-time PINs

- Provides proof of: ???
One-time PINs

- **Provides proof of:** possession
  - A PIN/code valid for only one login session or transaction

- **Delivering One-time PINs:**
  - ???
One-time PINs

- **Provides proof of:** possession
  - A PIN/code valid for only one login session or transaction

- **Delivering One-time PINs:**
  - **SMS**
    - Phone call
    - Text message
  - **Hardware**
    - Yubico YubiKey
    - RSA SecureID
  - **Application**
    - DUO Mobile
    - Google authenticator
Better idea: independently generate OTP codes based on a moving factor
- E.g., intervals of time, unique session count, etc.

Common OTP protocols:
- HMAC-based OTP (HOTP)
  - Use session count as factor
- Time-based OTP (TOTP)
  - Use time interval as factor

Problem: desynchronization
- E.g., user hits “login” one too many times
- Solution: make a few OTPs; user matches once

Pre-generated OTPs

<table>
<thead>
<tr>
<th>F</th>
<th>OTP</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>123456</td>
</tr>
<tr>
<td>2</td>
<td>124536</td>
</tr>
<tr>
<td>3</td>
<td>654321</td>
</tr>
<tr>
<td>4</td>
<td>321456</td>
</tr>
<tr>
<td>5</td>
<td>563412</td>
</tr>
</tbody>
</table>

Match
Biometrics

- Provides proof of ???
Biometrics

- Provides proof of **physical identity**
- **Something unique to you** (hopefully)
  - Fingerprint, iris, retina, DNA
- **Security = unlikely match probability**
  - Fingerprint match chance: 1 in $64 \times 10^{13}$
  - Iris pattern match chance: 1 in $10^{78}$
- **Trade-offs?**
  - Engineering effort, storage size, privacy concerns
Biometric Challenges

- **Replay attacks**
  - ???

- **Poisoning attacks**
  - ???

- **Noisy sensors**
  - ???

- **Change / loss of biometric**
  - ???

---

After an initial analysis, the Indian and American scientists used three iris sensors and two commercial iris biometric matchers to check if the new irises passed biometric authentication. They found that the iris sensors’ success rate dropped to 75% after surgery. The biometric matchers did better, authenticating 93% of the irises.

Crane horror *Reg* reader uses his severed finger to unlock Samsung Galaxy phone. **On the other hand he was fine.**
Biometric Challenges

- **Replay attacks**
  - Spoofs an enrolled user

- **Poisoning attacks**
  - Alter enrollment template
  - Alter one user’s enrollment

- **Noisy sensors**
  - Gives attackers “leeway” in crafting adversarial inputs

- **Change / loss of biometric**
  - **Change:** cataracts surgery
  - **Loss:** losing your finger

---

After an initial analysis, the Indian and American scientists used three iris sensors and two commercial iris biometric matchers to check if the new irises passed biometric authentication. They found that the iris sensors’ success rate dropped to 75% after surgery. The biometric matchers did better, authenticating 93% of the irises.

Crane horror Reg reader uses his severed finger to unlock Samsung Galaxy phone.
On the other hand, he was fine...
Facebook, Inc. has settled a class action that claimed Facebook collected and stored the biometric data of Facebook users in Illinois without the proper notice and consent in violation of Illinois law as part of its “Tag Suggestions” feature and other features involving facial recognition technology. Facebook denies it violated any law.
Proof of something you ????

Caution: Before entering your uNID or password, verify that the address in the URL bar of your browser is directing you to a University of Utah web site.

Important security information: This login uses cookies to provide access to the site you requested and to other protected University of Utah websites. For your security, log out of the services you are using and exit your browser when you have finished your session. Some browsers, including Google Chrome, retain cookie information by default even after you close your browser. Review your browser’s support documentation to set your browser to clear cookies automatically upon exit. Instructions for Google Chrome.
Passwords

- **Proof of something you know**
  - Something that you forget?

- **A secret** string of data that confirms a user’s identity
  - **Letters** (ABCDEFGHIJKLMNOPQRSTUVWXYZ)
  - **Digits** (0123456789)
  - **Other symbols** ($#%-_!)

- **Cryptographically secure?**
  - ???

---

**Login**

- **uNID**: (e.g. u86755209)
- **Password**

*Caution*: Before entering your uNID or password, verify that the address in the URL bar of your browser is directing you to a University of Utah web site.

*Important security information*: This login uses cookies to provide access to the site you requested and to other protected University of Utah websites. For your security, log out of the services you are using and exit your browser when you have finished your session. Some browsers, including Google Chrome, retain cookie information by default even after you close your browser. Review your browser’s support documentation to set your browser to clear cookies automatically upon exit. [Instructions for Google Chrome](#).
Passwords

- **Proof of something you know**
  - Something that you forget?

- **A secret** string of data that confirms a user’s identity
  - Letters (ABCDEFGH)
  - Digits (0123456789)
  - Other symbols ($#%-_.!)

- **Cryptographically secure?**
  - Not at all!

---

**Login**

- **uNID:** (e.g. u8675509)
- **Password:**

---

**Caution:** Before entering your uNID or password, verify that the address in the URL bar of your browser is directing you to a University of Utah web site.

**Important security information:** This login uses cookies to provide access to the site you requested and to other protected University of Utah websites. For your security, log out of the services you are using and exit your browser when you have finished your session. Some browsers, including Google Chrome, retain cookie information by default even after you close your browser. Review your browser’s support documentation to set your browser to clear cookies automatically upon exit. See [Instructions for Google Chrome](#).
Password Attacks

- **Passwords stored in plaintext**
  - ???

- **Passwords that are reused**
  - ???

- **Passwords that aren’t random**
  - ???

- **Device-issued default passwords**
  - ???

### Table: Sample Passwords

<table>
<thead>
<tr>
<th>Username</th>
<th>Password</th>
</tr>
</thead>
<tbody>
<tr>
<td>666666</td>
<td>666666</td>
</tr>
<tr>
<td>888888</td>
<td>888888</td>
</tr>
<tr>
<td>admin</td>
<td>(none)</td>
</tr>
<tr>
<td>admin</td>
<td>1111</td>
</tr>
<tr>
<td>admin</td>
<td>11111111</td>
</tr>
<tr>
<td>admin</td>
<td>1234</td>
</tr>
<tr>
<td>admin</td>
<td>12345</td>
</tr>
<tr>
<td>admin</td>
<td>123456</td>
</tr>
<tr>
<td>admin</td>
<td>54321</td>
</tr>
<tr>
<td>admin</td>
<td>7ujMko0admin</td>
</tr>
<tr>
<td>admin</td>
<td>admin</td>
</tr>
</tbody>
</table>

1 in 3 U.S. Pet Parents Have Used Their Pet’s Name as Their Password
Password Attacks

- **Passwords stored in plaintext**
  - Easily stolen if attacker breaches DB

- **Passwords that are reused**
  - Only takes one plaintext breach

- **Passwords that aren’t random**
  - Easily guessable via info about you

- **Device-issued default passwords**
  - Attacker can make one big dictionary

<table>
<thead>
<tr>
<th>Username</th>
<th>Password</th>
</tr>
</thead>
<tbody>
<tr>
<td>666666</td>
<td>666666</td>
</tr>
<tr>
<td>888888</td>
<td>888888</td>
</tr>
<tr>
<td>admin</td>
<td>(none)</td>
</tr>
<tr>
<td>admin</td>
<td>1111</td>
</tr>
<tr>
<td>admin</td>
<td>1111111</td>
</tr>
<tr>
<td>admin</td>
<td>1234</td>
</tr>
<tr>
<td>admin</td>
<td>12345</td>
</tr>
<tr>
<td>admin</td>
<td>123456</td>
</tr>
<tr>
<td>admin</td>
<td>54321</td>
</tr>
<tr>
<td>admin</td>
<td>7ujMko0admin</td>
</tr>
<tr>
<td>admin</td>
<td>admin</td>
</tr>
</tbody>
</table>
Assume attacker knows hash function and wants to find a single password

- Rapidly becoming more doable with advances in hardware!
Better Password Storage

- **Hashing passwords:** increases security by ???

- Why are weak hash functions bad?
  - ???

- Why are fast hash functions bad?
  - ???

![Hashing Process Diagram]

- password to hash function to hashed password
- Strawberry: SHA-256 to 5e737f891db1175442a39fd...
- Banana: SHA-256 to b493d48364afe44d11c016...
- Kiwi: SHA-256 to 1a5afeda973d776e31d1d72...
Better Password Storage

- **Hashing passwords**: increases security by **obfuscating passwords**

- Why are **weak** hash functions bad?
  - **Collision and pre-image attacks** = attacker easily finds working password

- Why are **fast** hash functions bad?
  - **Rainbow table attack** = attacker an efficiently pre-generate nearly all (password, hash) pairs
Better Password Storage

- **Salting passwords**: increases security via ???

- Examples of password **salts**:
  - ???

- Rainbow table attacks are ???
  - ???
Better Password Storage

- **Salting passwords**: increases security via more obfuscation

- Examples of password salts:
  - Injecting in a random string per user
  - Injecting user-selected password hint
  - Combos of hints and random strings

- Rainbow table attacks are **harder**
  - More entropy = orders-of-magnitude more chains to pre-compute!
Attack: Client-side Password Theft

- How?
Attack: Client-side Password Theft

How?
- Keyloggers, unencrypted transit, phishing, angry ex-partner
Forgetting and Recovering Passwords

- Security questions:
  - What’s your childhood pet?

- Password recovery email
  - Click here to reset your password!

- Send in plaintext to email
  - Your password is “in$3cur3”

Good security?
Forgetting and Recovering Passwords

- Security questions:
  - What’s your childhood pet?

- Password recovery email
  - Click here to reset your password!

- Send in plaintext to email
  - Your password is “in$3cur3”

Bad security! Attacker might have control of the victim’s email!
Forgetting and Recovering Passwords

- Security questions:
  - What’s your childhood pet?

- Password recovery email
  - Click here to reset your password!

- Send in plaintext to email
  - Your password is “in$3cur3”

- Other approaches:
  - Phone call
  - Session-specific PIN

**Bad security!** Attacker might have control of the victim’s email!

**Trade-offs?**
Authentication trade-offs?

Does anybody else find it kind of frustrating and disturbing that University of Utah students are required to have a smartphone to participate in classes? You can't access CIS, your UMail, or Canvas without using Duo's 2FA on your phone. If you lose your phone, if it gets damaged, or if it simply stops working you suddenly don't have the ability to turn in assignments. Duo also doesn't work on older devices. How many students have been unable to turn in their finals over this? Of course, you could email the helpdesk, but are you really going to do that every time you need to log in?

I can't believe this University charges this much money for such terrible infrastructure. The Wi-Fi barely works, you can easily get soft-locked out of your accounts, and they require you to own expensive devices just to attend. Everything is price gouged to hell. It's like going to school at a goddamn mall. What the hell are they wasting our tuition on?
Always be vigilant!

GoDaddy Breached – Plaintext Passwords – 1.2M Affected

There is an update available here: GoDaddy Breach Widens to tsoHost, Media Temple, 123Reg, Domain Factory, Heart Internet, and Host Europe

This morning, GoDaddy disclosed that an unknown attacker had gained unauthorized access to the system used to provision the company’s Managed WordPress sites, impacting up to 1.2 million of their WordPress customers. Note that this number does not include the number of customers who are already aware that their passwords are exposed. This includes some GoDaddy customers who have been attacked in the past.

Facebook Stored Hundreds of Millions of User Passwords in Plain Text for Years

March 21, 2019

Hundreds of millions of Facebook users had their account passwords stored in plain text and searchable by thousands of Facebook employees — in some cases going back to 2012, KrebsOnSecurity has learned. The issue has so far found no signs of data theft, but the company has been seeking to make sure that there is no indication.

Why Was Equifax So Stupid About Passwords?

Massive Credit Bureau Stored Users' Plaintext Passwords in Testing Environment

Mathew J. Schwartz (@europwofsec) · September 24, 2018
Always be vigilant!
Always be vigilant!
Questions?
This time on CS 4440...

Tor: The Onion Router
Internet Anonymity
Attacks on Tor
Project 4 Tips
What is Tor?

“Tor protects you by bouncing your communications around a distributed network of relays run by volunteers all around the world: it prevents somebody watching your Internet connection from learning what sites you visit, it prevents the sites you visit from learning your physical location, and it lets you access sites which are blocked.”
Tor’s Goal: Anonymity

- What is anonymity?
  - ???

- Versus confidentiality?
  - ???
Tor’s Goal: Anonymity

- What is *anonymity*?
  - I want to *say or do something* without the adversary knowing *that it was me* who said/did it

- Versus *confidentiality*?
  - Confidentiality = the contents
  - Anonymity = the identities

How/why does *anonymity* matter to *you*?
Why does internet anonymity matter?
How do the internet/web provide anonymity?

- **App Layer**: Application Message
  - Encrypted
- **Transport Layer**: Segment Header, Segment Data
  - Encrypted
- **Network Layer**: Packet Header, Packet Data
  - Unencrypted
- **Link Layer**: Frame Header, Frame Data, Frame Footer
  - Unencrypted
- **Physical Layer**: Bits Sent Over-the-Wire
  - Unencrypted
Even when you encrypt your packet data, the control data is still in-the-clear. Traffic analysis also reveals a great deal of info, because it focuses on the header, which must disclose source, destination, size, timing, and so on.
How do the internet/web provide anonymity?
How do the internet/web provide anonymity?

How can we maintain **anonymity** on the internet?
Tor: The Onion Router
Anonymity Primitive: Onion Routing

- Each message is **repeatedly encrypted**
  - **Analogy:** multiple layers of an onion
Anonymity Primitive: Onion Routing

- Each message is **repeatedly encrypted**
  - Analogy: multiple layers of an onion

- Sent through **multiple network nodes**
  - These nodes are called **onion routers**
  - Each node removes an encryption layer to uncover the message **routing instructions**
  - Process repeats when sent to next router
Anonymity Primitive: Onion Routing

- Each message is **repeatedly encrypted**
  - Analogy: multiple layers of an onion

- Sent through **multiple network nodes**
  - These nodes are called **onion routers**
  - Each node removes an encryption layer to uncover the message **routing instructions**
  - Process repeats when sent to next router

- **Anonymity:** prevents any intermediary nodes from knowing message **origin**, **destination**, and **contents**
Onion Routing Visualized

Sending data to a website

Client → Entry → Middle → Exit → Website
Onion Routing Visualized

Sending data to a website:

Client → Entry → Middle → Exit → Website

Receiving data from a website:

Client ← Entry ← Middle ← Exit ← Website
**Tor: The Onion Router**

- **Tor:** a distributed overlay network
  - Anonymizes TCP-based applications
    - Secure shell
    - Web browsing
    - Instant messaging
Tor: The Onion Router

- **Tor**: a distributed overlay network
  - Anonymizes TCP-based applications
    - Secure shell
    - Web browsing
    - Instant messaging

- Clients choose the **circuit paths**
  - Messages unwrapped at each onion router using a symmetric key
**Tor: The Onion Router**

- **Tor**: a distributed overlay network
  - Anonymizes TCP-based applications
    - Secure shell
    - Web browsing
    - Instant messaging

- Clients choose the **circuit paths**
  - Messages unwrapped at each onion router using a symmetric key

- Onion routers only know their **successor** or **predecessor** nodes
  - They don’t know of any other nodes
How Tor Works

Tor Client

Entry guard

Encrypted by Tor
Not encrypted by Tor

Tor Network

Middle relay

Exit relay

Destination
Trust in Tor

- **Entry node:** knows that Alice is using Tor as well as the identity of **middle node**
  - Does not know the destination!

- **Exit node:** knows a Tor user is connecting to the destination, but not **which** user

- **Destination:** knows that some Tor user is connecting to it via the exit node

- Tor does **not** provide encryption between the **exit node** and **message destination**
  - That is what **HTTPS** is for!
The Tor Network

- Lots of nodes spread out around the world
The Tor Network

- Lots of nodes spread out around the world
Questions?
Attacking Tor
Recap: The Domain Name System

- ???
Recap: The Domain Name System

- **Distributed database** implemented in hierarchy of many name servers

- **Application-layer protocol:**
  - Hosts and domain name servers communicate to resolve **domain names**
    - Address–name translation

- **Result:** user requests **domain name**
  - But their host really gets its **IP address**
  - Convenient!
- **DNS requests** are **not** sent through Tor by default.
- DNS requests are not sent through Tor by default

- Attackers could see what websites are being visited
### Attack 1: DNS Leaks

- **DNS requests** are **not** sent through Tor by default

- Attackers could see what **websites** are being visited

- **Fix:** external software can be used to reroute DNS via Tor
  - This is **not** default behavior
  - **Examples:** FoxyProxy, Privoxy
Brave browser’s Tor feature found to leak .onion queries to ISPs

Jessica Haworth 19 February 2021 at 14:27 UTC
Updated: 01 July 2021 at 16:27 UTC

Privacy  Dark Web  Browsers

Developers are issuing hotfix

UPDATED Brave, the privacy-focused web browser, is exposing users’ activity on Tor’s hidden servers – aka the ‘dark web’ – to their internet service providers, it has been confirmed.

Brave is shipped with a built-in feature that integrates the Tor anonymity network into the browser, providing both security and privacy features that can help obscure a user’s activity on the web.

Tor is also used to access .onion websites, which are hosted on the dark net.

Earlier today (February 19), a blog post from ‘Rambler’ claimed that Brave was leaking DNS requests made in the Brave browser to a user’s ISP.
Attack 2: Traffic Analysis

- ???
Attack 2: Traffic Analysis

- **Volume and Timing Analysis:**
  - Measure traffic going in/out of Tor network
  - Identify patterns to aid in reconnaissance
  - Identify likelihood you are accessing a page
Attack 2: Traffic Analysis

- **Volume and Timing Analysis:**
  - Measure traffic going in/out of Tor network
  - Identify patterns to aid in reconnaissance
  - Identify likelihood you are accessing a page

- **Examples:**
  - **Volume:** watch video vs. reading webpage
  - **Timing:** when you sent/received packets

- 11:30:11 Server sent 5kb
- 11:30:12 Your node received 6kb
- 11:33:17 Server sent 14kb
- 11:33:18 Your node received 15kb
**Attack 2: Traffic Analysis**

- **Volume and Timing Analysis:**
  - Measure traffic **going in/out** of Tor network
  - Identify patterns to aid in reconnaissance
  - Identify likelihood you are accessing a page

- **Examples:**
  - **Volume:** watch video vs. reading webpage
  - **Timing:** when you sent/received packets

- **Defenses:**
  - Intentionally adding noisy traffic
    - Cons: latency atop of latency
Attack 3: Malicious Nodes

- **Traffic leaving exit nodes** (e.g., a request to a website) is **unencrypted**
Attack 3: Malicious Nodes

- Traffic leaving exit nodes (e.g., a request to a website) is unencrypted
“Honey Onions” probe the Dark Web: at least 3% of Tor nodes are rogues

“If you control enough of the Tor network, it's possible to get a kind of bird's eye view of the traffic being routed through it.”
Questions?
Tor Users and Websites
Who uses Tor?

- ???
Who uses Tor?

- **Normal People**
  - Privacy-conscious folks

- **Intelligence Agencies**
  - Secret agents in the field

- **Law Enforcement**
  - Online “undercover” operations

- **Journalists and Bloggers**
  - Citizen journalists inspiring social change

- **Activists and Whistleblowers**
  - Raising their voice and avoiding persecution

- **White-hat and Black-hat Hackers**
  - And everyone in between!
Who uses Tor?

The anonymous Internet

Daily Tor users per 100,000 Internet users
- > 200
- 100 - 200
- 50 - 100
- 25 - 50
- 10 - 25
- 5 - 10
- < 5
- no information

Average number of Tor users per day calculated between August 2012 and July 2013

Data sources:
- Tor Metrics Portal: metrics.torproject.org
- World Bank: data.worldbank.org

by Mark Graham (@geoplaces) and Stefano De Sabbata (@maps4nothings)

Internet Geographies at the Oxford Internet Institute
2014 • geography.oi.ox.ac.uk

Oxford Internet Institute
University of Oxford

Stefan Nagy
Who uses Tor?

Internet censorship in the Arab Spring

From Wikipedia, the free encyclopedia

Main articles: Arab Spring and Internet censorship

The level of Internet censorship in the Arab Spring was escalated. Lack of Internet freedom was a tactic employed by authorities to quell protests. Rulers and governments across the Arab world utilized the law, technology, and violence to control what was being posted on and disseminated through the Internet. In Egypt, Libya, and Syria, the populations witnessed full Internet shutdowns as their respective governments attempted to quell protests. In Tunisia, the government of Zine El Abidine Ben Ali hacked into and stole passwords from citizens’ Facebook accounts. In Saudi Arabia and Bahrain, bloggers and “netizens” were arrested and some are alleged to have been killed. The developments since the beginning of the Arab Spring in 2010 have raised the issue of Internet access as a human right and have revealed the type of power certain authoritarian governments retain over the people and the Internet.
How can you use Tor?

Download Tor Browser
Protect yourself against tracking, surveillance, and censorship.
Hidden Services
Hidden Services

Tor Network

INP 1

INP 2

INP 3

Bob
Hidden Services

Tor Network

INP 1
INP 2
INP 3

Tor Database

“xyz.onion”
INP: INP-1

Bob
Hidden Services

Alice

Tor Network

INP 1

INP 2

INP 3

REN

Tor Database

Bob
Hidden Services

Alice

Tor Network

INP 1

INP 2

REN

INP 3

Bob

Tor Database

SCHOOL OF COMPUTING
UNIVERSITY OF UTAH
What services get hidden?
What services get hidden?
What services get hidden?

THIS HIDDEN SITE HAS BEEN SEIZED
by the Federal Bureau of Investigation,
in conjunction with the IRS Criminal Investigation Division,
ICE Homeland Security Investigations, and the Drug Enforcement Administration,
in accordance with a seizure warrant obtained by the
United States Attorney’s Office for the Southern District of New York
and issued pursuant to 18 U.S.C. § 983(j) by the
United States District Court for the Southern District of New York
Introducing DNS Resolver for Tor

06/05/2018

Mahrudd Sayrafi

In case you haven't heard yet, Cloudflare launched a privacy-first DNS resolver service on April 1st. It was no joke! The service, which was our first consumer-focused service, supports emerging DNS standards such as DNS over HTTPS:443 and TLS:853 in addition to traditional protocols over UDP:53 and TCP:53, all in one easy to remember address: 1.1.1.1.
Positive Tor Use Cases

Privacy is a human right

HANDS OFF MY DATA
Questions?
Project 4 Tips
Focuses on network packet analysis
  - Leveraging data contained within packets to achieve network defenses and attacks
Project 4 Overview

- Focuses on **network packet analysis**
  - Leveraging data contained within packets to achieve network defenses and attacks

- **Scenario:** helping a fictional university secure its enterprise campus network
  - Detect and characterizing likely attacks
  - Demonstrate how info can be intercepted
Project 4 Overview

- We provide a series of network packet traces (*pcaps*)
  - **Your job:** write scripts to analyze them!
Project 4 Overview

- We provide a series of network packet traces (pcaps)
  - **Your job**: write scripts to analyze them!

- **Part 1**: detecting **network attacks**
  - Password cracking, port scanning, SYN floods

- **Part 2**: stealing **sensitive information**
  - Unencrypted credentials, browsing history
  - **Extra credit**: stealing transferred files
Project 4 Overview

- We provide a series of network packet traces (**pcaps**)
  - **Your job:** write scripts to analyze them!

- **Part 1:** detecting **network attacks**
  - Password cracking, port scanning, SYN floods

- **Part 2:** stealing **sensitive information**
  - Unencrypted credentials, browsing history
  - **Extra credit:** stealing transferred files

- You will use Python 3’s **Scapy** library
  - A huge and powerful packet analysis API...
  - But we’ll really only use **a few parts** of it
Scapy Fundamentals

- Python API for programmatic packet capture and analysis
  - Think of it as “Wireshark in API form”
Scapy Fundamentals

- Python API for programmatic packet capture and analysis
  - Think of it as “Wireshark in API form”
- We provide skeleton code template
  - Sets-up the packet parsing workflow

```python
#!/usr/bin/python3
import logging
logging.getLogger("scapy.runtime").setLevel(logging.ERROR)
from scapy.all import *
import re

def parsePacket(packet):
    if not packet.haslayer("TCP"): return
    # TODO: finish implementing parsePacket()
    # ------------------------------------------
    return

if __name__ == "__main__":
    for packet in rdpcap(sys.argv[1]):
        parsePacket(packet)
```
Scapy Fundamentals

- Python API for programmatic packet capture and analysis
  - Think of it as “Wireshark in API form”

- We provide skeleton code template
  - Sets-up the packet parsing workflow
  - **Your job:** finish implementing the function `parsePacket()`

```python
#!/usr/bin/python3
import logging
logging.getLogger("scapy.runtime").setLevel(logging.ERROR)
from scapy.all import *
import re

def parsePacket(packet):
    if not packet.haslayer("TCP"):
        return
    # ---------------------------------------------
    # TODO: finish implementing parsePacket()!
    # ---------------------------------------------
    return

if __name__ == "__main__":
    for packet in rdpcap(sys.argv[1]):
        parsePacket(packet)
```
Scapy Fundamentals

- Python API for programmatic packet capture and analysis
  - Think of it as “Wireshark in API form”

- We provide **skeleton code** template
  - Sets-up the packet parsing workflow
  - **Your job:** finish implementing the function `parsePacket()`

- You may also add **additional code**
  - E.g., global variables or data structures
  - E.g., printing functionality in `main()`

```python
#!/usr/bin/python3
import logging
logging.getLogger("scapy.runtime").setLevel(logging.ERROR)
from scapy.all import *
import re

def parsePacket(packet):
    if not packet.haslayer("TCP"): return
    # -----------------------------------------------
    # TODO: finish implementing parsePacket()!
    # -----------------------------------------------
    return

if __name__ == "__main__":
    for packet in rdpcap(sys.argv[1]):
        parsePacket(packet)
```
Only a few things you’ll need…
Scapy Fundamentals

- Only a few things you’ll need...
  - Get a packet’s **TCP flags**: `packet["TCP"].flags`
Scapy Fundamentals

- Only a few things you’ll need...
  - Get a packet’s **TCP flags**: `packet[“TCP”].flags`
  - Get a packet’s **destination port**: `packet[“TCP”].dport`
Scapy Fundamentals

- Only a few things you’ll need...
  - Get a packet’s TCP flags:
    - `packet[“TCP”].flags`
  - Get a packet’s destination port
    - `packet[“TCP”].dport`
  - Get a packet’s source IP address
    - `packet[“IP”].src`
Scapy Fundamentals

Only a few things you’ll need...

- Get a packet’s **TCP flags**:
  ```python
  packet[“TCP”].flags  
  ```

- Get a packet’s **destination port**
  ```python
  packet[“TCP”].dport  
  ```

- Get a packet’s **source IP address**
  ```python
  packet[“IP”].src  
  ```

- Get a packet’s **TCP payload**:
  ```python
  bytes(packet[“TCP”].load).decode(‘utf-8’, ‘replace’)  
  ```

Scapy API reference

Scapy: create, send, sniff, dissect and manipulate network packets.

Usable either from an interactive console or as a Python library. [https://scapy.net](https://scapy.net)

**Subpackages**
- scapy.ansimachine
- scapy.ans_resolvers
- scapy.and_fields
- scapy.auxpacket
- scapy.auton
- scapy.bases_classes
- scapy.config
- scapy.consts
- scapy.diffields
- scapy.data
- scapy.error
- scapy.fields
- scapy.interfaces
- scapy.main
- scapy.packet
- scapy.plisttool
Only a few things you’ll need...

- Get a packet’s **TCP flags**:
  ```python
  packet["TCP"].flags
  ```

- Get a packet’s destination port:
  ```python
  packet["TCP"].dport
  ```

- Get a packet’s source IP address:
  ```python
  packet["IP"].src
  ```

- Get a packet’s TCP payload:
  ```python
  bytes(packet["TCP"].load).decode('utf-8', 'replace')
  ```

All of the targets can be solved using a few **fundamental Scapy objects**!
Suggested Workflow

- Before you start writing a **Scapy** script, inspect the trace *manually* via **Wireshark**
  - Super helpful for viewing a packet’s contents
  - Use this to bootstrap your script’s approach!
Suggested Workflow

- Before you start writing a **Scapy** script, inspect the trace *manually* via **Wireshark**
  - Super helpful for viewing a packet’s contents
  - Use this to bootstrap your script’s approach!

- For each target, answer the following:
  - What **packet fields** matter?
  - How to **extract** relevant data?
  - How to **store and process** this data?
Suggested Workflow

- Before you start writing a Scapy script, inspect the trace manually via Wireshark
  - Super helpful for viewing a packet’s contents
  - Use this to bootstrap your script’s approach!

- For each target, answer the following:
  - What packet fields matter?
  - How to extract relevant data?
  - How to store and process this data?

- Finalize your high-level game plan first!
  - Then start developing your solution scripts!
Questions?
Next time on CS 4440...

Adversarial Machine Learning