IT A14 Assignment:

By Deakin Fisher

Table Of Contents

Summary and Outline of Assessment Brief

Design and submit an educational **Capture-the-Flag (CTF) style challenge** demonstrating your understanding of network administration and security. You may create either a:

- **Virtual Machine (VM) based challenge**, or
- **Packet Tracer challenge** using the Activity Wizard.

Your challenge must demonstrate technical accuracy, educational value, and secure design principles. Challenges will be deployed on CTFd.

Deliverables

1. **Challenge Deployment Video**
   A 1–2 minute screen recording showing how your challenge is set up.
   - VM: Show script and command process.
   - Packet Tracer: Show Activity Wizard and initial setup steps.
2. **Challenge-Solving Video**
   A 2–3 minute recording demonstrating the challenge being solved.
   - Narrate or subtitle the steps to reach the solution.
   - Showcase tools, clues, or techniques used to capture the flag.
3. **Deployment Script**
   A Bash script or written instructions that set up and remove the challenge.
   - Must be commented and tested in a clean environment.
4. **Challenge Document (Markdown)**
   A brief challenge overview suitable for uploading to a CTF platform.
   - Include title, category, points, description, hints, and flag format.
5. **Tutorial Document (Markdown)**
   A peer-facing document explaining the concept behind your challenge.
   - Include background theory, guidance for solving, and optional references or walkthroughs.
6. **Evidence Guide (Slides)**
   A visual and written portfolio (15–25 slides) responding to five reflection questions.
   - Include speaker notes, code snippets, screenshots, diagrams, and annotations.

Criterion/Expectations

Recording of the Challenge Deployment

- A 1 - 2 minute screen recording showing the setup or deployment of your challenge.
- For VM: show your script, command process and setup steps.
- For Packet Tracer: show the activity wizard and initial state/configuration steps.
- Clearly explain what the challenge is and what is being set up

Recording of the Challenge Solved

- A 2 - 3 minute screen recording demonstrating the challenge being solved by a test user (yourself or a peer).
- Narrate, subtitle, or script the steps taken by a test user (yourself or a peer)
- Narrate or subtitle the steps taken to reach the solution.
- Showcase key steps, tools, or clues required to find or capture the flag.
- Optional: pause to explain moments of complexity or clever design

Deployment Script

- A literal, bash, or instructions that set up and remove the challenge.
- Includes clear comments for each step
- Must cleanly reset or remove the challenge environment
- Should be tested on a clean VM or Packet Tracer File

Challenge Document (Markdown)

- A brief challenge overview suitable for uploading to CTFd.
- Include title, category, points, description, hints, and intended flag.
- Make sure it’s clear but not overly revealing.

Tutorial Document (markdown)

- A learning-focused document for peers or junior students.
- Explains the concept behind your challenge (e.g., what is a cron job exploit?)
- Includes basic background, what learners should know before attempting, and guidance for solving
- Optional: include reference links or simplified walkthroughs.

Evidence Guide (Slides)

- Collate written, visual, and annotated evidence to support responses to reflection questions.
- Title Slide: Student name, team name, robot name (if applicable), challenge focus
- 3–5 slides per question (total: approx. 15–25 slides)

Each question's slides should include:

- A written response (totally no more than 200–300 words) to the question (in the speaker's notes)
- Visual evidence: diagrams, code snippets, photos, sketches, test data, etc.
- Annotated elements explaining what each piece of evidence shows

---

Reflection Questions:

Question 1:

How did your design process evolve from initial concept to final deployment, and what constraints or opportunities influenced your decisions along the way?

Evidence to consider:

- Annotated design sketches or planning documents
- Version history of scripts or Packet Tracer files (e.g., early, mid, final)
- Screenshots of issue tracking, notes on bugs or dead ends
- Diagrams showing architecture changes over time
- Comments in code reflecting iterative decisions (e.g., "replaced file scan loop for performance")

A	B	C	D	E
Critically analyses the design process with insightful evaluation of opportunities, constraints, and decision-making; supports response with comprehensive and annotated evidence.	Analyses the design process and clearly explains how opportunities and constraints influenced decisions; uses a range of relevant evidence.	Explains the design process with reference to decisions and constraints; includes appropriate supporting evidence.	Describes major elements of the design process with minimal connection to constraints or decisions; some evidence provided.	Identifies basic steps of the design process with little or no reference to reasoning; minimal or unclear evidence.
Analyses the design process and clearly explains how opportunities and constraints influenced decisions; uses a range of relevant evidence.	Explains the design process with reference to decisions and constraints; includes appropriate supporting evidence.	Describes major elements of the design process with minimal connection to constraints or decisions; some evidence provided.	Identifies basic steps of the design process with little or no reference to reasoning; minimal or unclear evidence.	Identifies some steps of the design process with little or no justification. Shows limited evidence of planning or consideration of constraints. Evidence may be vague or incomplete.

Question 2:

In what ways did your challenge demonstrate secure or insecure practices in network, host, or application configuration? What trade-offs did you need to manage?

Evidence to consider:

- Snippets of firewall rules, ACLs, or SSH config
- Screenshots of vulnerability in action (e.g., flag capture via exploit)
- Before-and-after comparisons of secure vs. insecure settings
- A diagram or table showing what was exposed, hidden, or protected
- Markdown excerpt explaining trade-offs (e.g., ease of access vs realism)

A	B	C	D	E
Critically analyses secure/insecure practices with detailed technical insight; evaluates trade-offs with clear justification and thorough supporting examples.	Analyses secure/insecure practices with technical clarity; explains trade-offs and uses sound examples.	Describes secure/insecure practices with reference to design; explains trade-offs with relevant evidence.	Identifies security features or flaws with some description of impact; limited discussion of trade-offs.	Mentions security concepts with minimal explanation; lacks understanding of trade-offs or supporting evidence.
Analyses secure/insecure practices with technical clarity; explains trade-offs and uses sound examples.	Describes secure/insecure practices with reference to design; explains trade-offs with relevant evidence.	Identifies security features or flaws with some description of impact; limited discussion of trade-offs.	Mentions security concepts with minimal explanation; lacks understanding of trade-offs or supporting evidence.	Mentions basic security ideas but lacks clarity or detail. Fails to show understanding of secure setup or risk trade-offs. Supporting examples are minimal or incorrect.

Question 3:

How does your challenge connect to real-world network administration scenarios, and what broader insights can it offer about digital infrastructure or cybersecurity?

Evidence to consider:

- Real-world news article or incident with a similar vulnerability
- Short paragraph summarising relevant CVE or protocol weakness (if applicable)
- Diagram linking their challenge to common infrastructure (e.g., SSH in enterprise)
- Comparison chart showing what their challenge simulates or simplifies
- Link to tutorial or documentation that inspired their design

A	B	C	D	E
Critically analyses real-world relevance with sophisticated connections to infrastructure and cybersecurity insights.	Analyses real-world applications with clear connections and supporting examples.	Describe how the challenge relates to real-world systems with appropriate examples.	Identifies some relevance to real-world scenarios with general or superficial links.	Provides minimal or unclear connections to real-world scenarios.
Analyses real-world applications with clear connections and supporting examples.	Describe how the challenge relates to real-world systems with appropriate examples.	Identifies some relevance to real-world scenarios with general or superficial links.	Provides minimal or unclear connections to real-world scenarios.	Provides limited or unclear connection to real-world examples. Response may be off-topic, overly general, or lacking supporting evidence.

Question 4:

What technical strategies, tools, or methodologies did you apply to manage, secure, or troubleshoot systems during development, and why were they appropriate?

Evidence to consider:

- Command history logs or excerpts (`history | grep`)
- Script snippets with inline comments
- List of tools used and why (`find`, `grep`, `iptables`, Packet Tracer tools, etc.)
- Screenshots of testing or debugging output
- A flowchart or a list of troubleshooting steps taken during development

A	B	C	D	E
Applies technical strategies and tools with control and precision; justifies appropriateness and effectiveness in context.	Applies strategies and tools with confidence; explains effectiveness clearly.	Uses appropriate tools and techniques; describes why they were used.	Identifies some tools or techniques with a basic description of function.	Shows limited understanding or application of tools or strategies.
Applies strategies and tools with confidence; explains effectiveness clearly.	Uses appropriate tools and techniques; describes why they were used.	Identifies some tools or techniques with a basic description of function.	Shows limited understanding or application of tools or strategies.	Demonstrates limited use or understanding of tools. May name the tools used but not explain their purpose or effectiveness—minimal or incorrect technical evidence.

Question 5:

Reflect on your personal contribution to the challenge. How did you plan, manage time, collaborate (if applicable), and develop or adapt your thinking throughout the project?

Evidence to consider:

- Screenshots of to-do lists, schedules, or Kanban boards
- Journal/logbook entries showing time management or delays
- Annotated feedback from peers or teachers
- Slides showing a comparison between the initial goal and the outcome
- Brief self-assessment with strengths, challenges, and what was learned

A	B	C	D	E
Reflects insightfully on learning, time management, and collaboration; clearly articulates personal growth and adaptation.	Analyses learning and contribution with good detail and reflection.	Describes own role and learning; explains some challenges and changes.	Identifies own role and tasks with minimal reflection.	Provides minimal or vague reflection on own contribution.
Analyses learning and contribution with good detail and reflection.	Describes own role and learning; explains some challenges and changes.	Identifies own role and tasks with minimal reflection.	Provides minimal or vague reflection on own contribution.	Gives a vague or superficial reflection on their role. Little insight into planning, problem-solving, or learning. A few examples are provided to support the response.

Submission Guidelines

	Submission Guidelines			
Elegance and brevity of work	Your work is presented with **clarity, precision, and efficiency**, focusing on essential details without unnecessary complexity. Explanations, diagrams, and designs are **concise yet comprehensive**, demonstrating **thoughtful organisation and effective communication**. Your final product reflects a **streamlined approach** that balances **technical accuracy with user-friendly presentation**.			

A	B	C	D	E
Presents ideas with clarity, precision, and efficiency. Explanations are streamlined, accurate, and well-organised with no redundancy.	**Communicates clearly and efficiently. Most content is purposeful and well-structured, with only minor excess or lack of**	**Content is mostly explicit and includes necessary information, though some sections may be verbose, disorganised, or**	**Ideas are communicated with limited clarity. Presentation may be cluttered, inconsistent, or lacking structure.**	**The work lacks clarity and is poorly organised. Explanations are often excessive, confusing, or overly simplistic, failing to focus on key**

Diagrams and code are concise yet comprehensive.	**clarity.**	**underdeveloped.**		**details.**
Communicates clearly and efficiently. Most content is purposeful and well-structured, with only minor excess or lack of clarity.	**Content is mostly explicit and includes necessary information, though some sections may be verbose, disorganised, or underdeveloped.**	**Ideas are communicated with limited clarity. Presentation may be cluttered, inconsistent, or lacking structure.**	**The work lacks clarity and is poorly organised. Explanations are often excessive, confusing, or overly simplistic, failing to focus on key details.**	**Work shows basic or inconsistent communication. Explanations may be unclear, incomplete, or overly detailed without relevance. The presentation lacks focus or organisation, making it difficult to follow.**

Overall submission quality	Your submission is **well-organised, cohesive, and polished**, meeting all project requirements. Each component—**research, presentation, conceptual design, and recorded video**—is thoughtfully executed, demonstrating **technical accuracy, critical thinking, and creativity**. Your work reflects a **high standard of professionalism**, effectively showcasing your **understanding, problem-solving skills, and ability to synthesise complex information**.

A	B	C	D	E
Submission is highly polished, well-integrated, and demonstrates a deep understanding of technical concepts. All components exceed expectations in presentation, accuracy, and creativity.	**Submission is complete, accurate, and well-presented. Shows clear technical understanding and thoughtful execution across all components.**	**Submission meets the basic requirements with mostly accurate content and structure. Some areas may lack depth or cohesion.**	**Submission is incomplete or uneven in quality. Technical errors or lack of detail may hinder clarity or coherence.**	**Submission is disorganised or lacks required components. Shows minimal understanding or effort across multiple sections.**
Submission is complete, accurate, and well-presented. Shows clear technical understanding and thoughtful execution across all	**Submission meets the basic requirements with mostly accurate content and structure. Some areas may lack depth or cohesion.**	**Submission is incomplete or uneven in quality. Technical errors or lack of detail may hinder clarity or coherence.**	**Submission is disorganised or lacks required components. Shows minimal understanding or effort across multiple sections.**	**Submission is incomplete or lacks clarity. Several required components may be missing or attempted at a basic level. The**

components.				**work demonstrates limited understanding and minimal effort in presentation or execution.**

	DAYS LATE ___/7 = ___%			**FINAL**	

Project Plan - Web Exploitation CTF Challenge

Project Aim

To simulate a real-world example of exploiting vulnerabilities within a webserver, using a range of OpenSSL tools, to decrypt and extract secure information. It aims to simulate a multi-step process, demonstrating skills that are both grounded in realism, and within the scope of possibility at a college level.

Timeline & Milestones Plan

Day	Milestone	Tasks	Evidence to Capture
Day 1	**Planning & Setup**	- Read VM challenge brief - Draft initial file discovery strategy -Consult with system administrators and let them help design initial processes and CTF processes. -What real world application can the system admin give which I can apply into my CTF	- Outline creation of Virtual Machine using fedora - Sketch initial plans
Day 2	**Initial Scripting & Testing**	- Download and install UTM for mac and run fedora as VM -Begin writing scripts	- Code version 1 (commented) - Screenshot of command-line test results - Notes on bugs/misfires

Day 3	**Continue scripts**	Continue on scripts, making sure to stick to the guidelines given the time frame	- Save bash scripts, make sure to write down any and all initial problems
Day 4	**Secrets File**	- complete the secrets file and make sure that the script runs it adequately and is able to generate unique checksum each time	- Possible table comparing secure/insecure configuration - Design process, save bash script, begin mental notes for walkthrough (Day 7).
Day 5	**Real-World Mapping & Research**	- Find 1–2 real incidents of information leakage - Connect challenge to enterprise and ask system administrator -Create model to overcome obstacles for slides (reflection question) - Compile list of all problems	- Diagram or markdown linking your challenge to real systems - Any notes made when consulting with the system administrator.
Day 6	**Documentation & Evidence**	- begin and complete markdown doc, and information for challenge in cookbook - Create general script for voiceovers/walkthroughs of the code for tomorrow(MAKE SURE TO KEEP THEM) - Show each obstacle/challenge during the course of scripting and outline it with adherence to the previously stated	- Markdown file with structure: Introduction, Tools, Execution, Reflections - Screenshots, diagrams, comments

		model.	
Day 7	**Create Videos Finalise Everything**	- Record short voiceover reflecting on process of creating secret flag using script - Record Voiceover of the creation of the bash scripts. - Record walkthrough of CTF challenge -Finalise markdown information document.	- Deployment video file - 5–6 reflection slides for class or submission - Finalise slides

---

Folder & File Structure

```
bushranger-discovery/
├── README.md
├── scripts/
│   ├── find_flags.sh
│   └── final_flags.sh
├── logs/
│   └── discovery_log.txt
├── evidence/
│   ├── screenshot1.png
│   ├── early_script.png
│   ├── final_script.png
│   └── perms_table.md
├── reflections/
│   ├── Q1_design.md
│   ├── Q2_security.md
│   ├── Q3_realworld.md
│   ├── Q4_tech.md
│   └── Q5_self.md
```

Plan To Answer Reflection Questions well

Question	Plan to Answer	Evidence to Capture
Q1: Design Process	Keep notes/diagrams of early designs and scripts. Save “version 1”, “version 2”, and final version. Explain what was improved and why.	Code comments, screenshots, architecture sketch, inline notes
Q2: Secure/ Insecure Practices	Use a folder with bad permissions. Capture differences in `ls -l` output, or simulate hidden files.	Table/diagram of secure vs insecure setup. Markdown doc on trade-offs
Q3: Real-World Relevance	Link to a real case (e.g. data leak from poor permissions). Explain how your setup mirrors a known risk.	CVE summary, news clip, diagram connecting to SSH or enterprise backup servers
Q4: Technical Tools	List tools and explain why they were chosen over others. Add snippets showing logic.	Snippets + screenshots. Flowchart of troubleshooting steps
Q5: Reflection	Keep a log or bullet journal per day. Write 3–4 dot points of what worked, what didn’t, what was learned.	Logbook, to-do list screenshot, peer feedback if any, strengths vs challenges table