

chapter one

What a Capture the Flag Is

A capture-the-flag competition presents security challenges inside an authorized environment. Each challenge asks a player or team to recover a secret string called a **flag**. Submitting the expected string to the scoreboard records a solve and awards points according to the event's rules.

The challenge description defines the intended target: perhaps a supplied file, a temporary web application, or a network service created for the event. Permission is limited to that target and the methods allowed by the rules. The scoreboard, hosting platform, other players, and unrelated systems remain outside scope.

The loop is: inspect the provided target, recover the flag, and submit it.

CTF organizers usually isolate challenge data from production systems, but no online platform can promise that risk is literally zero. Follow the event's rules, avoid unnecessary traffic, and report unexpected platform defects through the stated channel rather than testing them further.

The flag

A flag is a string chosen by the challenge author and recognized by the scoreboard. It often includes an event prefix and braces. This book uses:

```
brewCTF{y0u_found_the_secret}
```

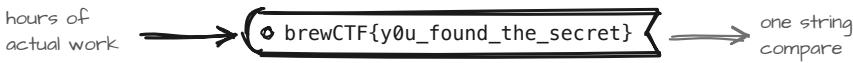
picoCTF challenges commonly use the same structure with another prefix:

```
picoCTF{s0me_real_flag_here}
```

Recognize the structure, but copy the complete value exactly as the challenge produced it.

The prefix, opening brace, contents, and closing brace are all part of the submitted value. Other events use different formats, so the challenge prompt and rules are authoritative.

a recovered flag is expensive to produce and cheap to check



Recovery can take hours; the scoreboard performs a string comparison.

two examples with the same structural parts

ours `brewCTF{y0u_found_the_secret}`

Diagram: A horizontal line with two vertical tick marks. The segment between the first tick mark and the start of the flag is labeled "prefix". The segment between the two tick marks is labeled "challenge-specific contents".

theirs `picoCTF{s0me_real_flag_here}`

Diagram: A horizontal line with two vertical tick marks. The segment between the first tick mark and the start of the flag is labeled "prefix". The segment between the two tick marks is labeled "challenge-specific contents".

The braces are part of the submitted value.

A scoreboard can compare a submitted string cheaply even when recovering it required substantial analysis. Both examples contain a prefix, braces, and challenge-specific contents at different character offsets.

A flag demonstrates possession of the expected value; by itself, it does not document how the value was obtained. After an event, a write-up can supply the missing evidence: observations, failed hypotheses, commands, and the reasoning that connected them.

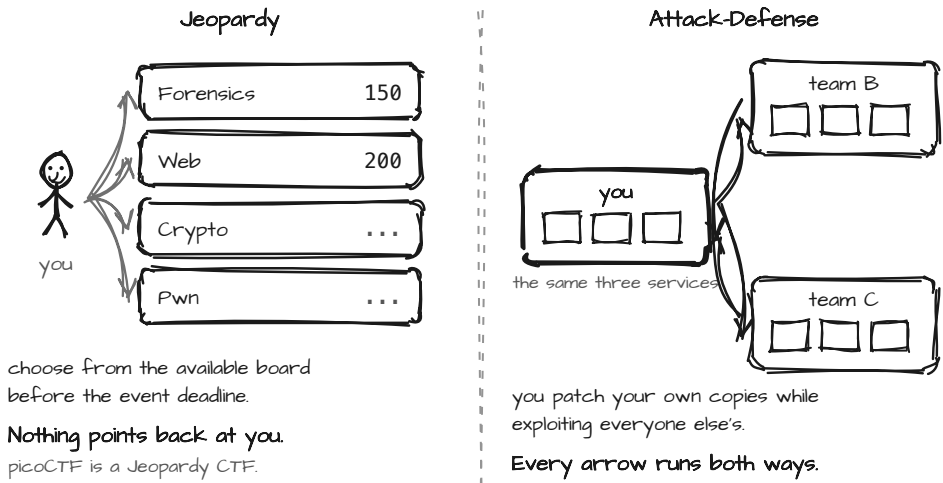
The two main formats

Two common competition formats organize challenges differently.

In a **Jeopardy-style CTF**, the event presents a board of mostly independent challenges grouped by category. Players choose challenges, submit flags, and accumulate points before the event deadline. Some boards unlock later challenges after prerequisites; others allow any order. picoCTF competitions use this general format.

In an **Attack-Defense CTF**, teams operate equivalent vulnerable services. They patch their own instances while attempting to retrieve flags from other teams' instances under a tightly specified ruleset. Scoring can include both service availability and successful attacks. The format assumes experience with system administration, rapid patching, and coordinated team work.

Jeopardy-style events remove live defense and are usually easier places to learn the categories.



A Jeopardy board connects a player to independent challenge targets. In Attack-Defense, permitted traffic runs between team services in both directions, and each team must also maintain its own services.

Aspect	Jeopardy-style	Attack-Defense
Structure	Challenge board	Live team services
Incoming attacks	Normally none	Part of the format
Service defense	Normally none	Required
Timing	Event deadline	Continuous rounds
This book	Primary format	Context only

Categories

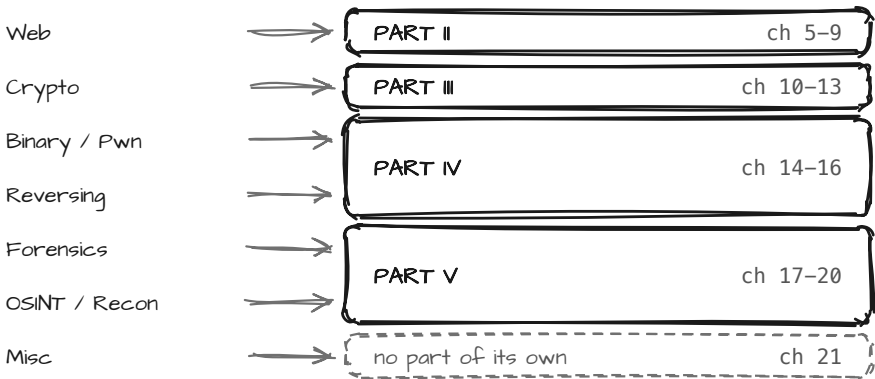
Category names vary between events, but the following labels are common:

Category	Typical task
Web	Analyze a deliberately vulnerable site or API
Crypto	Identify a weak construction or implementation
Binary / Pwn	Analyze memory behavior in a supplied program
Reversing	Infer behavior from a compiled program
Forensics	Recover and interpret artifacts from files
OSINT / Recon	Answer a bounded question using public sources
Misc	Solve tasks outside the other groupings

Use the category to choose an initial tool, then follow evidence from the target.

These categories overlap. A web challenge may require decoding, a forensic artifact may contain network traffic, and a reversing challenge may depend on cryptographic code. The label is a starting clue, not a guarantee about the solution.

seven categories, and where this book teaches each



Misc groups tasks that do not fit the other event categories.

The figure maps the chapter's seven category labels to the parts that introduce them. Binary/Pwn and Reversing share Part IV; Forensics and OSINT share Part V; Misc appears in the final walkthrough rather than receiving a separate part.

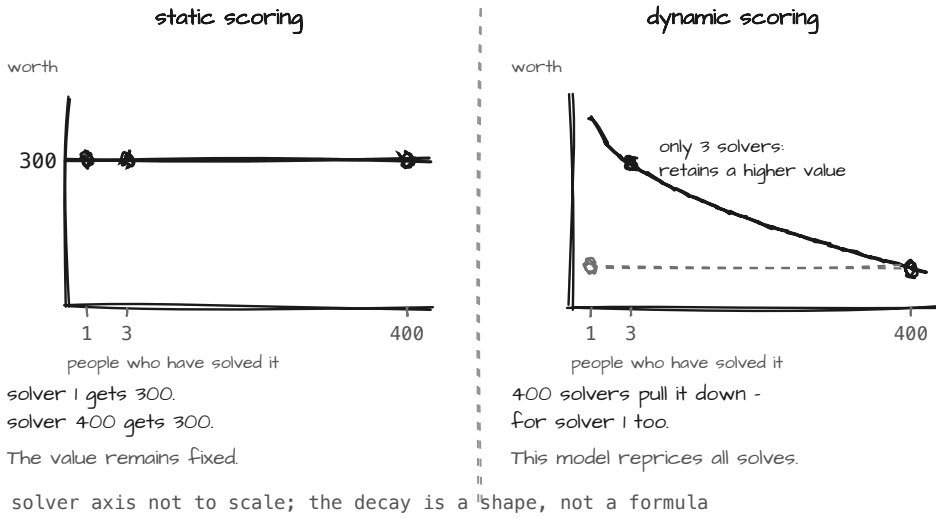
Beginners do not need equal skill in every category. Broad familiarity helps with triage, while repeated work in one or two categories develops depth.

Scoring and first solves

With **static scoring**, a challenge retains a fixed point value. Every valid solve receives that value unless the event rules specify bonuses or penalties.

With **dynamic scoring**, a challenge's value depends on a formula that can include solve count, minimum value, and decay parameters. Some platforms recalculate the value for all solvers as the count changes; others award the value that applied at solve time. The event rules and scoreboard implementation determine which behavior is in use.

A point value has meaning only with the event's scoring formula and current solve count.



The left panel shows a fixed 300-point value. The right panel illustrates one repricing model in which a challenge falls as solve count rises and earlier solvers receive the current value. It is a structural example, not a universal decay formula.

Some events mark the first valid solve as **first blood** and may attach a bonus. Others display no special marker. It can be useful event information, but it does not change the technical requirements of the challenge.

Confirm whether first solves affect points or are only displayed.

```
[ SCOREBOARD ]
web/login-bypass ..... first: h4wk
crypto/weak-rsa ..... first: mochi
forensics/lost-photo .. first: you
```

How challenges are delivered

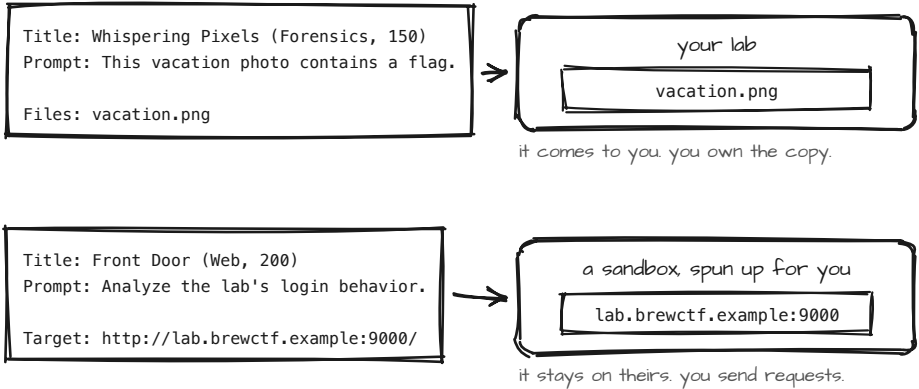
A file-based challenge supplies an artifact to analyze locally, such as an image, executable, packet capture, archive, or ciphertext.

```
Title: Whispering Pixels (Forensics, 150)
Prompt: This vacation photo contains a flag.
Files: vacation.png
```

A service-based challenge supplies an address for an isolated instance. Send only the kinds of requests permitted by the event rules.

Title: Front Door (Web, 200)
Prompt: Analyze the lab's login behavior.
Target: `http://lab.brewctf.example:9000/`

If the organizer did not identify it as a target, treat it as outside scope.



In both cases, the challenge card identifies the authorized target.

A file challenge moves a supplied copy into the player's lab. A service challenge remains in an organizer-controlled sandbox and receives authorized requests. In both cases, the challenge card identifies the target.

Challenge instances can still share infrastructure with a platform. Stay within the supplied host, port, account, and time window. If an observation appears to expose another player's data or a platform component, stop and use the event's reporting process.

Solving records and write-ups

Being unable to solve a challenge immediately is normal, but difficulty alone does not guarantee learning. Progress comes from making testable observations: identify the artifact, record the baseline behavior, change one input, and compare the result.

When stuck, return to the prompt and list facts separately from assumptions.

Write for a reader who has the same artifact but not your unstated assumptions.

After an event or after its rules permit disclosure, players often publish **write-ups**. A useful write-up records the challenge version, target type, tools, relevant commands, intermediate evidence, final method, and defense or lesson. Do not publish flags or solutions while an event prohibits sharing.

Good sportsmanship follows the explicit rules: do not attack event infrastructure, interfere with other participants, share prohibited flags, or continue testing beyond the named target. Chapter 2 develops authorization and scope in more detail.

Exercises

Exercise 1.1. For each blurb, name the most likely category and identify the clue: (a) a photograph contains hidden data; (b) a lab login trusts an unsafe input; (c) every byte was combined with one repeating key; (d) a supplied compiled program checks a password. Name one reason each initial category could still overlap another.

Exercise 1.2. An event gives every team equivalent vulnerable services and requires teams to patch their copies while retrieving flags from opponents. Name the format, list two responsibilities absent from a typical Jeopardy challenge, and explain why a beginner might start with Jeopardy instead.

Exercise 1.3. A dynamic board lists challenges at 480 and 120 points. Explain why those values alone reveal neither solve count nor difficulty, and list the scoring information needed to interpret the difference.

Exercise 1.4. Outline a write-up with scope, evidence, solution, and defensive lesson. Name two technically possible actions forbidden by the event's rules.