

KAPE and EZ Tools

FOR201

Collect the artifacts that answer the question, prove the collection matched what you specified, drive the parsers that make it readable, and read what comes back. Targets and modules are text files you write, test and version. Built on the 2026 licensing position, which most published material predates.

Modules	15
Lessons	70
Phases	5
Free preview	Course Orientation

The method

Triage forensics is a decision about what to collect before you know what happened. This course runs the same loop for every artifact class it covers.

1. Collect narrow and fast, or broad and slow. KAPE targets decide this and the decision is irreversible on a host that gets rebuilt. Know what each target set actually reaches.
2. Separate collection from processing. Collect once, process many times. Processing on the live host wastes the one chance you had at the evidence.
3. Parse with the tool built for the artifact. Each Eric Zimmerman tool exists because a format needs it. Reading an artifact with the wrong parser produces plausible output and wrong answers.
4. Corroborate across artifact classes. Execution evidence in one artifact is a lead. The same execution in three artifacts, with consistent timestamps, is a finding.
5. Build the timeline last. A timeline assembled before the artifacts are understood inherits every parsing error in it.

What this course assumes

No minimum experience and no prerequisite course. The Windows artifacts, what each records and why it exists are explained where first used.

What makes it go faster: a Windows machine you can run the tools on and an image or a live system to practice against. Neither is required, and the course provides output for every artifact it examines.

What this course does not cover: memory forensics, malware analysis, and cloud investigation. This is disk triage on Windows, taught deeply, with the tooling that does it fastest.

Full syllabus

Course Orientation

Course Orientation FREE PREVIEW

What this course teaches: how KAPE collects and processes in two separate phases, how to build targets and modules that fit your investigation, which EZ Tool answers which question, and how to read the output well enough to write a defensible finding.

Phase 1: The Toolkit

Triage and the Toolkit

When triage collection is the right answer and when it is the wrong one, what KAPE cannot reach, the licensing line that decides whether you may run it at all, how to install EZ Tools so the tools actually launch, how KAPE is laid out on disk and how it finds its own configuration, how to...

- 1.1 What Triage Collection Buys You, and What It Costs
- 1.2 Licensing, and the Decision It Forces
- 1.3 Building the EZ Tools Toolkit
- 1.4 KAPE on Disk
- 1.5 Collection Media and Destination
- 1.6 Your First Collection, Verified
- Module Summary
- Check My Knowledge

Phase 2: Running KAPE

KAPE Collection

The target file format field by field, how paths, masks and recursion combine to select files, what a published triage target actually collects rather than what its name implies, writing and testing a target of your own, shadow copies and containers, and the full target-side switch set with the...

- 2.1 The Target File
- 2.2 Paths, Masks and Recursion
- 2.3 What a Published Triage Target Actually Collects
- 2.4 Writing and Testing Your Own Target
- 2.5 Shadow Copies and Containers
- 2.6 The Collection Switch Set, and Proving It Worked
- Module Summary
- Check My Knowledge

KAPE Processing and Scale

The module file format and how a definition drives a program, the processing phase and its defaults, what the compound processing module actually runs, writing modules of your own, processing collections somebody else produced, and taking the whole pipeline to a fleet.

- 3.1 The Module File
- 3.2 The Processing Phase
- 3.3 What the Compound Processing Module Runs
- 3.4 Writing Your Own Module
- 3.5 Processing Collections You Did Not Take
- 3.6 At Scale
- Module Summary
- Check My Knowledge

Phase 3: Reading the Artifacts

File System

Reading the file system as evidence: what a master file table record holds, the two sets of timestamps and why one of them resists tampering, parsing with MFTECmd, the change journal, the supporting metadata structures, and what the file system can and cannot establish.

1. 4.1 The Master File Table
2. 4.2 The Two Timestamp Sets
3. 4.3 Parsing the Table
4. 4.4 The Change Journal
5. 4.5 The Supporting Structures
6. 4.6 What the File System Proves
7. Module Summary
8. Check My Knowledge

Execution and Resource Use

Four sources that partially answer what ran on a machine, what each actually proves, why their timestamps mean different things, and how to reconcile them into a defensible statement about execution.

1. 5.1 What Execution Actually Means
2. 5.2 Prefetch
3. 5.3 ShimCache
4. 5.4 Amcache
5. 5.5 Resource Usage
6. 5.6 Reconciling the Sources
7. Module Summary
8. Check My Knowledge

Registry and User Activity

The artifacts that record what a person did rather than what a process did: registry hives and how to read them properly, devices and volumes, shortcut files, jump lists, shellbags, and what user activity evidence actually establishes.

1. 6.1 The Registry as Evidence
2. 6.2 Accounts, Devices and Volumes
3. 6.3 Shortcut Files
4. 6.4 Jump Lists
5. 6.5 Shellbags
6. 6.6 Search, Execution and What It Proves
7. Module Summary
8. Check My Knowledge

Phase 4: To a Finding

Event Logs, Timelines and Findings

The only artifact written deliberately as a record: parsing event logs, reading authentication and sessions, joining process creation to the session that ran it, persistence and privilege, what a cleared log looks like, and taking one endpoint from a collection command to a written finding.

1. 7.1 Event Logs as Deliberate Records
2. 7.2 Authentication and Sessions
3. 7.3 What Ran, and Who Ran It
4. 7.4 Persistence and Privilege
5. 7.5 Anti-Forensics and Gaps
6. 7.6 From Collection to Finding
7. Module Summary
8. Check My Knowledge

Phase 0: Course Resources

Cheatsheets

The licensing position, collection and processing syntax, the artifacts and what each proves, and where each one is silent.

1. What You May Run, and How to Collect
2. Processing, and the Tools That Do It
3. What Each Artifact Actually Proves
4. From Collection to a Finding You Can Defend

Cookbooks

Ordered procedures for a first collection, remote triage, working an inherited collection, and reaching a defensible finding.

1. Collecting From a Live Machine
2. Working an Inherited Collection
3. Answering: Did This Program Run
4. Answering: Did This User Take Data

Walkthroughs

Four cases reasoned end to end, including the execution finding that was not one and the absence that was configuration.

1. The Execution That Was Not One
2. The Wiped Machine That Was Not Wiped
3. The Departure Nobody Could Prove
4. The Collection That Could Not Answer

Playground

A Windows VM you can dirty, public evidence images, and the free reference page that sits alongside this course.

Operational Reference

Every command, switch and definition from the course in one task-indexed reference, from building the toolkit and taking a collection to parsing each artifact family and merging the output into one timeline.

1. KE99.1 Operational Reference

References & Further Reading

Where to check whether anything in this course has changed, the 2026 licensing position and why three published sources disagree about it, the authoritative source for each class of fact, and how the course was research-gated against primary material rather than against the legacy skill.

Course Completion

Course Exam

KAPE and EZ Tools end-of-course exam: one triage collection from a host that no longer exists, three questions from IR, and the discipline of establishing what the evidence can support before answering any of it.

1. Course Completion. KAPE and EZ Tools

Generated 2026-09-01 from the published course. The current syllabus is always at ridgelinecyber.com/training/courses/kape-ez-tools/