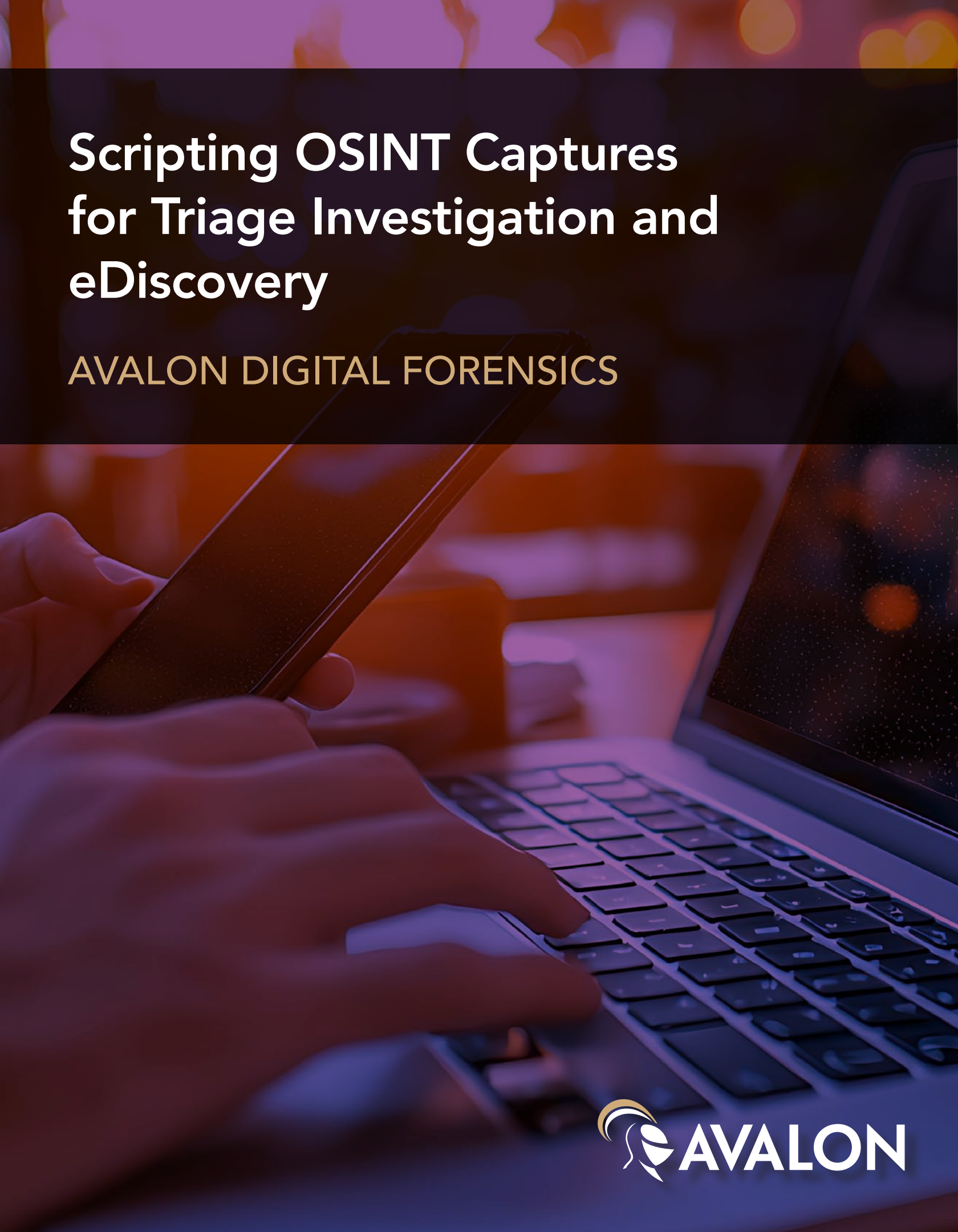
Scripting OSINT Captures for Triage Investigation and eDiscovery

AVALON DIGITAL FORENSICS



Scripting OSINT Captures for Triage Investigation and eDiscovery

A reusable workflow for extracting, classifying, validating, and organizing large social-media and advertising datasets.

Core concept: Treat preservation packages as immutable source material. Use scriptable extraction and explicit classification rules to create a separate review layer organized by dates, names, keywords, people, entities, URLs, campaigns, content types, or other matter-specific fields.

Jonathan Edwards
Senior Forensic Investigator and Digital Forensic Examiner, Team Avalon- Forensics
October 2026 | Expanded workflow edition

Anonymization notice
All examples use synthetic names, identifiers, paths, dates, URLs, and outputs. No client, account, post, campaign, or matter is identified.

Executive Summary

Large OSINT and advertising-manager exports frequently arrive as directories of ZIP packages containing PDFs, OCR derivatives, videos, analytics pages, URLs, and platform identifiers. Those packages may be preservation-complete yet inefficient for review because filenames often reflect capture mechanics instead of the issue that matters to the legal team.

A lightweight PowerShell workflow can bridge that gap. The script opens each package read-only, locates a text-bearing artifact, temporarily extracts it using a short working path, converts the PDF text layer with Poppler pdftotext, applies a documented classification rule, copies the original package into a separate review directory, and records the result in a CSV audit log. Poppler documentation describes pdftotext as a PDF-to-plain-text converter; the -layout option attempts to retain physical layout and -f/-l can restrict processing to selected pages. [3]

Dates are only one classification target. The same pattern can sort or tag collections by names, keywords, organizations, account handles, campaign identifiers, URLs, platform/content type, or review priority. The defensibility comes from keeping the source untouched, separating extraction from classification, validating rules against known examples, logging exceptions, and bounding what each derived field means.

Preservation boundary

Derived filenames and tags are review metadata. They should not replace the preserved source or be characterized as source metadata unless the underlying artifact actually contains that value.

AVALON

Scripting OSINT Captures for Triage Investigation and eDiscovery

1. Where This Fits in Discovery

EDRM 2.0 defines preservation as safeguarding relevant ESI/documents against alteration or destruction, processing as normalizing and converting ESI/documents for review and analysis, and review as evaluating them for relevance, responsiveness, privilege, confidentiality, privacy, or issue significance. It also states that the model is conceptual rather than a literal linear workflow. [1][2]

The workflow in this paper is best treated as a targeted processing and review-preparation layer. The original OSINT preservation packages remain unchanged. Derived copies, tags, and manifests exist to make the collection easier to navigate and analyze.

2. Generalized Processing Architecture

STEP	STAGE	ACTION
1	Inventory	Enumerate preserved packages and assign a processing record.
2	Identify	Locate the OCR PDF, native report, JSON, HTML, CSV, or other text-bearing artifact.
3	Extract	Write only the needed artifact to a short temporary path.
4	Normalize	Convert content to searchable text and collect relevant source metadata.
5	Classify	Apply an explicit rule for date, person, keyword, entity, campaign, URL, or other field.
6	Validate	Compare scripted output to manually established expected results.
7	Package	Copy the original preservation package into a separate review structure or generate a manifest.
8	Log	Record source, rule version, extracted value, output, status, and exception.
9	QC	Review errors, manual-review records, and representative successful results.
10	Retain	Keep the script, configuration, validation set, audit CSV, and tool versions.

3. PDF Text Extraction and Long-Path Control

A common OSINT failure mode occurs when a preservation ZIP contains a very long OCR filename. Windows Explorer may fail to extract the package because the resulting path is too long. The script does not need to reproduce that internal filename on disk. It can locate the ZIP entry and extract it directly to a deliberately short temporary name such as C:\T\o00001.pdf.

```
$OcrEntry = $Archive.Entries |
   Where-Object { [IO.Path]::GetFileName($_.FullName) -like 'OCR-*.pdf' } |
   Select-Object -First 1

$TempPdf  = 'C:\T\o00001.pdf'
$TempText = 'C:\T\o00001.txt'

[IO.Compression.ZipFileExtensions]::ExtractToFile(
   $OcrEntry, $TempPdf, $true
)

& $PdfToText -layout -enc UTF-8 $TempPdf $TempText
```

Poppler pdftotext supports page-range selection through -f and -l and UTF-8 output through -enc. Its documentation also notes that some PDFs with unusable font encodings cannot be text-extracted without OCR, which should be treated as an exception state rather than silently guessed. [3]

4. Classification is the Reusable Part

CLASSIFICATION TARGET	EXAMPLE SOURCE	DERIVED REVIEW VALUE
Dates	Explicit publication date, capture-correlated relative date	2025-07
Names/handles	Defined people, account handles, authors, speakers	PERSON_EXAMPLE
Keywords/issues	Matter terms, claims, products, allegations	ISSUE_PRODUCT

CLASSIFICATION TARGET	EXAMPLE SOURCE	DERIVED REVIEW VALUE
Organizations	Brands, agencies, counterparties, affiliates	ENTITY_EXAMPLE
Campaign identifiers	Ad IDs, campaign IDs, creative IDs	CAMPAIGN_123
URLs/domains	Channel, host, post URL, landing-page domain	DOMAIN_FACEBOOK
Content type	Post, reel, story, video, ad, analytics page	CONTENT_REEL
Review queue	Rule-based routing without asserting relevance	MANUAL_REVIEW

A hit is not a conclusion. A matched keyword does not prove relevance, and a matched name string does not prove identity. These fields are routing and review aids unless independently corroborated.

5. Three Social Media Data Patterns

Explicit publication date

Some platform analytics or manager captures expose a complete "Published on" date. This is the simplest case. Extract the date, normalize it, and preserve the original text in the audit record.

```
$Pattern = '(?i)Published\s+on:\s*([A-Za-z]{3,9})\s+(\d{1,2}),\s+(\d{4})'

# Example normalized review prefix: 2025-12
```

Relative post age

Other captures expose a relative value such as 4w or 63w rather than a complete date. The value must be correlated to the preserved capture timestamp for that package, not to the date the examiner later runs the script. The result is derived review metadata and may be approximate because relative-age displays can be rounded.

```
$CalculatedDate = $CaptureTimestamp.Date.AddDays(-(7 * $RelativeWeeks))

$MonthPrefix = $CalculatedDate.ToString('yyyy-MM')
```

Unresolved content

If the preserved OCR does not expose a defensible date, the automation should stop short of guessing. Route the package to MANUAL_REVIEW, retain any recovered URL in the audit log, and perform live-page validation separately if authorized and necessary. Information retrieved from a live page should be distinctly labeled from information present in the preserved capture.

6. Separate Extraction from Classification

The most likely explanation is a definitional mismatch between "idle" as interpreted by the OS or browser runtime and "idle" as assumed by MDM policy administrators. Web platform features exist to keep screens awake for legitimate use cases, and the OS may treat that condition as "in use," even in the absence of touch interaction.[2]

If an MDM "maximum auto-lock" policy does not override features that keep the screen awake – such as wake locks or similar mechanisms – the policy may not be enforced as intended. In those cases, the device may stay unlocked longer than administrators expect.

When a device does not reliably lock itself and require Face ID, Touch ID, or a passcode, its ability to protect sensitive data is weakened. As a result, the MDM control meant to enforce automatic locking may not be as effective as administrators assume.

```
$Record = [pscustomobject]@{
    SourcePackage = $ZipFile.Name
    ExtractedText = $Text
    SourceUrl     = $Url
    CaptureTime   = $CaptureTime
}

$Tags = @()

if ($Record.ExtractedText -match '(?i)example product') { $Tags += 'PRODUCT_EXAMPLE' }

if ($Record.ExtractedText -match '(?i)example person')  { $Tags += 'PERSON_EXAMPLE' }
```

```
$Audit = [pscustomobject]@{
    SourcePackage = $Record.SourcePackage
    Tags          = ($Tags -join ';')
    RuleVersion   = 'v1'
}
```

This architecture lets the same collection be organized several ways without repeatedly opening the source ZIPs or rewriting evidence. The output can be a filename prefix, folder assignment, CSV field, load-file field, or review manifest depending on the downstream platform.

7. Audit Log Design

FIELD	PURPOSE
SourcePackage	Original preservation package name
SourceHash	Optional package hash when included in the matter protocol
ArtifactExamined	OCR PDF, JSON, report, HTML, CSV, or other parsed artifact
CaptureTimestamp	Source timestamp used for temporal correlation
ExtractedValue	Date, name, keyword, entity, identifier, or URL recovered
Classification	Derived review value or tag
DateSource/Rule	Explicit OCR, relative OCR, keyword rule, entity rule, or manual review
RuleVersion	Version of the classifier used
OutputPath	Destination review copy or manifest entry
Status	Copied, classified, no hit, manual review, error, duplicate
Details	Exception message or QC note

8. Validation Before Scaling Out

Validation should be established before batch processing and repeated whenever classification logic changes. The goal is not merely to show that the script runs. The goal is to show that it produces the expected classification for representative records and a predictable exception state when it cannot.

VALIDATION STEP	CONTROL
Build a representative validation set	Include positive hits, negative controls, different content types, long paths, missing values, multiple timestamps, and known extraction failures
Establish expected results manually	Record the expected date, entity, keyword, URL, classification, or MANUAL_REVIEW state independently of the script
Run non-destructively	Write to a separate review location and audit log
Compare expected and observed	Treat mismatches as failures requiring investigation
Version rule changes	Change logic only after the failure mode is understood and record the new rule version
Re-run the entire validation set	Avoid fixing one sample while breaking another
Promote the validated rule	Scale only after all validation records produce the expected result or documented exception state
QC the batch	Review every error/manual-review item and sample successful results from each classification category
Retain work papers	Keep validation results, script, tool versions, audit log, and documented exceptions

Recommended validation standard

For deterministic sorting rules, every validation record should produce its expected result before the rule is promoted. An intentional exception such as MANUAL_REVIEW counts as the expected result when the source does not support a reliable classification.

9. Validation Techniques for Names and Keywords

Use exact test strings and known near-misses to measure whether the regex or search logic is too broad.

Test case variation, punctuation, whitespace, OCR substitutions, and line breaks that occur in representative captures.

For names, maintain alternate spellings or handles as explicit rule inputs rather than assuming fuzzy matches are the same person.

For keywords, distinguish literal hits from stemmed, wildcard, or proximity logic in the rule documentation.

Retain the matched source text or a bounded context snippet in the audit record when appropriate.

Use negative controls that contain similar but non-responsive strings to test false-positive behavior.

Route ambiguous matches to a review queue instead of forcing a classification.

10. Batch QC and Reproducibility

The final output should be reproducible from the same source packages, extraction utility, script version, and rule set. NIST IR 8387 addresses considerations for preserving digital evidence and identifies evidence preservation and chain of custody as central concerns. [4] The processing layer should therefore make its own changes transparent rather than blending them into source evidence.

Confirm source packages remain unchanged.

If required by the protocol, hash source and copied ZIPs to demonstrate that only the destination filename/path changed and file content remained identical.

Record Poppler/pdftotext version and command options.

Retain rule version, date run, and configuration files.

Review duplicate-destination conditions rather than overwriting.

Review every ERROR and MANUAL_REVIEW status.

Spot-check successful records across the range of output categories.

Keep a rollback-safe source and treat review outputs as reproducible derivatives.

11. Failure Modes and Controls

FAILURE MODE	RISK	CONTROL
Windows extraction path too long	Explorer cannot extract a long OCR filename	Extract the ZIP entry directly to a short temporary filename
General-purpose PDF conversion fails	Long webpage PDFs exceed another application's page-size limits	Use a PDF-native text extractor
Raw PDF byte grep returns plausible garbage	Regex matches compressed or embedded data	Run the classifier against extracted text, not the PDF container bytes
Comment date mistaken for post date	Multiple relative timestamps appear on a social page	Constrain the parser to the post region/page and validate representative captures
No usable date	Video or post capture lacks a defensible temporal value	Return MANUAL_REVIEW rather than guessing
Keyword false positive	A term occurs in unrelated text	Use validation controls, context, and explicit exception routing
Rule drift	A new platform layout changes text order	Version rules and re-run the validation set before continuing

12. Implementation Checklist

☐ Preserve and validate the source collection under the applicable matter protocol.

☐ Define the text-bearing artifacts expected inside each package.

☐ Use short temporary filenames for long-path collections.

☐ Document the extraction tool and command options.

☐ Normalize the extracted text and source metadata before classification.

☐ Define each classification rule and its evidentiary limitation.

☐ Build a representative validation set with positive and negative controls.

☐ Establish expected results manually.

☐ Require deterministic rules to pass the validation set before scaling.

☐ Version scripts and rule changes.

☐ Generate a per-package audit record.

☐ Prevent destination overwrites.

☐ Route unresolved records to MANUAL_REVIEW.

☐ QC errors and representative successes.

☐ Retain scripts, logs, validation work papers, and tool versions.

Conclusion

The value of OSINT scripting is not limited to recovering dates. The reusable capability is a controlled process for turning preservation packages into normalized, auditable review metadata. Dates, names, keywords, people, entities, URL patterns, campaign identifiers, and content types can all become sorting or tagging fields when the source supports them and the rule is documented and validated.

The defensible workflow remains constant: protect the source, extract the correct content, apply an explicit rule, validate against known examples, preserve exceptions, create only derived review outputs, and record enough information to reproduce and explain the result.

References

1. EDRM, Current EDRM Model 2.0 (2026). Source

2. EDRM, Preservation Guide. Source

3. Poppler pdftotext manual. Source

4. NIST IR 8387, Digital Evidence Preservation: Considerations for Evidence Handlers. Source

About the Author

Jonathan Edwards is a senior forensic investigator and digital forensic examiner whose work focuses on defensible collection, technical examination, timeline reconstruction, and translating complex electronic evidence into clear findings for counsel and fact finders.



For further technical details, reproduction artifacts under controlled distribution, or coordination of remediation efforts, contact:

Jonathan Edwards
Senior Forensic Investigator
jonathan.edwards@teamavalon.com