Network Working Group
Internet-Draft
Intended status: Informational
Expires: 4 October 2026

S. Grimminck, Ed.

2 April 2026

A Standard for Safe and Reversible Sharing of Malicious URLs and Indicators

draft-grimminck-safe-ioc-sharing-08

Abstract

This document codifies a consistent and reversible convention used in the threat intelligence and security communities for sharing potentially malicious indicators of compromise (IOCs), such as URLs, IP addresses, email addresses, and domain names. It describes a safe obfuscation format that reduces the risk of accidental execution or activation when IOCs are displayed or transmitted. The recommended form brackets the URI scheme name (for example, [http]) so that the string is not syntactically a valid URI per generic URI parsers; legacy scheme-substitution tokens are defined for de-obfuscation interoperability. These conventions aim to improve interoperability among tools and feeds that exchange threat intelligence data.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 4 October 2026.

Copyright Notice

Table of Contents

1. Introduction 2
2. Terminology 4
3. Problem Statement 4
4. Canonical Transformation Rule 4
4.1. Step 1: Scheme 5
4.2. Step 2: Userinfo 5
4.3. Step 3: Host 5
4.4. Step 4: Stop 6
5. Formal ABNF Grammar 6
6. De-obfuscation Techniques 6
6.1. Safety Check for Reversibility 7
7. Example Use Cases 7
8. Implementation Guidance 8
9. Edge Cases and Special Handling 8
10. Test Vectors 8
11. Security Considerations 9
11.1. Relationship to the URI Scheme Registry 9
11.2. Partial Obfuscation 10
11.3. Parser Confusion 10
11.4. De-obfuscation in Non-Executable Contexts 10
11.5. Additional Considerations 10
12. IANA Considerations 11
13. References 11
13.1. Normative References 11
13.2. Informative References 11
Author's Address 12

1. Introduction

The secure sharing of malicious artifacts is vital to threat intelligence, open-source intelligence (OSINT), and incident response. However, sharing raw URLs, IP addresses, and email addresses associated with malware or threat actors poses a risk of accidental activation.

Participants who routinely share indicators of compromise (IOCs) include security operations center (SOC) analysts, computer security incident response teams (CSIRTs), OSINT researchers, incident responders, and vendors of threat intelligence platforms and feeds. IOCs appear in email threads, instant-messaging channels, ticketing

systems, PDF and HTML reports, blog posts, paste sites, machine-readable formats such as STIX [STIX21] / TAXII [TAXII21], and platforms such as MISP [MISP]. Both human readers and automated pipelines consume this material.

When a raw URI such as "https://malicious-host.example/path" is embedded in those channels, many systems automatically detect it and render it as a clickable or otherwise actionable link. An analyst may then activate the resource unintentionally: navigating to an attacker-controlled URI can reveal the analyst's IP address and organizational affiliation, trigger delivery of malware, or alert the threat actor that a particular indicator is under active investigation. Some mail and web infrastructure pre-fetches or resolves links for scanning or preview purposes, producing the same exposure without any deliberate user action. PDF viewers and rich-text editors may turn strings that resemble URIs into hyperlinks even when the author intended plain text.

A longstanding practice in the security community is to alter IOCs so that they remain human-readable but are not treated as live URIs by typical software: for example, replacing "." with "[.]" and wrapping the scheme name in brackets so that "http" becomes "[http]". Older conventions substituted characters within the scheme name (e.g., "http" to "hxxp"), but many variant spellings emerged (e.g., "h**p", "hXXp"), hindering reliable parsing and automation. This document defines a canonical form that uses uniform square-bracket wrapping for the scheme, dot, and at-sign delimiters, and a strict order of operations so that independent implementations can interoperate. Legacy scheme substitutions ("hxxp", "hxxps") are documented for de-obfuscation interoperability but are NOT RECOMMENDED for new output.

The canonical "[scheme]" form does not occupy the URI scheme namespace (see Section 4). The legacy tokens "hxxp" and "hxxps" do occupy that syntactic position; this document requests that IANA reserve them to prevent future collision (see Section 12). The security implications are discussed in Section 11. Implementations MUST NOT treat any obfuscated form as a resolvable URI scheme.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2. Terminology

Obfuscating: The process of altering an indicator so that it cannot be accidentally activated or clicked. The goal is to prevent automatic execution or resolution, not to conceal the content from human readers; the original indicator remains visually recognizable.

De-obfuscating: The process of restoring an obfuscated indicator to its original, actionable form.

IOC: Indicator of Compromise - data such as a URL, IP address, domain name, email address, or hash associated with malicious activity.

3. Problem Statement

Inconsistent obfuscation practices hinder the reliable and automated exchange of threat intelligence. For example:

* A URL obfuscated as "h**p://example[.]example" cannot be reliably parsed by tools expecting "[http]://example[.]example".

* An IP address obfuscated with parentheses (e.g., "192.0.2(.)1") may fail to de-obfuscate in systems expecting "[.]".

Such inconsistencies reduce the effectiveness of threat detection and response.

4. Canonical Transformation Rule

To prevent nested obfuscation (e.g., "[[https]]://example[[.]]example") when a tool processes the same string twice or in the wrong order, implementations MUST apply transformations in the following strict order of operations. Implementations MUST treat already-obfuscated substrings (e.g., "[scheme]", "[.]", "[@]") as opaque and MUST NOT apply transformations to them again; thus, the transformation is idempotent. Implementations SHOULD NOT use percent-encoded characters (such as %2e for ".") in obfuscated output to prevent ambiguity. When an input indicator already contains percent-encoded delimiters, implementations MAY decode them before applying obfuscation.

4.1. Step 1: Scheme

Identify the URI scheme and wrap it in square brackets. For example, "http" becomes "[http]", "https" becomes "[https]", "ftp" becomes "[ftp]". This is the RECOMMENDED canonical form. Only the scheme name is wrapped; the scheme delimiter ("://" for hierarchical URIs, ":" for schemes such as mailto: or tel:) is preserved unchanged. Scheme names SHOULD be normalized to lowercase before wrapping, consistent with Section 3.1 of [RFC3986]. The leading "[" character is not permitted in a URI scheme name (Section 3.1 of [RFC3986]), so the result cannot be mistaken for a valid URI by compliant parsers. Because the transformation is purely syntactic wrapping, it extends to any current or future scheme without requiring a per-scheme mapping table.

The legacy tokens "hxxp" and "hxxps" are in widespread operational use as obfuscated forms of "http" and "https" respectively. Implementations encountering these tokens in existing data SHOULD recognize them as obfuscated indicators during de-obfuscation (see Section 6). These legacy forms are NOT RECOMMENDED for new output. No other legacy substitutions (e.g., "h**p", "fxxps") are defined by this specification.

4.2. Step 2: Userinfo

Identify the "@" symbol in the userinfo subcomponent (per [RFC3986]) and replace it with "[@]". This applies to email addresses and URIs containing userinfo (e.g., "username:password@host").

4.3. Step 3: Host

Replace all "." (period) characters in the Host subcomponent with "[.]". This applies to domain names and IPv4 addresses, including standalone values (e.g., "evil.example" or "198.51.100.1" without a scheme). IPv4 addresses use dotted-decimal notation that overlaps with domain name syntax, so the "[.]" substitution applies to them naturally. IPv6 addresses use colon-hexadecimal notation inside square brackets (e.g., "[2001:db8::1]"), which does not trigger auto-linking in typical software; the entire bracket-enclosed literal, including any dots within it (e.g., the dots in "::ffff:192.0.2.1"), MUST be left unchanged. Port numbers following the host (e.g., ":8080") are not part of the Host subcomponent and MUST NOT be altered.

When a scheme is present, host subcomponent boundaries are determined by the URI syntax defined in [RFC3986]. For bare indicators without a scheme (e.g., a standalone domain or IP address), host identification is implementation-dependent; implementations SHOULD apply reasonable heuristics such as recognizing dotted-decimal IPv4 addresses and domain-name patterns.

4.4. Step 4: Stop

Do not process the Path, Query, or Fragment components. Applying transformations beyond the Host in the primary URI may cause incorrect results. If a path or query string contains a nested URI (e.g., a redirect target), that nested URI should be extracted and obfuscated as a separate indicator.

5. Formal ABNF Grammar

The following uses Augmented BNF (ABNF) per [RFC5234] to define the tokens used in obfuscated IOC strings. The scheme production matches the syntax in Section 3.1 of [RFC3986]. An implementation MAY use this grammar to validate whether a string is already obfuscated or still requires processing.

```
safe-scheme    = "[" scheme "]"
scheme         = ALPHA *( ALPHA / DIGIT / "+" / "-" / "." )
legacy-scheme  = "hxxp" / "hxxps"
safe-dot       = "[" "." "]"
safe-at        = "[" "@" "]"
```

Figure 1

A compliant implementation MUST recognize strings containing safe-scheme or legacy-scheme, safe-dot, and safe-at as obfuscated when those tokens appear in the roles described in this document.

6. De-obfuscation Techniques

Tools designed to ingest obfuscated data SHOULD automatically reverse these transformations in a deterministic manner:

* Strip enclosing brackets from "[scheme]" to recover the original scheme name (e.g., "[https]" becomes "https").

* Convert legacy token "hxxps" back to "https".

* Convert legacy token "hxxp" back to "http".

* Convert "[.]" back to ".".

* Convert "[@]" back to "@".

For legacy tokens, longer strings MUST be reversed before shorter prefixes that are substrings of them (e.g., reverse "hxxps" before "hxxp"). De-obfuscation MUST maintain the original semantics of the data to avoid misinterpretation.

6.1. Safety Check for Reversibility

De-obfuscation MUST only be performed when the output is written to a non-executable buffer (e.g., a variable, string, or file) that cannot be automatically interpreted, executed, or rendered as a clickable link by the system or application. The tool MUST NOT de-obfuscate a string if it is currently being rendered in a "live" environment (e.g., a web browser preview, an active document viewer, or any context where the resulting string could be automatically executed, resolved, or displayed as a clickable link).

De-obfuscation SHOULD only occur in controlled contexts such as:

* Command-line tools with explicit user confirmation

* Isolated analysis environments (sandboxes)

* Backend processing pipelines that do not render output to users

Accidental activation during the de-obfuscation process poses a security risk and MUST be prevented.

7. Example Use Cases

Common scenarios include:

* *OSINT Sharing:* A report lists obfuscated URLs (e.g., "[http]://malware[.]example/payload") to prevent accidental clicks.

* *Email Communication:* Security teams share obfuscated IOCs like "attacker[@]example[.]example" or "[mailto]:attacker[@]example[.]com" in email threads.

* *Threat Intelligence Platforms:* Automated ingestion of obfuscated IPs (e.g., "192[.]0[.]2[.]1") for blocklist updates.

8. Implementation Guidance

Software designed to parse threat intelligence feeds should explicitly support these obfuscation and de-obfuscation conventions. Implementations SHOULD verify correct behavior through unit tests and validation scripts using the test vectors in Section 10.

9. Edge Cases and Special Handling

Internationalized Domain Names (IDNs): Obfuscate punycode domains similarly (e.g., "xn--n3h[.]example").

Non-Standard URI Schemes: The "[scheme]" canonical form extends to any URI scheme. For example, "smb://fileserver.example/share" becomes "[smb]://fileserver[.]example/share".

IPv6 Literals in URIs: Do not alter colon characters (":") or brackets ("[", "]") in IPv6 addresses. For example, "[2001:db8::1]" MUST remain unchanged. Only scheme names or domain elements surrounding them should be obfuscated.

10. Test Vectors

The following provides a "golden set" of inputs and expected outputs. Domain names use the "example" reserved space per [RFC2606]; IPv4 addresses use documentation ranges per [RFC5737]; IPv6 addresses use the documentation prefix per [RFC3849]. Implementations SHOULD use these vectors to ensure correct behavior and to avoid under-obfuscation (e.g., missing email addresses) or over-obfuscation (e.g., obfuscating IPv6 colons).

* Standard URL: https://bad.example -> [https]://bad[.]example
* URL with path: https://evil.example/path -> [https]://evil[.]example/path
* Deep-link URL: https://bad.example/path/to/page?q=1#frag -> [https]://bad[.]example/path/to/page?q=1#frag
* HTTP URL: http://attacker.example -> [http]://attacker[.]example
* FTP URL: ftp://files.example/ -> [ftp]://files[.]example/
* Mailto: mailto:user@example.com -> [mailto]:user[@]example[.]com
* IPv4 address: 198.51.100.1 -> 198[.]51[.]100[.]1
* IPv4 in URL: http://192.0.2.1 -> [http]://192[.]0[.]2[.]1

* IPv6 in URL: http://[2001:db8::1]:8080 -> [http]://[2001:db8::1]:8080

* IPv4-mapped IPv6: http://[::ffff:192.0.2.1] -> [http]://[::ffff:192.0.2.1]

* Email address: phish@target.example -> phish[@]target[.]example

* Punycode domain: xn--n3h.example -> xn--n3h[.]example

* URL with userinfo: http://user:pass@attacker.example -> [http]://user:pass[@]attacker[.]example

* Bare domain with port: evil.example:443 -> evil[.]example:443

* Idempotency check: [https]://bad[.]example -> [https]://bad[.]example

* Legacy de-obfuscation: hxxps://bad[.]example de-obfuscates to https://bad.example

Note: The IPv6 rows demonstrate that colons and brackets within the IPv6 literal MUST NOT be altered, including dots in IPv4-mapped IPv6 (::ffff:192.0.2.1). The deep-link row shows that Path, Query, and Fragment (per Step 4) are not processed. The Punycode row shows that IDN labels in punycode form receive the same "[.]" treatment as regular domain labels. The bare-domain-with-port row demonstrates that port numbers are unaffected by the transformation. The idempotency row confirms that applying the transformation to an already-obfuscated string produces no change. The legacy de-obfuscation row confirms backward compatibility with the older "hxxp" convention.

11. Security Considerations

While these obfuscation techniques reduce the risk of accidental activation, systems processing obfuscated indicators SHOULD apply the safeguards described in this section.

11.1. Relationship to the URI Scheme Registry

The canonical "[scheme]" form does not occupy the URI scheme namespace (see Section 4). Software that encounters bracketed schemes MUST NOT attempt to resolve or dereference them.

The legacy tokens "hxxp" and "hxxps" do occupy the syntactic position of a URI scheme. This document requests IANA reservation of those strings (see Section 12) to prevent a future scheme registration from

colliding with their established operational use. Until that reservation is in place, implementers should design parsers to consider surrounding context when disambiguating these tokens.

11.2. Partial Obfuscation

When a scheme is present, a compliant tool MUST obfuscate both the scheme (Step 1) and the host-level delimiters (Steps 2-3). Partial obfuscation (for example, replacing only "." with "[.]" while leaving "https" unchanged) creates a false sense of security: the scheme remains active and could still trigger automatic linkification or execution. Implementations MUST NOT produce partially obfuscated output when full obfuscation is intended. For bare indicators without a scheme (e.g., a standalone IP address or email address), host-level and userinfo obfuscation (Steps 2-3) alone constitutes valid Safe-IOC output.

11.3. Parser Confusion

Implementations that parse Safe-IOC strings may become confused by malformed or inconsistently obfuscated input. For example, "[https]://example.example" (scheme obfuscated but dots not) or "https://example[.]example" (dots obfuscated but scheme not) are not valid Safe-IOC formats. Parsers SHOULD validate that obfuscated strings conform to the canonical transformation rule and the ABNF before de-obfuscation. Rejecting or flagging ambiguous input reduces the risk of misinterpretation.

11.4. De-obfuscation in Non-Executable Contexts

As stated in Section 6, de-obfuscation MUST only occur when the result is placed in a non-executable buffer. A non-executable buffer is one that cannot be automatically interpreted by the system (e.g., as a URI to fetch, a command to run, or a link to display). Writing de-obfuscated output into a live document, rich-text editor, or browser address bar before explicit user action creates an unacceptable risk of accidental activation.

11.5. Additional Considerations

* Implementations that do not follow the canonical transformation rule (e.g., by not treating "[.]" and "[@]" as opaque) MAY produce nested or non-reversible output when obfuscation is applied repeatedly. Compliant implementations avoid this by design.

* Obfuscated URLs in PDFs may still be rendered as hyperlinks; use plain-text formatting.

* Systems processing obfuscated indicators MUST treat them as untrusted data, applying sandboxing or isolated environments for analysis.

* Credentials (e.g., _username:password_) SHOULD NOT be shared, even in obfuscated form, due to inherent security risks.

* Analysts accessing de-obfuscated indicators for investigation SHOULD do so within protected environments such as containers or dedicated analysis VMs to limit exposure to exploits.

12. IANA Considerations

IANA is requested to reserve the following two strings in the URI Schemes registry so that they are not assigned as scheme names in the future:

* hxxp

* hxxps

These strings are already in widespread operational use within the threat intelligence community as obfuscated forms of "http" and "https" respectively. They are NOT RECOMMENDED for new use (see Section 4). The reservation prevents a future scheme registration from colliding with these established tokens.

13. References

13.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997, <https://www.rfc-editor.org/rfc/rfc2119>.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005, <https://www.rfc-editor.org/rfc/rfc3986>.

[RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008, <https://www.rfc-editor.org/rfc/rfc5234>.

13.2. Informative References

[RFC2606] Eastlake, D. and A. Panitz, "Reserved Top Level DNS Names", BCP 32, RFC 2606, June 1999, <https://www.rfc-editor.org/rfc/rfc2606>.

[RFC3849] Huston, G., Lord, A., and P. Smith, "IPv6 Address Prefix Reserved for Documentation", RFC 3849, July 2004, <https://www.rfc-editor.org/rfc/rfc3849>.

[RFC5737] Arkko, J., Cotton, M., and L. Vegoda, "IPv4 Address Blocks Reserved for Documentation", RFC 5737, January 2010, <https://www.rfc-editor.org/rfc/rfc5737>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, May 2017, <https://www.rfc-editor.org/rfc/rfc8174>.

[STIX21] Jordan, B., Ed., Piazza, R., Ed., and T. Darley, Ed., "STIX Version 2.1", OASIS Standard stix-v2.1-os, 10 June 2021, <https://docs.oasis-open.org/cti/stix/v2.1/os/stix-v2.1-os.html>.

[TAXII21] Jordan, B., Ed. and D. Varner, Ed., "TAXII Version 2.1", OASIS Standard taxii-v2.1-os, 10 June 2021, <https://docs.oasis-open.org/cti/taxii/v2.1/os/taxii-v2.1-os.html>.

[MISP] MISP Project, "MISP - Malware Information Sharing Platform and Threat Sharing", <https://www.misp-project.org/>.

Author’s Address

Stefan Grimminck (editor)
Email: ietf@stefangrimminck.nl