©2026 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works. Accepted for publication in the CyberHunt 2026 workshop, 2026 IEEE International Conference on Big Data (BigData), Phoenix, AZ, USA. DOI to be added when published.

STIX and STONES: Formal Semantics and Cross-Framework Links for STIX 2.1 Communities

Ryan Hohimer
Hohimer Intelligence Strategies LLC
West Richland, WA, USA
ryan.hohimer@cyberterrain.org

Jans Aasman
Franz Inc.
Lafayette, CA, USA
ja@franz.com

Keven Ates
Independent Researcher
VA, USA
atescomp@gmail.com

Jeff Gentes
Dataminr Inc.
Charlotte, NC, USA
jeff.gentes@dataminr.com

***Abstract*—STIX 2.1 has become the common language for exchanging cyber threat intelligence, but its meaning lives in prose and JSON schemas. It can cite reference knowledge through external references, but it cannot say how the things it describes relate to it: which product versions a CVE affects, which weakness it exploits, which ATT&CK techniques a control mitigates. As AI-enabled defense puts language models in front of threat intelligence, that gap matters more, because a model reasoning over untyped, unlinked records has to guess the links itself. We present two open OWL 2 ontologies that close the gap while keeping STIX interchange intact. STONES is a faithful binding of STIX 2.1: every domain object, observable and relationship becomes a named class or property. STONEWORK is an independent cyber ontology whose STIX adapter aligns STONES to it through 52 class, 17 property and 62 vocabulary alignments, and which links CVE, CPE, CWE, CAPEC, MITRE ATT&CK and NIST SP 800-53 with typed, cross-framework predicates. Converting STIX bundles to STONES and back reproduced all 33,940 objects exactly. We report on using the ontologies to build a 55-million-triple knowledge graph updated nightly, and on an illustrative case study of a recent CISA advisory: its STIX 2.1 file was valid, yet connected to nothing until links were proposed and confirmed by an analyst, each confirmation recorded as a STONEWORK opinion on a qualified assertion. Grounding a language-model agent in those links reduced, but did not end, its guessed control-to-technique mappings, and made every remaining guess checkable against the graph. We close with recommendations for the communities that produce and consume STIX.**

***Index Terms*—cyber threat intelligence, STIX, ontologies, knowledge graphs, OWL, neuro-symbolic AI**

I. Introduction

Structured Threat Information Expression (STIX) version 2.1 [1] is how governments, ISACs and vendors package cyber threat intelligence (CTI) for exchange. It is detailed and widely implemented. It is also, by design, an exchange format: a JSON serialization whose semantics are defined in specification prose. Two consequences follow.

First, STIX cannot be reasoned over directly. A consumer cannot ask a STIX processor whether an intrusion set is a kind of activity cluster, or infer that a `uses` relationship from a group to a technique implies anything about either. Each tool re-implements the prose.

Second, and more important in practice, STIX has nowhere to put the reference knowledge that makes intelligence actionable. An `external_references` entry can cite a CVE or a technique, and a custom extension can carry more, but neither says how the cited things relate to anything else. An advisory may name CVE-2019-11510 and ATT&CK technique T1560, but the fact that CVE-2019-11510 affects Ivanti Connect Secure 9.0R2.1, exploits a path-traversal weakness, and is mitigated by flaw remediation lives in NVD, CWE, CAPEC, ATT&CK and NIST SP 800-53, each with its own format and identifiers. Joining them is left to every analyst, every time.

This gap widens as CTI work becomes AI-enabled. Retrieval-augmented and graph-augmented language models [2], [3] answer best when the facts they need are typed and explicitly linked. Given unlinked records, a model fills the gaps from its training data, plausibly and sometimes wrongly. Section VI shows a concrete instance.

We argue that STIX communities need a formal semantic layer *beside* the exchange format, not instead of it, and we contribute:

- **STONES** (STIX ONtology Expression System), a faithful OWL 2 binding of STIX 2.1, with a measured JSON round trip (Section III);
- **STONEWORK** (Semantic Threat Ontology for Next-generation Extensions, Workflows, Objects, Relationships, and Knowledge), an independent cyber ontology with typed cross-framework links and a model of plans, their runs and the hypotheses they test, composed with STONES through an explicit adapter rather than an extension (Section IV);
- a **reconciliation pattern** that connects STIX content to reference knowledge through qualified assertions and analyst opinions, so that machine-proposed links are never silently merged (Section V);
- an **experience report**: a 55-million-triple knowledge graph built on both ontologies, an illustrative case study of a 2026 CISA advisory, and observations from grounding a language-model agent in the graph (Section VI); and
- **recommendations** for STIX producers, consumers and standards work (Section VII).

Both ontologies are open source under the MIT license. Fig. 1 shows how the pieces fit together.

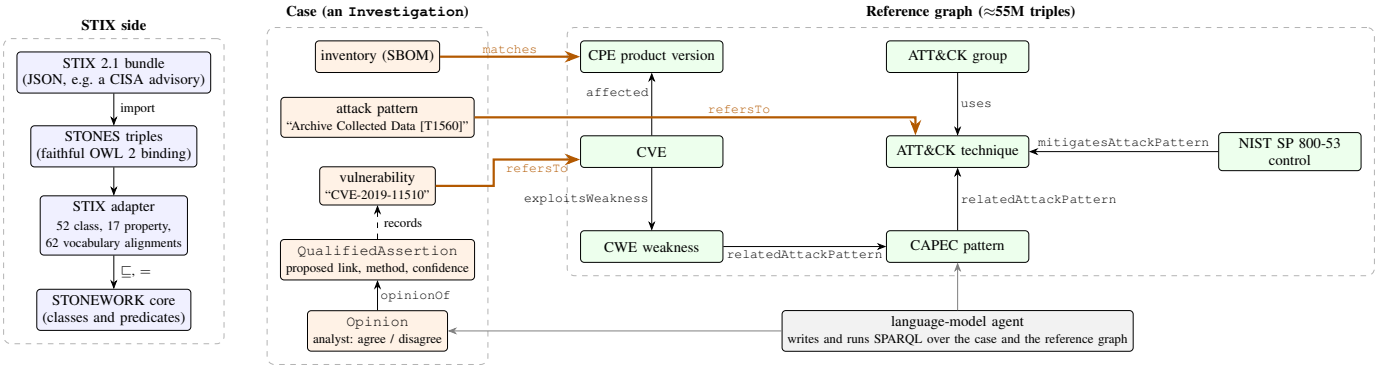


Fig. 1. STIX content enters through STONES and the STIX adapter. Proposed `refersTo` links from case objects to reference entities are recorded as qualified assertions, and take effect when an analyst's opinion confirms them. Typed cross-framework predicates then connect the case to products, weaknesses, patterns, techniques, groups and controls, for both computed queries and a language-model agent.

II. Background and Related Work

CTI standards. Mavroeidis and Bromander [4] surveyed CTI taxonomies, sharing standards and ontologies, and found that sharing standards such as STIX lack the formal semantics needed for reasoning, while existing ontologies lack coverage of the standards practitioners use. CACAO [5] brings the same exchange-first approach to security playbooks.

Cybersecurity ontologies and knowledge graphs. UCO [6] integrated several earlier vocabularies, including STIX 1.x concepts, into one OWL ontology. Iannacone et al. [7] built an ontology for cyber security knowledge graphs from NVD and related sources. Asgarli and Burger [8] examined semantic renderings of threat-sharing standards. MITRE's D3FEND [9] models defensive countermeasures as an OWL knowledge graph linked to ATT&CK. The SEPSES knowledge graph [10] integrates CVE, CPE, CWE, CAPEC and CVSS data as linked data and keeps it current as the sources change; ATT&CK-KG [11] builds on it to link observed attacks to ATT&CK techniques. Akbar et al. [12] designed a unified cyber security ontology over several reference sources. gistCyber [13], earlier work by two of us and separate from STONES and STONEWORK, builds on gistBFO, which combines the gist enterprise upper ontology with BFO, to harmonize ATT&CK, CAPEC, CVE, CWE, CPE and NIST SP 800-53 under shared upper-level classes, relating sources through shared technique identifiers. Ontology-based inference over STIX has been used to characterize threat actors [14]. In earlier practitioner writing, two of us [15] argued that STIX 2.1 data should be translated from JSON into OWL knowledge graphs so that the relationships between threat entities can be explored and queried; this paper supplies the ontologies and the evidence that argument lacked.

The gap. Table I compares the closest work. None provides both (a) a complete, faithful binding of STIX 2.1 that round-trips with the JSON form and (b) a maintained, populated graph that links STIX content to CVE, CPE, CWE, CAPEC, ATT&CK [16] and NIST SP 800-53 [17] with typed predicates. STONES and STONEWORK target exactly that pairing, keeping the two concerns in separate ontologies.

TABLE I
Closest related work. *STIX*: coverage of STIX as OWL. *Linked*: reference sources joined by typed links. *Prov.*: links carry provenance and analyst judgment. *Graph*: a populated graph is released and kept current.

Work	STIX	Linked	Prov.	Graph
UCO [6]	1.x concepts	CVE, CAPEC, CVSS	–	–
D3FEND [9]	–	ATT&CK, CWE, NIST	–	✓
SEPSES [10]	–	CVE, CPE, CWE, CAPEC, CVSS	–	✓
ATT&CK-KG [11]	–	ATT&CK, CAPEC, CWE, CVE	–	–
gistCyber [13]	–	ATT&CK, CAPEC, CVE, CWE, CPE, NIST	–	–
This work	2.1, full; round trip	CVE, CPE, CWE, CAPEC, CVSS, KEV, ATT&CK, NIST	✓	✓

What is new. Linking the reference frameworks is not new: SEPSES, D3FEND and gistCyber do it, and we reuse the sources' own links and the control-to-technique mapping of the Center for Threat-Informed Defense (CTID) [18] rather than inventing our own. What this paper adds is (i) a faithful, round-trip-tested OWL binding of STIX 2.1; (ii) composing it with a cyber ontology through an adapter of alignment axioms, so that STIX content joins the linked frameworks without being rewritten; and (iii) a reconciliation pattern in which links from STIX content to reference knowledge are qualified assertions that analysts judge. OWL 2, SKOS and SHACL are used as published.

III. STONES: A Faithful OWL Binding of STIX 2.1

STONES (version 0.3.1) renders STIX 2.1 in OWL 2 [19]. It was built directly from the text of the STIX 2.1 specification, object by object. Its design rule is *faithfulness*: it says what the specification says and no more, so that a STIX object can be turned into triples and back without loss, and so that STONES can track the specification as it evolves. It has 96 classes, 91 object properties and 198 datatype properties, organized in four modules (classes, object properties, datatype properties, and vocabularies).

Objects. Every STIX Domain Object, Cyber-observable Object, Relationship Object and meta object has a named class (`stones:IntrusionSet`, `stones:FileObject`, `stones:Relationship`, ...). Common properties such as `created`, `modified` and `external_references` are defined once, on `stones:StixObject`.

Relationships, twice. A STIX Relationship carries its own identifier, timestamps, confidence and markings, so STONES keeps it as an individual of `stones:Relationship` with `sourceRef`, `targetRef` and the `relationshipType` string. It also asserts a direct shortcut property from source to target (`stones:uses`, `stones:mitigates`, ...) so that queries need not traverse the reified node. Listing 1 shows both.

```
ex:rel-1 a stones:Relationship ;
    stones:relationshipType "uses" ;
    stones:sourceRef ex:intrusion-set-1 ;
    stones:targetRef ex:attack-pattern-7 ;
    stones:confidence 80 .
ex:intrusion-set-1 stones:uses ex:attack-pattern-7 .
```

Listing 1. A STIX `uses` relationship, reified and projected.

Open vocabularies. STIX open-vocabulary values (malware types, threat-actor roles, infrastructure types, ...) are named individuals in a separate `stonex:` namespace, so that a value such as `ransomware` is a resource that can be described and aligned, not just a string.

Transport. A STIX Bundle is packaging, not an assertion about the cyber domain; STONES keeps it, but nothing downstream treats it as a domain entity.

Round trip. We tested faithfulness by converting STIX bundles to STONES triples and back, then comparing every property of every object with the original (Table II). The forward direction is a declarative SPARQL CONSTRUCT mapping over the JSON; the reverse reads the triples back through the STIX key each STONES property documents. Lists are the one thing triples cannot carry on their own: a property with several values becomes several triples, which keep neither the members' order nor repeated members. STONES therefore has a few serialization-fidelity terms. Where a list's order differs from the sorted order a reader would otherwise produce, `stones:ListOrder` records each member's position on an index node (not an `rdf:List`, whose vocabulary OWL 2 DL reserves). Properties the mapping does not model, such as vendor `x_` extensions, are kept verbatim as one JSON string in `stones:unmodeledProperties`. Both are written only where the modeled triples fall short, found by reading each object back and comparing. Neither is aligned to STONEWORK, because they describe the document, not the domain.

With the fidelity terms, every bundle comes back exactly: all 536,246 properties of 33,940 objects. Each regenerated bundle also gets the same verdict from the OASIS STIX validator as its source. The two Unit 42 bundles are invalid as published (an `identity_class` given as a list, for example) and stay so. A hand-built bundle of edge cases also comes back exactly: granular markings, observable extensions, hashes on external references, and values with quotes, backslashes and spaces.

TABLE II
STIX JSON → STONES → JSON (ATT&CK RELEASE 20.1; UNIT 42 OUTLOOK: CVE-2023-23397). *Props.*: TOP-LEVEL PROPERTIES. *Std.*: STANDARD STIX PROPERTY VALUES CARRIED BY THE MODELED TRIPLES, LIST ORDER ASIDE. *Cust.*: THE SAME FOR `x_` PROPERTIES. *Exact*: OBJECTS IDENTICAL AFTER THE ROUND TRIP, WITH THE FIDELITY TERMS.

Bundle	Objects	Props.	Std. %	Cust. %	Exact %
ATT&CK Enterprise [20]	26,699	432,435	99.99	67.9	100
ATT&CK Mobile	2,706	44,303	99.99	68.2	100
ATT&CK ICS	2,570	40,147	99.99	67.7	100
CISA AA26-281A [21]	1,575	15,696	99.99	—	100
Unit 42 Qakbot [22]	327	3,062	99.87	2.6	100
Unit 42 Outlook	63	603	99.79	13.4	100

The modeled triples alone carry 99.99% of the standard STIX property values; 27 of 413,233 miss. The misses are repeated list members (17 ATT&CK objects cite one reference twice and one repeats a label), the `tactic_refs` of ATT&CK's matrices (an ordered list on a custom object), one marking definition's `extensions`, and the malformed Unit 42 values. List order is lost in 4,236 Enterprise lists, most of them `external_references`, whose first entry ATT&CK uses for its own identifier. Custom properties are where the modeled triples fall short. About two thirds of ATT&CK's `x_mitre_` values are modeled; the rest are bookkeeping such as `x_mitre_modified_by_ref`, present on every object, which are kept verbatim. Of Unit 42's custom properties, almost none are modeled. On Enterprise, the fidelity terms add 13% to the 655,095 modeled triples.

IV. STONEWORK: CROSS-FRAMEWORK SEMANTICS

STONEWORK (version 0.20.1) is a cyber ontology for reasoning across threats, weaknesses, controls, assets, events, evidence and the processes that connect them. Its core has 150 classes, 286 object properties and 159 datatype properties; a separate categories module adds 39 category classes and 250 controlled-vocabulary concepts.

A. Peers, not an extension

A naive design would make STONEWORK an extension of STONES. We chose an alignment approach. STIX is one source among many: CVE records, CPE dictionaries, CWE, CAPEC, ATT&CK, D3FEND, EMB3D, OCSF and UCO each have their own shape. Making any one of them the base of the model would force the others into its categories. Instead, STONEWORK has a lightweight core with its own stable identifiers, and each source framework, STIX included, is a *peer* module that imports the core and aligns to it.

The STIX adapter consists only of alignment axioms:

- 52 class alignments, e.g. `stones:IntrusionSet` $\sqsubseteq$ `stonework:CyberActivityCluster`;
- 17 property alignments, e.g. `stones:uses` $\sqsubseteq$ `stonework:uses`, `stones:sourceRef` $\sqsubseteq$ `stonework:assertionSubject`;

- 62 vocabulary alignments (`owl:sameAs`), e.g. `stonex:_ransomware` = `stonework:Ransomware`.

With the adapter loaded, STIX data answers STONEWORK queries without being copied or rewritten, and the STIX triples remain available for faithful export.

The vocabulary alignments purposely use `owl:sameAs` rather than `skos:exactMatch`. Both sides are individuals naming one category, and STIX data should be categorized by STONEWORK's concept with no extra rule: a malware object typed with `stonex:_ransomware` is then categorized by `stonework:Ransomware` for any OWL reasoner, whereas `exactMatch` carries no entailment and would leave the step to every query. The cost of `owl:sameAs` is that every statement about one individual holds of the other. It is acceptable here because a `stonex:` individual carries only its label and the specification's definition, and only the 62 of 354 values whose meaning a STONEWORK concept already has are aligned; the rest stay unaligned rather than approximated.

What OWL adds beyond SPARQL. The axioms do work a query would otherwise repeat: subproperty entailment lets one query over `stonework:exploitsWeakness` cover every framework's own property; the class hierarchy places `stones:IntrusionSet` under `stonework:CyberActivityCluster`; property chains turn nested source structures into direct links; and domain, range and disjointness axioms make inconsistent data detectable (Section VI). In the deployed graph these entailments are materialized by SPARQL updates written from the axioms rather than computed at query time; the ontology states which updates are correct, and the result is checked for logical inconsistency.

B. Qualified assertions

A STIX Relationship is an instance of a more general need: an assertion that carries provenance, time and confidence. STONEWORK models this as `stonework:QualifiedAssertion`, with functional `assertionSubject`, `assertionObject` and `assertionRelationType`. STONES relationships specialize it through the adapter. Direct properties serve as query-friendly projections when context loss is acceptable.

C. The cross-framework predicates

Table III lists the predicates that connect one reference framework to another; it is the complete set. STONEWORK's other properties (`uses`, `detects`, `attributedTo`, ...) relate entities within a framework. Each framework module declares its own concrete property as a subproperty of the STONEWORK one (e.g. `cve:exploitsWeakness` $\sqsubseteq$ `stonework:exploitsWeakness`), so queries can be written once against STONEWORK. Where a source nests a relationship, a property chain projects it: a CVE record's `affected` entry is a component context, and when it names a canonical product version a chain yields the direct `stonework:affected` link.

TABLE III
STONEWORK CROSS-FRAMEWORK PREDICATES AND WHERE EACH LINK COMES FROM

Predicate	From → To	Source of the links
`affected`	CVE → CPE product (version)	NVD CPE configurations
`exploitsWeakness`	CVE → CWE	CVE records
`relatedAttackPattern`	CWE → CAPEC → ATT&CK	CWE and CAPEC records
`mitigatesAttackPattern`	NIST control → ATT&CK technique	CTID mapping [18]
`refersTo`	source object → canonical entity	reconciliation (Section V)

What the links mean. Every link but `refersTo` restates something a source asserts; STONEWORK types it and joins it, but does not strengthen it. `affected` links are taken from NVD's CPE configurations: each CPE match NVD marks vulnerable becomes a link. The configurations' AND/OR structure and their non-vulnerable platform conditions are not kept, and a match given as a version range links to the product as a whole, not to each version in the range. An `affected` link therefore means that NVD lists the product, or the exact version, as vulnerable in some configuration, not that a given deployment is vulnerable; exact-version links are the strong case. `mitigatesAttackPattern` means that CTID judged the control relevant to the technique, not that it is effective against it.

D. Adding frameworks and absorbing new versions

A framework is added as a module in three steps: it imports the STONEWORK core; it declares its classes and properties as subclasses and subproperties of STONEWORK terms, with property chains where the source nests a relationship; and a declarative mapping, like the STIX one in Section III, turns the source's records into instances. Modules are versioned separately (the STIX adapter is at 0.4.1), so a change in one source's representation, such as a new CVE record format or CWE schema, touches that module and its mapping, not the core or the other modules, and queries written against STONEWORK terms keep working. New releases of a source's content need no ontology change at all. A new STIX version would be a new STONES release, built from the specification as before, with the adapter updated to match.

E. Modeling conventions

Classes or concepts. An OWL class is used for an intrinsic kind of thing whose instances take part in class reasoning (`Malware`); a SKOS [23] concept is used for a controlled value that classifies something (`Ransomware`, a `MalwareType`). A ransomware sample is *categorized by* the concept; the concept is not a subclass of malware. This keeps vocabularies extensible without destabilizing the class hierarchy.

Validation is a profile, not the contract. SHACL [24] shapes ship as optional data-quality profiles, tiered by severity, and never closed. The interoperability contract is the open-world

ontology; a record can be a legitimate STONEWORK instance while failing a shape that a particular consumer cares about.

Investigations. STONEWORK has the concepts an analyst's work needs: an `Investigation` with status, priority and lead investigator; `Hypothesis` (`testsHypothesis`); and `Opinion`, which may judge a `QualifiedAssertion` (`opinionOf`). Section V uses all three.

F. Plans, runs and hunts

STONEWORK models processes as plans and their runs. A `Plan` specifies intended behavior: `Step`s with input and output `Variable`s, connected by `Transition`s whose guards express branches, loops and parallel paths. Its subclasses differ in whose plan it is and what running it does, not in structure: a `BusinessProcess`, a `ControlPlan`, a defender's `Playbook`, an adversary's `OperationPlan`, and an `InvestigationPlan` or `DetectionPlan`, whose runs produce a judgment rather than a change in the world. A run is an activity that is `executionOf` the plan; each step's execution `correspondsToStep` its step and is `partOfExecution` the run; and the values the variables take in that run are `VariableBinding`s, so a plan run a hundred times keeps a hundred sets of values. A run that departs from its plan (a skipped step, the wrong role, steps out of order) is recorded as a `Deviation`, itself a qualified assertion. The plan side aligns with PROV [25] and P-Plan [26].

A hunt is this pattern with a hypothesis in front of it. A `Hypothesis` predicts what a variable will resolve to; the run binds the variable; and `hasHypothesisStatus` records whether the prediction held. Rival explanations are linked by `competesWith`, and a hypothesis that is set aside keeps its lineage. Listing 2 is a hunt for credential dumping from LSASS memory. Because its detection rule is linked to the ATT&CK technique, the hunt joins the rest of the graph: the groups known to use the technique, the controls mapped to it, and the evidence the run cited.

```
ex:hunt-lsass a stonework:DetectionPlan ;
    stonework:hasStep ex:s1 ;
    stonework:hasOutputVariable ex:dumped-host .
ex:s1 a stonework:Step ;
    stonework:usesDetectionRule ex:rule-lsass-read .
ex:rule-lsass-read a stonework:DetectionRule ;
    stonework:detects attack:T1003.001 .
ex:h1 a stonework:Hypothesis ;
    stonework:predictsVariable ex:dumped-host ;
    stonework:hasHypothesisStatus
        stonework:hypothesisStatusConfirmed .
ex:run-1 a stonework:Investigation ;
    stonework:executionOf ex:hunt-lsass ;
    stonework:testsHypothesis ex:h1 ;
    stonework:evidencedBy ex:edr-log-0814 .
ex:b1 a stonework:VariableBinding ;
    stonework:bindingOf ex:dumped-host ;
    stonework:boundIn ex:run-1 ;
    stonework:boundTo ex:host-7 .
```

Listing 2. A hunt: a detection plan, its hypothesis, and one run.

CACAO [5] is to playbooks what STIX is to intelligence: an exchange format, carried out by orchestration tools. STONEWORK's plans are meant as its semantic companion, as STONES is for STIX. Most CACAO 2.0 constructs have a direct STONEWORK form (playbook types, action and playbook-action steps, `on_success` and `on_failure`, if, switch and while conditions, parallel branches), a few such as timeouts and exception handlers do not yet, and a plan can keep its CACAO document alongside. What STONEWORK adds is what happens after a playbook is shared. CACAO leaves a step's outcome to the implementation and has no object for a run; STONEWORK records each run, its bindings, its deviations and the hypotheses it tested, linked to the techniques, controls and evidence in the same graph.

V. Reconciling STIX Content with Reference Knowledge

A faithful binding exposes a problem that JSON hides: *valid STIX can connect to nothing.* Producers often identify techniques and vulnerabilities only in an object's `name`, e.g. "Collection / Archive Collected Data [T1560] (ENT)", with no `external_references` entry. To a machine, that attack pattern and ATT&CK's T1560 are unrelated individuals.

STONEWORK's answer is `stonework:refersTo`, which relates an object from an intelligence source to the canonical entity it refers to when the two are distinct individuals. Our reconciler proposes such links, each with a confidence that is highest for a structured external reference and lowest for a match on the name alone.

Crucially, a proposed link is not a fact. Each is written as a `QualifiedAssertion` with a stable identifier, the method that found it and its confidence. Only structured references are projected to a direct `refersTo` triple automatically. Name-based links stay proposals until an analyst records an `Opinion` of the assertion, agreeing or disagreeing, with author, time and explanation (Listing 3). Agreement projects the link; disagreement records why the two are not the same; either can be withdrawn. The reasoning behind every link that later analysis depends on is therefore in the graph, in the same vocabulary as the data.

```
ex:link-3f2a a stonework:QualifiedAssertion ;
    stonework:assertionSubject ex:attack-pattern-7 ;
    stonework:assertionObject  attack:T1560 ;
    stonework:assertionRelationType "refers-to" .
ex:op-91 a stonework:Opinion ;
    stonework:opinionOf ex:link-3f2a ;
    stones:opinion "agree" ; dcterms:creator "analyst-1" .
ex:attack-pattern-7 stonework:refersTo attack:T1560 .
```

Listing 3. A proposed link and the analyst's confirmation.

Lifecycle. A link is in one of three states. *Proposed*: the assertion exists, with its method and confidence, but no `refersTo` triple. *Confirmed*: an agreeing opinion, as in Listing 3, projects `refersTo`. *Rejected*: a disagreeing opinion records why; for a link from a structured reference it also removes the automatic projection, and the removal survives every later re-import, because analysts' opinions are stored apart from what imports write and are applied again after each. Withdrawing an opinion returns the link to its automatic state. The assertion itself is never deleted, so a retraction leaves the record of what was proposed, how and why.

Before any link is written, the identifier it rests on, structured or found in a name, must exist in the reference graph as

an entity of a kind compatible with the object's STIX type: an attack pattern may refer to a technique, but not to a mitigation that reuses an old technique number. Identifiers that match nothing are reported, not linked.

Each link has one current opinion: a later opinion replaces the earlier one and records its own author and time. STONEWORK allows several opinions of one assertion, so keeping analysts' disagreements side by side, with a policy for resolving them, is a matter of workflow rather than modeling; it is future work.

VI. Experience Report

This section reports one operator's experience and one illustrative case, with a fictitious inventory. It is not an evaluation; its purpose is to show the mechanisms of Sections III–V at work on real data and to report what happened when they did.

A. A shared community knowledge graph

Linked data has long envisaged common reference graphs for a domain: one curated, openly linked body of the facts an industry shares, which each organization joins to its own data instead of rebuilding it. We built such a resource for CTI: a reference knowledge graph that holds the public sources a STIX community already relies on, linked under one ontology and kept current, so that every case, tool and agent stands on the same facts. It is built on both ontologies in AllegroGraph 9.1, with each source in its own named graph. Table IV gives its size on 18 October 2026. A pipeline updates it nightly: it pulls new and changed CVE records, fetches CPE and CPE-match deltas from NVD, re-materializes the cross-framework links with SPARQL 1.1 [27] updates, and runs a health check against per-graph minimum counts. ATT&CK, CWE and CAPEC are reloaded when MITRE publishes a release; when MITRE revokes an ATT&CK technique, its control mappings follow MITRE's revoked-by relationship to the successor, so the links survive until the mapping catches up. NIST control-to-technique links come from CTID's mapping [18], since the NIST catalog defines none of its own. Each CVE's CVSS metrics are typed nodes taken from its record. Since the snapshot in Table IV, CISA's Known Exploited Vulnerabilities catalog [28] has been added: each entry is a typed observation of its CVE, with the required action and CTID's mapping of the catalog to ATT&CK [18]. Textual descriptions are copied to a separate vector store (about 376,000 embeddings) for semantic search by language-model agents.

ATT&CK Enterprise is loaded from MITRE's own STIX 2.1 release through STONES. The ATT&CK data in the graph is therefore STIX, and the adapter is what lets it join CVE, CWE and NIST data that was never STIX. A single SPARQL query can follow a product to its CVEs, their weaknesses, the CAPEC patterns and ATT&CK techniques related to those weaknesses, the groups that use the techniques, and the controls that mitigate them.

Two operational lessons are worth reporting. First, incremental delete-and-reload loaders leaked duplicate triples (720,699 removed in one clean-up) until loading was changed to insert only missing triples. Second, the graph is checked for logical inconsistency after each rebuild and each import; this caught a transform that typed URLs as strings against an `xsd:anyURI` range.

TABLE IV
KNOWLEDGE GRAPH CONTENTS, 18 OCTOBER 2026. THE CPE MATCH GRAPH HOLDS THE `AFFECTED` LINKS; THE MATERIALIZED-LINKS GRAPH HOLDS THE OTHER CROSS-FRAMEWORK LINKS, THE WITHIN-ATT&CK LINKS AND THE CLASS ASSERTIONS THEY NEED, OF WHICH THE TWO LARGEST LINK SETS ARE SHOWN.

Named graph	Triples
CVE records	30,556,713
CPE products	18,977,466
CPE match criteria	4,273,471
ATT&CK Enterprise (STIX, via STONES)	640,065
STONEWORK materialized links	427,521
Descriptions	394,586
CWE, CAPEC, NIST SP 800-53	30,584
Total (data graphs)	≈55.3 million
`exploitsWeakness` links	191,180
`mitigatesAttackPattern` links	6,392

B. Case study: CISA advisory AA26-281A

On 8 October 2026 CISA published advisory AA26-281A [21] with a STIX 2.1 file of 1,575 objects: 773 indicators, 25 attack patterns, 8 vulnerabilities, 8 malware and tool objects, and three threat actors. We imported it into a case (an `Investigation`), together with a small fictitious software inventory (Ivanti Connect Secure 9.0R2.1, GitLab Community Edition 13.10.2, Strapi 4.5.0 and PostgreSQL 15.4).

Validity and consistency. The OASIS validator found the file valid, with 775 warnings (773 indicators lacked a name and a description). The OWL consistency check found no contradictions with the ontologies or the reference graph.

Connection. Despite being valid, the file referred to ATT&CK techniques and CVEs only inside object names; none of its objects has an `external_references` entry. The reconciliation of Section V therefore proposed 35 `refersTo` links, all from identifiers in names and none projected automatically; the analyst confirmed all 35, rejecting none, each as an opinion.

What the links unlock. Following the confirmed `refersTo` links into the reference graph and its `affected` and `mitigatesAttackPattern` links (Table III), computed queries (no language model involved) reduced the inventory's roughly 1,700 known CVEs to three that the advisory makes urgent: CVE-2019-11510 against the exact Ivanti version, and CVE-2021-22205 and CVE-2023-22894 against the same GitLab and Strapi products. They ranked NIST controls by how many of the advisory's 16 linked techniques each mitigates (SI-4 System Monitoring and CM-2 Baseline Configuration 12 each, CM-6 Configuration Settings 11), and listed known ATT&CK groups by technique overlap (9 of 16 for the best match). None of this is stated in the advisory. Per Section IV, these are leads, not findings: the two product-level matches

need a version check, and the control ranking counts techniques a control is mapped to, which measures relevance, not protection.

C. Grounding a language-model agent

The same case was queried in plain English through the store's built-in language-model agent (GraphTalker, using Claude), which writes and runs its own SPARQL. Asked "Which of our systems does this advisory put at risk, and what should we patch first?" (Fig. 2), it ranked the Ivanti appliance above GitLab despite GitLab's higher CVSS score (10.0 against 9.9, the CVSS v3 base scores in the CVE records), because, in its account, the appliance is the perimeter VPN, the CVE names the exact version in use, and its credential-disclosure flaw is the entry point for the behavior the advisory describes. That is a judgment a CVSS sort cannot make. The inventory listed products and versions only; that the appliance sits at the perimeter is the agent's assumption about what such a product is for. The agent also drew the subgraph its answer rests on, and the SPARQL it ran is kept with the answer.

Graph evidence and agent reasoning. The answer mixes three kinds of statement, which deserve different trust. *From the advisory's STIX:* 749 of its 773 indicators have an `indicates` relationship to the threat actor Flax Typhoon; the CVEs, tools and techniques are listed in the report, but no STIX relationship ties them to any actor. *From the reference graph*, through the confirmed links: which inventory products each CVE affects, and the CVSS scores. *From the agent:* the ranking and its reasons, and, in Fig. 2(b), the edges from the advisory to the actors and from Flax Typhoon to the CVEs and tools, which it drew from the report's contents and text. The advisory frames its attribution with care ("Chinese Government-linked cyber threat actors"); the diagram states it as plain edges. Only the `AFFECTS` edges are links in the graph.

A follow-up question about controls exposed the failure mode that motivates this work. When given only the list of top controls, the agent explained each control's coverage from its own knowledge, and some of its control-to-technique pairings did not exist in the graph. We then gave the agent the graph's own `mitigatesAttackPattern` links for the case's techniques. Its answers leaned on those mappings, but the failure did not disappear. It still named SI-4 System Monitoring as the main control for automated exfiltration (T1020), local data staging (T1074.001) and archiving via a custom method (T1560.003). The mapping links SI-4 only to the parent technique T1560; it gives no NIST control for the other three at all. The agent's advice (monitor what cannot be prevented) is reasonable, but it presented a gap in the mapping as coverage. Grounding narrowed the model's improvisation without ending it. What the graph changes is that the claim can be checked: each pairing either is a `mitigatesAttackPattern` link or it is not, and this one took a single query to refute. Typed, explicit links do not make a fluent answer correct; they make it checkable.

The agent's answers are not deterministic, and cost money (on the order of US$0.20 to US$2 per question in our runs). The computed results above are deterministic and free to repeat. We treat the agent as a capable colleague whose draft is checked against evidence that is one query away.

VII. Discussion: Implications for STIX Communities

For producers. Name-embedded identifiers are the single largest obstacle we met. A STIX object that refers to an ATT&CK technique, CVE or CWE should carry a structured `external_references` entry with the source name and identifier. It costs a producer one field; without it, every consumer must guess, and a guessed link needs a human to confirm it.

For consumers. Keep the exchange form and the semantic form side by side. STIX is the right wire format; a formal binding such as STONES is the right working form for reasoning, joining and AI grounding. Keeping the binding faithful means the working form can always be serialized back to STIX for sharing.

For standards work. A future STIX revision would benefit from a normative or companion formal binding, maintained with the specification so that new objects arrive with their semantics. Qualified assertions also suggest a general pattern for confidence, provenance and analyst opinion that relationships, sightings and playbook steps (as in CACAO [5]) could share. Community vocabularies, kept as SKOS concepts rather than strings, can be extended without changing the core.

Limitations. Our evaluation is a single operator's experience and one detailed case study; it shows feasibility, not general effectiveness. The reconciler's confidence values are heuristic and have not been calibrated. The agent observations are qualitative.

VIII. Future Work

Measuring grounding. Section VI reports a qualitative effect. We plan a benchmark of questions whose answers can be checked against the graph (which controls mitigate a technique, which CVEs affect a product version) to compare an agent given only names with one given the typed links, and to measure how often each states a mapping the graph does not contain, alongside an evaluation with analysts outside the project. Because such claims are checkable, we also plan to check them automatically: each control-to-technique or CVE-to-product pairing in an answer is tested against the graph, and unsupported ones are flagged before the analyst reads the answer. We also plan a read-only tool interface so that any agent, not only one built into the store, can ground its answers in the reference graph.

Producers and reconciliation. A survey of public STIX advisories would show how often producers use structured `external_references`, and would provide labeled data for calibrating the reconciler's confidence values across producers.

Systems at Risk from CISA AA26-281A

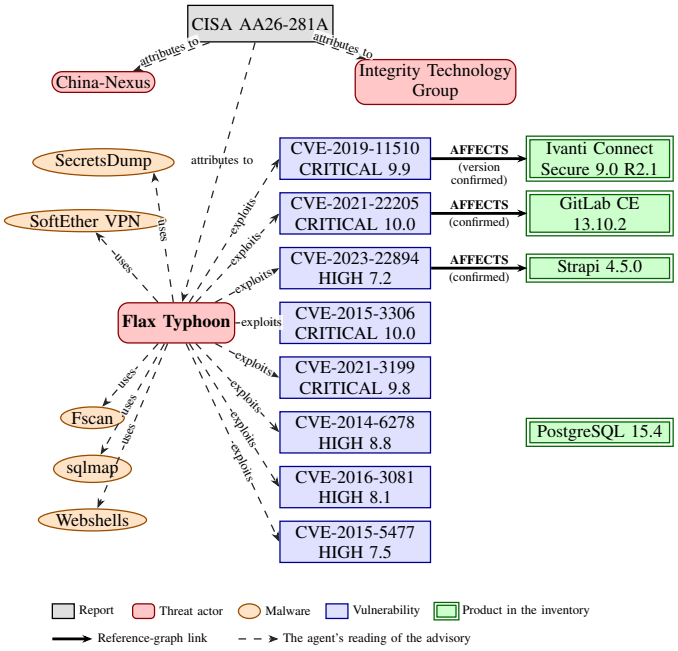
This advisory – *"Chinese Government-linked Cyber Threat Actors Combine Automated and Hands-on Hacking Tools to Steal Sensitive Data"* – is attributed to **Flax Typhoon**, backed by **Integrity Technology Group** (a China-based company), under the umbrella of **China-Nexus** state-sponsored operations. The case carries **749 indicators of compromise** tied directly to Flax Typhoon and documents an arsenal including **sqlmap**, **Webshells**, **SoftEther VPN**, **SecretsDump**, **Fscan**, **JuicyPotato**, and **NBTscan**.

Your inventory has **4 systems**. **Three are directly exposed** to CVEs this advisory weaponizes:

Patch Priority (ranked)

Priority	System	CVE	CVSS	Impact	Version Match
TIER 1	**Ivanti Connect Secure 9.0 R2.1**	CVE-2019-11510	**9.9 CRITICAL**	Unauthenticated arbitrary file read – attacker reads any file on the VPN appliance (credentials, session tokens, private keys)	**Version confirmed**: the CVE explicitly names "9.0 before 9.0R3.4" and you run 9.0 R2.1
TIER 2	**GitLab CE 13.10.2**	CVE-2021-22205	**10.0 CRITICAL**	Remote command execution via malicious image file – no authentication required, full server compromise	Confirmed: affects all GitLab CE/EE from 11.9 onward; 13.10.2 is in range
TIER 3	**Strapi 4.5.0 for Node.js**	CVE-2023-22894	**7.2 HIGH**	Attackers with admin panel access can extract password hashes and reset tokens via query filter abuse	Confirmed: affects Strapi through 4.5.5; your 4.5.0 is in range
–	**PostgreSQL 15.4**	(none in advisory)	–	Not targeted by this advisory. However, it has its own HIGH-severity CVEs (e.g. CVE-2023-5869, CVE-2024-10979 at CVSS 8.8) worth reviewing separately	Not in scope of this case

(a) Opening of the answer: summary and ranked patch list



(b) The subgraph behind the answer, redrawn from the agent's visualization

Fig. 2. The language-model agent's answer to "Which of our systems does this advisory put at risk, and what should we patch first?", asked from the AA26-281A case. The agent wrote and ran its own SPARQL over the case and the reference graph. The exposure of each product follows the case's confirmed `refersTo` links and the reference graph's `affected` links; the ranking and its reasoning are the model's. Panel (b) keeps the nodes, edges and labels of the agent's own drawing; color and shape give each node's type. Only the solid `AFFECTS` edges are graph links; the dashed edges are the agent's reading of the advisory. The 749 indicators are those with an `indicates` relationship to Flax Typhoon.

Standards. We intend to offer STONES to the STIX community as a candidate companion binding. With the CACAO community, we would like to work out a two-way mapping between CACAO playbooks and STONEWORK plans (Section IV-F), closing the gaps such as timeouts and exception handling, so that a playbook shared in CACAO can be run, recorded and reasoned over, and so that a hunt's steps, the evidence they produce and the analyst's conclusions share one vocabulary.

Reasoning. Disjointness axioms for STONEWORK's top-level classes are under discussion in its community. With them, the consistency check run on every import becomes a sharper test of data quality. We also plan to let analysts add their own findings to a case as typed, attributed triples.

IX. Conclusion

STIX made threat intelligence exchangeable. STONES makes it formally interpretable without changing how it is exchanged, and STONEWORK connects it to the vulnerability, weakness, technique and control knowledge that turns an advisory into a decision. In a 55-million-triple graph and a real advisory, the combination answered questions the advisory alone could not, and gave a language-model agent facts to stand on, against which its remaining guesses could be checked.

Availability

STONES and STONEWORK are open source: https://github.com/Cyber-Terrain-Ontology/stones and https://github.com/Cyber-Terrain-Ontology/stonework. Their namespaces resolve under https://cyberterrain.org/ns/, and their documentation, including interactive WebVOWL diagrams, is at https://cyberterrain.org/ns/stones/doc/ and https://cyberterrain.org/ns/stonework/doc/. The shared knowledge graph includes only sources whose licenses allow redistribution. The CIS Controls [29] are not included: their license does not permit derivative works, and linking them to other frameworks in a shared graph would be one. All results reported here use only the included sources.

Acknowledgment

Franz Inc. provided the AllegroGraph license used in this work. J. Aasman is CEO of Franz Inc., which develops AllegroGraph and GraphTalker. The authors used Claude (Anthropic) to help draft and edit the text and to write the round-trip test software; the authors reviewed all content and take responsibility for it.

References

[1] OASIS, "STIX version 2.1," OASIS Standard, B. Jordan, R. Piazza and T. Darley (eds.), 2021, https://docs.oasis-open.org/cti/stix/v2.1/stix-v2.1.html.

[2] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, S. Riedel, and D. Kiela, "Retrieval-augmented generation for knowledge-intensive NLP tasks," *Advances in Neural Information Processing Systems*, vol. 33, pp. 9459–9474, 2020.

[3] D. Edge, H. Trinh, N. Cheng, J. Bradley, A. Chao, A. Mody, S. Truitt, and J. Larson, "From local to global: A graph RAG approach to query-focused summarization," arXiv:2404.16130, 2024.

[4] V. Mavroeidis and S. Bromander, "Cyber threat intelligence model: An evaluation of taxonomies, sharing standards, and ontologies within cyber threat intelligence," in *Proc. European Intelligence and Security Informatics Conference (EISIC)*, 2017, pp. 91–98.

[5] OASIS, "CACAO security playbooks version 2.0," OASIS Committee Specification, 2023.

[6] Z. Syed, A. Padia, T. Finin, L. Mathews, and A. Joshi, "UCO: A unified cybersecurity ontology," in *Proc. AAAI Workshop on Artificial Intelligence for Cyber Security*, 2016.

[7] M. Iannacone, S. Bohn, G. Nakamura, J. Gerth, K. Huffer, R. Bridges, E. Ferragut, and J. Goodall, "Developing an ontology for cyber security knowledge graphs," in *Proc. 10th Annual Cyber and Information Security Research Conference (CISR)*, 2015.

[8] E. Asgarli and E. W. Burger, "Semantic ontologies for cyber threat sharing standards," in *Proc. IEEE Symposium on Technologies for Homeland Security (HST)*, May 2016, pp. 1–6.

[9] P. E. Kaloroumakis and M. J. Smith, "Toward a knowledge graph of cybersecurity countermeasures," The MITRE Corporation, Tech. Rep., 2021.

[10] E. Kiesling, A. Ekelhart, K. Kurniawan, and F. Ekaputra, "The SEPSES knowledge graph: An integrated resource for cybersecurity," in *Proc. 18th International Semantic Web Conference (ISWC)*, ser. LNCS, vol. 11779. Springer, 2019, pp. 198–214.

[11] K. Kurniawan, A. Ekelhart, and E. Kiesling, "An ATT&CK-KG for linking cybersecurity attacks to adversary tactics and techniques," in *Proc. ISWC 2021 Posters, Demos and Industry Tracks*, ser. CEUR Workshop Proceedings, vol. 2980, 2021.

[12] K. A. Akbar, F. I. Rahman, A. Singhal, L. Khan, and B. Thuraisingham, "The design and application of a unified ontology for cyber security," in *Proc. Information Systems Security (ICISS)*, ser. LNCS, vol. 14424, 2023, pp. 23–39.

[13] R. E. Hohimer, P. Winstanley, K. Studzinski, D. Dahleh, C. Glasgow, and J. Aasman, "Semantic interoperability in cybersecurity: Harmonizing threat intelligence with gist and gistCyber," in *Semantic Technology for Intelligence, Defense, and Security (STIDS)*, 2026, presented; proceedings forthcoming.

[14] V. Mavroeidis, R. Hohimer, T. Casey, and A. Jøsang, "Threat actor type inference and characterization within cyber threat intelligence," in *Proc. 13th International Conference on Cyber Conflict (CyCon)*, 2021, pp. 327–352.

[15] R. Hohimer and J. Aasman, "Beyond STIX: Next-level cyber-threat intelligence," Dark Reading, Mar. 2025, https://www.darkreading.com/threat-intelligence/beyond-stix-next-level-cyber-threat-intelligence.

[16] B. E. Strom, A. Applebaum, D. P. Miller, K. C. Nickels, A. G. Pennington, and C. B. Thomas, "MITRE ATT&CK: Design and philosophy," The MITRE Corporation, Tech. Rep., 2018.

[17] Joint Task Force, "Security and privacy controls for information systems and organizations," National Institute of Standards and Technology, Tech. Rep. NIST SP 800-53 Rev. 5, 2020.

[18] Center for Threat-Informed Defense, "Mapping ATT&CK to NIST SP 800-53," Mappings Explorer, https://center-for-threat-informed-defense.github.io/mappings-explorer/.

[19] W3C OWL Working Group, "OWL 2 web ontology language document overview (second edition)," W3C Recommendation, 2012.

[20] The MITRE Corporation, "ATT&CK STIX data," Enterprise, Mobile and ICS, version 20.1, 2026, https://github.com/mitre-attack/attack-stix-data.

[21] CISA, "Chinese government-linked cyber threat actors combine automated and hands-on hacking tools to steal sensitive data," Cybersecurity Advisory AA26-281A, Oct. 2026.

[22] Palo Alto Networks Unit 42, "STIX 2 reports," pan-unit42/iocs repository, https://github.com/pan-unit42/iocs.

[23] A. Miles and S. Bechhofer, "SKOS simple knowledge organization system reference," W3C Recommendation, 2009.

[24] H. Knublauch and D. Kontokostas, "Shapes constraint language (SHACL)," W3C Recommendation, 2017.

[25] T. Lebo, S. Sahoo, and D. McGuinness, "PROV-O: The PROV ontology," W3C Recommendation, 2013, https://www.w3.org/TR/prov-o/.

[26] D. Garijo and Y. Gil, "Augmenting PROV with plans in P-PLAN: Scientific processes as linked data," in *Proc. 2nd International Workshop on Linked Science (LISC)*, ser. CEUR Workshop Proceedings, vol. 951, 2012.

[27] S. Harris and A. Seaborne, "SPARQL 1.1 query language," W3C Recommendation, 2013.

[28] Cybersecurity and Infrastructure Security Agency, "Known exploited vulnerabilities catalog," 2026, https://www.cisa.gov/known-exploited-vulnerabilities-catalog.

[29] Center for Internet Security, "CIS critical security controls, version 8.1," 2024, licensed under CC BY-NC-ND 4.0. https://www.cisecurity.org/controls/v8-1.