CRC

CONSOLIDATE, REDUCE, CLOSE

The Minimum Surface Standard for AI-era regulated infrastructure.

MIT LICENSE · v1.2 · GENERAL REASONING, INC. · SEPTEMBER 2026

CRC begins from a structural premise: machine-speed systems can discover and exercise composition paths that human organizations historically failed to perceive as one system. The response is not a universal ban on connectivity. It is to make connectivity explicit, bounded, attributable, measurable, and smaller than the authority it could otherwise expose.

I. The Dunbar Perimeter

In complex organizations, teams naturally partition systems into domains they can understand: CRM, ERP, HRIS, identity, finance, compliance, security, analytics, and infrastructure. Those ownership boundaries are useful organizationally, but they are not automatically security boundaries. Data flows, identities, service accounts, APIs, agents, and automation routinely cross them.

The resulting system is a graph even when no one designed it as one. Each application is a node; identities, integrations, delegated permissions, workflow transitions, and network paths are edges. Human teams often understand the nodes more clearly than the composition of the edges.

> ***The Dunbar Perimeter is the security boundary created by limits of perception rather than by enforceable architecture.***

The CRC Standard uses that observation as a starting point. The objective is not to eliminate every relationship. It is to eliminate unowned relationships, minimize unjustified capability, and make remaining reachability explicit enough to govern and measure.

II. Why the threat model changed

In April 2026, Anthropic announced Project Glasswing and Claude Mythos Preview, describing a frontier model with unusually strong vulnerability-discovery and exploit-reasoning capabilities. Anthropic reported that Mythos Preview had found thousands of high-severity vulnerabilities, including vulnerabilities across major operating systems and browsers. Subsequent Glasswing updates reported more than 10,000 high- or critical-severity findings across partner software. [1][2][3]

The important architectural lesson is broader than any one model. Machine-speed systems can inspect larger dependency sets, retain more cross-domain context, and search composition paths without the same cognitive bottlenecks that historically constrained human attackers and defenders.

> ***Defenders may still evaluate systems one domain at a time. Machine-speed analysis can traverse the composition as one graph.***

CRC therefore treats attack-chain resistance as an architectural problem: reduce unnecessary nodes and capabilities, close ambient paths, govern deliberate crossings, and constrain the reachability that remains.

III. CRC: four structural pillars

Pillar	Objective	Normative direction
01 · CONSOLIDATE	Collapse unowned composition.	Move regulated workflows onto governed platforms where boundaries, identity, data flow, and authority can be owned as one architecture. The goal is fewer unowned integration edges, not a doctrinal ban on graph relationships.
02 · REDUCE	Remove unjustified ambient capability.	The execution environment should contain only capabilities justified by the machine mission. Shells, compilers, debuggers, package managers, browsers, drivers, and services are present only when their role is explicit and governed - not because a general-purpose substrate happened to include them.
03 · CLOSE	Default deny the network.	Known authenticated peers and explicitly governed crossings replace ambient network reachability. High-assurance profiles may require a single auditable egress; other profiles may permit multiple named crossings under the same principle.
04 · BOUNDARY	Govern every external crossing.	External AI inference, remote graph query, data exchange, administrative access, and autonomous actors are distinct crossing classes. Each must be named, authenticated, policy-bound, logged, and attributable.

These pillars remain the structural Minimum Surface Standard. The reachability and federation model introduced in v1.2 does not replace them. It makes the consequences of the remaining graph measurable.

IV. Governed Server Boundary prerequisite

A CRC Minimum Surface Score is meaningful only inside a governed server boundary under formal organizational control. The assessment is inapplicable when consequential infrastructure can be changed or reached through unmanaged paths that the assessed architecture does not govern.

Requirement	Condition
Organizational control	Servers and infrastructure are under formal organizational control.
No unmanaged agent path	AI agents with computer-use capability do not gain unmanaged access into the governed boundary.
Governed change pipeline	Changes to execution, network, identity, endpoint registry, and policy are explicitly authorized and recorded.
Current crossing registry	External inference, federation, administration, and other deliberate crossings are enumerated and current.
Attributable principals	Human, agent, application, service, and machine actions resolve to

Requirement	Condition
	governed principals.

> ***If the governed boundary is not real, a CRC score is not low. It is inapplicable.***

V. Minimum Surface Score: posture screen, not universal risk number

The four structural pillars are scored from 0 to 4 for a total Minimum Surface Score (MSS) of 0 to 16. The MSS is deliberately simple: it is an architectural posture screen. It is not intended to compress graph reachability, consequence, federation, uncertainty, and execution behavior into one scalar.

MSS	Posture	Meaning
0-4	Critical exposure	Ambient, unowned, or largely ungoverned surface dominates.
5-8	High exposure	Some controls exist, but substantial chain material and unowned composition remain.
9-12	Moderate exposure	Meaningful reduction exists; material structural gaps remain.
13-15	Reduced surface	Meets the current CRC regulated-deployment threshold; specific gaps remain.
16	Minimal surface	All four structural pillars at the Minimal level; reachability/federation still require their own profile and runtime measurements.

This separation is important. Two deployments may both score 16 structurally while exposing very different graph authority through permitted traversal. MSS answers whether the architecture is minimal and governed. The execution envelopes below answer how far a particular operation may reach.

VI. Isolation surface sizing

CRC retains local isolation-surface sizing as its own optimization problem. The candidate model balances compromise consequence against the complexity and workflow cost of creating more boundaries:

RECOMMENDED LOCAL SURFACE SIZE

```
S* = sqrt[(c_r + c_c f) / (p v_bar)]
```

Symbol	Meaning
S*	Recommended principals-per-isolation-boundary limit.
c_r	Fixed recurring cost of maintaining an additional isolation boundary.
c_c	Cross-boundary coordination cost.
f	Expected frequency of legitimate cross-boundary coordination.

p	Compromise probability term for the modeled period/profile.
v_bar	Average consequence/value exposed per principal in the surface.

The formula is a partitioning model, not a graph-traversal model. v1.2 therefore does not overload S* with edge count or federation variables. Those are separate dimensions with different semantics and qualification methods.

VII. Reachability exposure

A small local surface can still be dangerous if a compromise or authorized query can traverse high-consequence external relationships. Raw edge count is therefore a guardrail, not the risk model. An inert descriptive edge is not equivalent to an authority-bearing relationship into a production domain.

```
CANDIDATE CONSEQUENCE-WEIGHTED EXPOSURE
X(P) = sum w_e   for e in R(P)
w_e = q_e * V_e * A_e * B_e
```

Symbol	Meaning
P	Candidate execution plan.
R(P)	Boundary-significant relationships actually exercised by the plan.
q_e	Effective traversability under the current principal and operation.
V_e	Consequence/value exposed beyond the relationship.
A_e	Authority amplification associated with the crossing.
B_e	Boundary significance or domain-sensitivity factor.

The exact weighting and calibration of q_e, V_e, A_e, and B_e are qualification candidates, not frozen universal constants. What is normative is the distinction: effective reachable authority and consequence matter more than connection count alone.

Traversal envelope

```
HARD EXECUTION CONSTRAINTS
X(P) <= X_max
D(P) <= D_max
E(P) <= E_max
N(P) <= N_max
edge_types(P) subset_of T_E
```

A CRC profile may constrain consequence-weighted exposure (X), traversal depth (D), edges examined/traversed (E), nodes reached or returned (N), and permitted relationship classes (T_E). These limits are intended to be machine-enforceable and observable during execution.

VIII. Federation envelope

Federation introduces a distinct class of exposure. A query can remain within a graph-traversal budget while moving excessive information across governed surfaces, coordinating inconsistent snapshots, or crossing authority domains unnecessarily. CRC therefore models federation as a vector rather than forcing unlike quantities into one arbitrary score.

```
FEDERATION VECTOR
F(P) = (S_f, C_f, B_f, V_f, A_f, U_f)
```

Dimension	Meaning
S_f	Participating governed surfaces.
C_f	Cross-surface requests, exchanges, or boundary crossings.
B_f	Bytes, records, bindings, or result material crossing boundaries.
V_f	Consequence-weighted information exported from surfaces.
A_f	Authority differentials crossed.
U_f	Uncertainty from snapshot age, incomplete remote knowledge, or coordination state.

> ***Federation is permitted only inside a declared envelope. Optimization happens after admissibility, not instead of it.***

Permitted, planned, and exercised federation

The standard distinguishes the federation that exists from the federation a principal could use and from what a query actually exercises:

```
FEDERATION SCOPE
S_available  -> all configured federation members
S_permitted  -> surfaces authorized for this principal/operation
S_planned    -> surfaces selected by the candidate plan
S_actual     -> surfaces actually contacted at runtime
```

This distinction makes over-broad authority visible. A federation may contain twenty surfaces while a query requires only three. A compliant planner should not contact the other seventeen merely because they are available.

IX. Constrained query planning

CRC places authority and exposure constraints ahead of physical optimization. A plan that crosses a forbidden edge or exceeds a federation budget is not an expensive plan. It is not a candidate plan.

```
PLAN SELECTION
P* = arg min C_execution(P)
subject to P |= CRC AND Authority AND Provenance
```

Stage	Rule
1. Admissibility	Eliminate plans that violate identity, authority, edge-class, traversal,

	federation, or provenance constraints.
2. Exposure minimization	Among permissible plans, prefer plans that avoid unnecessary cross-surface movement and graph expansion when semantics are equivalent.
3. Execution optimization	Use indexes, join ordering, predicate pushdown, aggregation, projection, caching, and other physical techniques inside the permitted space.
4. Attestation	Record the permitted envelope, selected plan, actual exposure, result identity, provenance, and authority used.

A weighted objective in which enough latency improvement mathematically compensates for impermissible exposure is not the default CRC semantics. Security constraints are hard constraints unless an explicitly authorized profile states otherwise.

X. Independent federation qualification

CRC v1.2 proposes an independent validation methodology so that the standard is not only tested against General Reasoning software. Existing RDF/SPARQL federation engines provide useful external controls because they can execute semantically equivalent queries over multiple independent graph endpoints.

Engine	Role in CRC qualification	Why it is useful
Eclipse RDF4J FedX	Primary reference	Transparent federation of multiple SPARQL endpoints; automatic source selection; query optimization and join-order logic; current RDF4J 6.0 adds grouped source selection to reduce remote source-selection round trips. [4][5][6]
Apache Jena ARQ / Fuseki	Explainable baseline	SPARQL engine with remote federated-query support. Explicit SERVICE-based workloads make boundary crossings visible and are useful as a simple control. [7]
GraphDB	Production-oriented comparison	Current GraphDB documentation describes SPARQL federation, remote repositories, and federation-related integrations. It is useful as a commercial graph-system comparison, not as a source of CRC semantics. [8][9]

Reference workload

The test harness should publish identical logical graph fixtures across 4, 8, and larger endpoint sets. Each endpoint represents a governed surface. Fixtures should deliberately vary selectivity, authority, edge classes, graph depth, result cardinality, and data consequence.

Measurement	Qualification question
Logical correctness	All engines/plans must return the same answer for the same declared snapshot semantics.
Surfaces	available / permitted / planned / actual.
Remote work	Requests, subqueries, statement patterns, and cross-surface joins.
Movement	Bytes, records, bindings, graph fragments, and aggregates transferred.

Measurement	Qualification question
Traversal	Edges examined, boundary edges crossed, max depth, nodes reached.
Exposure	X(P) and each component of F(P), using published fixture weights.
Performance	Latency, CPU, memory, remote round trips, and coordinator work.
Failure behavior	Timeout, stale endpoint, missing endpoint, authorization denial, partial result, and uncertainty handling.

Plan-shape experiment

Plan	Shape	Purpose
A · Pull fragments	Remote surfaces return broad graph fragments; joins happen centrally.	Expected to maximize cross-boundary movement.
B · Push predicates	Filtering occurs at each surface; qualifying bindings return to coordinator.	Tests source-local reduction.
C · Push predicate + projection/aggregation	Surfaces return minimal attributed results required for final answer.	Tests exposure-aware pushdown.

If A, B, and C are semantically equivalent but produce materially different X(P) or federation vectors, the experiment demonstrates a central CRC proposition: distributed query plans can have equal logical meaning and unequal boundary exposure.

XI. Implementation example: AletheiaGraph and Marshaller

AletheiaGraph is a General Reasoning implementation candidate for CRC-aware graph execution. It is being designed as an authoritative graph persistence, query, reasoning, proof, and governance engine rather than as a federation benchmark. CRC provides the admissibility model; AletheiaGraph and Marshaller provide one place to implement and qualify it.

Layer	CRC relevance
Authoritative graph	Vertices, edges, properties, and committed versions form the source of truth.
Native query planner	Filtering, joins, projection, ordering, aggregation, pagination, and bounded traversal are represented in a graph-native algebra.
Reasoning	Bounded Prolog inference operates over authoritative graph state.
Proof and provenance	Conclusions can preserve relationships to exact committed evidence and distinguish stored, inferred, and proposed state.
CRC-aware traversal	Principal, operation, surface, edge class, and exposure envelope constrain which candidate plans are admissible.
Marshaller federation	Distributed execution can prefer predicate pushdown, surface-local filtering, narrow projection, and attributed aggregates so that less material crosses boundaries.

Layer	CRC relevance
Chandra attestation	Query proposal, authorization, selected plan, actual exposure, and resulting evidence can be tied to attributed audit records.

> ***AletheiaGraph is not evidence that CRC is correct. It is an implementation example. FedX, ARQ, GraphDB, and other independent engines provide the external comparison needed to keep the standard honest.***

Execution record candidate

```
CRC QUERY ATTESTATION
permitted = (T_E, X_max, D_max, E_max, N_max, F_max)
planned   = plan_id + S_planned + predicted exposure
actual    = X(P), D(P), E(P), N(P), F(P)
evidence  = graph_state + provenance + principal + authority
```

The design objective is to make “why this query was allowed to go this far” as inspectable as “why this query returned this answer.”

XII. General Reasoning reference architecture

The General Reasoning stack is one reference architecture for implementing CRC across identity, persistence, reasoning, workflow, audit, and machine boundary. Components are at different implementation and qualification stages; the standard does not require this stack.

Component	Role
AegisGenera / AegisBSD	Machine and execution boundary. Purpose determines what software is present; governance determines how justified capabilities are used.
Chandra Passport	Governed principal, role, scope, and authority substrate for humans, agents, applications, services, and machines.
AletheiaGraph	Authoritative state, graph query, bounded traversal, reasoning, proof, provenance, and truth/evidence semantics.
Marshaller	Cross-surface query and movement layer; intended home for exposure-aware federation planning.
Chandra Protocol	Attributed, append-only evidentiary record for consequential acts and state transitions.
DXMachine / applications	Workflow and application surfaces operating above the governed substrate.

XIII. What CRC v1.2 claims - and what it does not

Category	Position
Claims	Structural minimization is measurable; local surface sizing and graph/federation exposure are different problems; traversal/federation budgets can be enforced as hard query constraints; permitted and exercised

Category	Position
	exposure should be distinguished; independent federation engines can be used to qualify the model.
Does not claim	That fewer edges are always safer; that one universal edge limit fits all domains; that X(P) coefficients are already empirically calibrated; that a high MSS alone proves safe graph execution; or that AletheiaGraph is required for CRC compliance.
Qualification target	Publish fixtures, instrumentation, formulas, profiles, and repeatable FedX/ARQ/GraphDB/AletheiaGraph workloads so that claims can be challenged and reproduced.

XIV. Path forward

CRC began as a minimum-surface architecture for an AI-era threat model. v1.2 extends that premise into graph and federated execution because modern autonomous systems do not merely call applications; they query, traverse, infer, combine, and act across relationships.

The governing principle is simple: a system should not gain broader reachability merely because its optimizer discovered a cheaper path. Architecture defines what is permitted. Optimization chooses among what remains.

> ***CRC = surface partitioning + reachability control + federation control + measured execution exposure.***

The immediate standardization work is empirical: freeze a public federation fixture, implement the qualification harness, collect measurements across independent engines, calibrate domain profiles, and publish where the proposed exposure model succeeds or fails. That is the path from an architectural thesis to an engineering standard.

References

[1] Anthropic, "Project Glasswing: Securing critical software for the AI era," Apr. 7, 2026. https://www.anthropic.com/glasswing

[2] Anthropic, "Assessing Claude Mythos Preview's cybersecurity capabilities," Apr. 7, 2026. https://www.anthropic.com/research/mythos-preview

[3] Anthropic, "Project Glasswing: An initial update," May 22, 2026. https://www.anthropic.com/research/glasswing-initial-update

[4] Eclipse RDF4J, "Federation with FedX" / RDF4J documentation. https://rdf4j.org/documentation/

[5] Eclipse RDF4J, FedX API and federation evaluation strategy, RDF4J 6.0.1. https://rdf4j.org/javadoc/latest/org/eclipse/rdf4j/federated/FedX.html

[6] Eclipse RDF4J, "RDF4J 6.0.0 Released," Jul. 2, 2026. https://rdf4j.org/news/2026/07/02/rdf4j-6.0.0-released/

[7] Apache Jena, ARQ / SPARQL and remote federated query documentation. https://jena.apache.org/

[8] GraphDB documentation, remote locations and SPARQL federation, current 11.x docs. https://graphdb.ontotext.com/documentation/11.5/connecting-to-remote-locations.html

[9] GraphDB documentation, GraphDB APIs and RDF4J/SPARQL integration. https://graphdb.ontotext.com/documentation/11.5/clients-and-apis.html