

Webinar

Toward a Systemic Will to Live – Patterns of Self-Organizing Agile Security

Rick Dove

Last Updated: 8 September 2011

**(subject to aperiodic and continuous updates at
www.parshift.com/s/TowardsSystemicWillToLive.pdf)**

**Portions of this work are sponsored by the Department of Homeland Security
under contract D10PC20039. The content of the material
contained herein does not necessarily reflect
the position or policy of the Government, and
no official endorsement is implied.**

Abstract

This talk puts a focus on systems that have an awareness of their environment, and that are sensitive to anomalous changes that might signal a threat.

Sensitivity to anomalous change is most useful when every possible change, within a domain of interest, accurately triggers attention – meaning no false positives (crying wolf) and no false negatives (undetected anomalies).

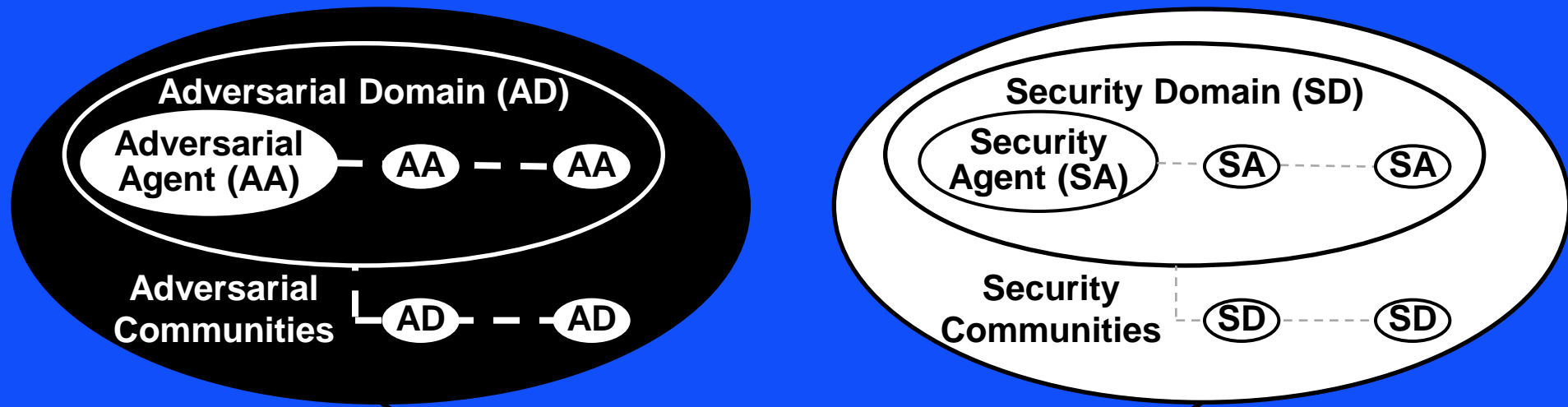
We will first explore four inspirational patterns from natural systems that effectively process noisy sensory input from uncertain & changing environments:

- horizontal gene/meme transfer,**
- bow tie processors,**
- proactive anomaly search, and**
- hierarchical sensemaking.**

Then the architecture of the biological immune system will be examined, and subsequently grounded with an artificial immune system example under development for a resilient cyber-network sense and sensemaking application.

Of special note is new anomaly detection technology that enables high fidelity immune system-like performance, effectively covering a vast detection space of 10 to the 15th anomalies in the example shown, with higher capacities practical.

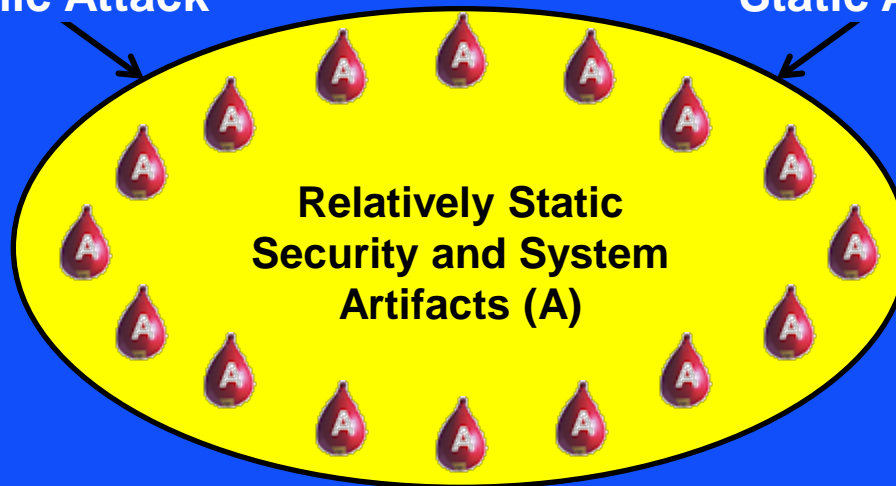
General Current Situation



Dynamic Attack

Static Artifact

Dynamic attack includes human and systemic adaptive control preying upon fixed artifact defenses.



Static artifacts are systems with and without security measures, updated occasionally.



**Static
System**

**Dynamic
Adversary**

Asymmetries

Adversary is a natural system, security strategy is an artificial system

Adversary leads with innovation and evolution

Adversary self-organizes as a dynamic system-of-systems

... up next ...

Pattern (Language) Project

Some Dynamic Self Organizing System-of-System Security Patterns

Pattern employment on the SORNS project

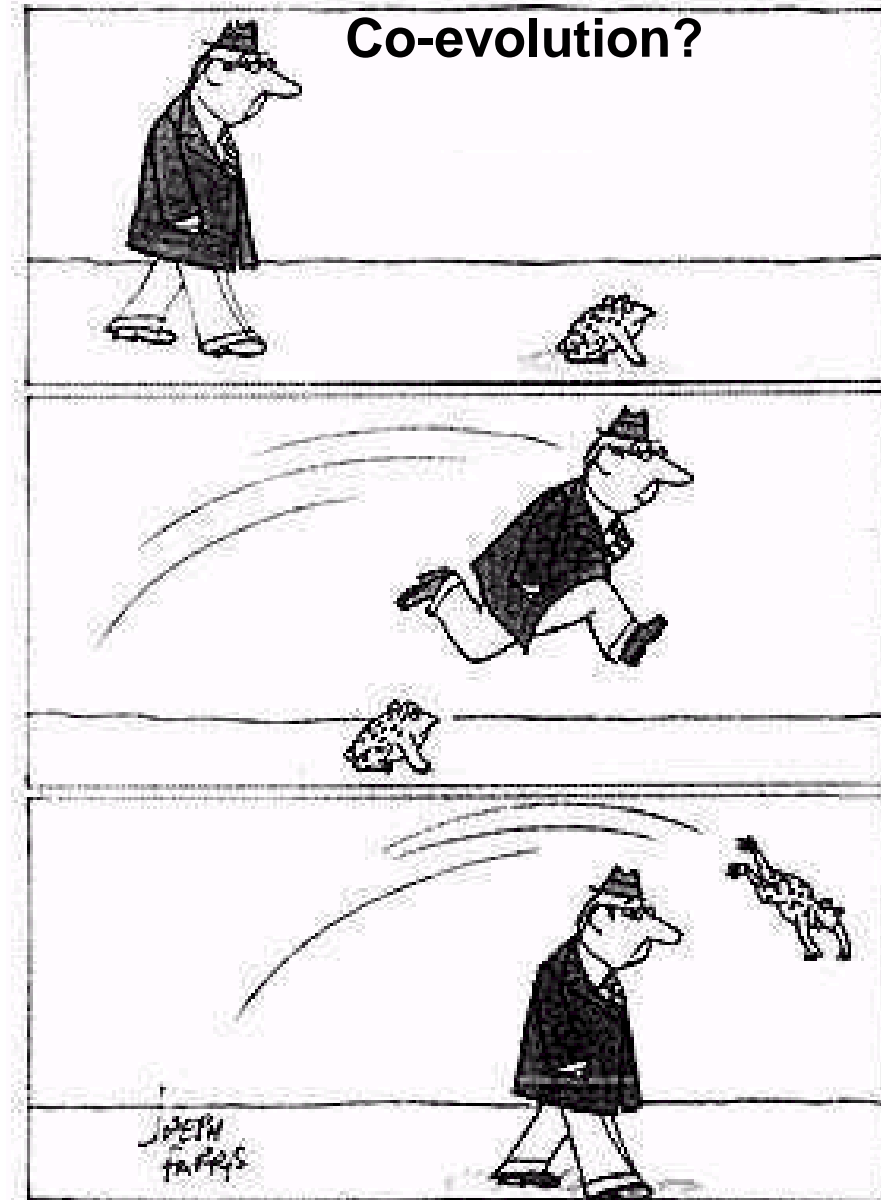
Adversarial Advantage

Architecture:

- Multi-agent
- Loosely coupled
- Self organizing
- Systems-of-systems

Behavior:

- Swarm intelligence
- Tight learning loops
- Fast evolution
- Adaptive innovation



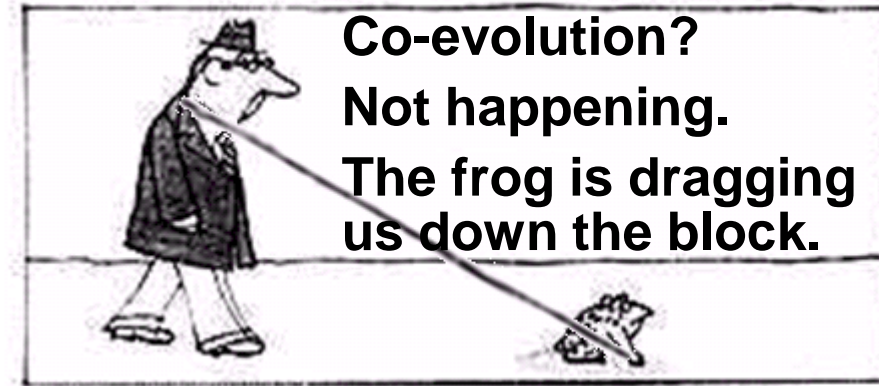
Adversarial Advantage

Architecture:

- Multi-agent
- Loosely coupled
- Self organizing
- Systems-of-systems

Behavior:

- Swarm intelligence
- Tight learning loops
- Fast evolution
- Adaptive innovation



**We are not in an arms race
– we haven't engaged.**

Mirror the Enemy



Agile system security, as a minimum, must mirror the agile characteristics exhibited by the system attack community:

- [S] Self-organizing – with humans embedded in the loop, or with systemic mechanisms.**
- [A] Adapting to unpredictable situations – with reconfigurable, readily employed resources.**
- [R] Reactively resilient – able to continue, perhaps with reduced functionality, while recovering.**
- [E] Evolving in concert with a changing environment – driven by vigilant awareness and fitness evaluation.**
- [P] Proactively innovative – acting preemptively, perhaps unpredictably, to gain advantage.**
- [H] Harmonious with system functional purpose – aiding rather than degrading system and user productivity.**

Inspirational Patterns

**from natural systems that effectively process
noisy sensory input from uncertain and changing environments**

Evolution and Innovation

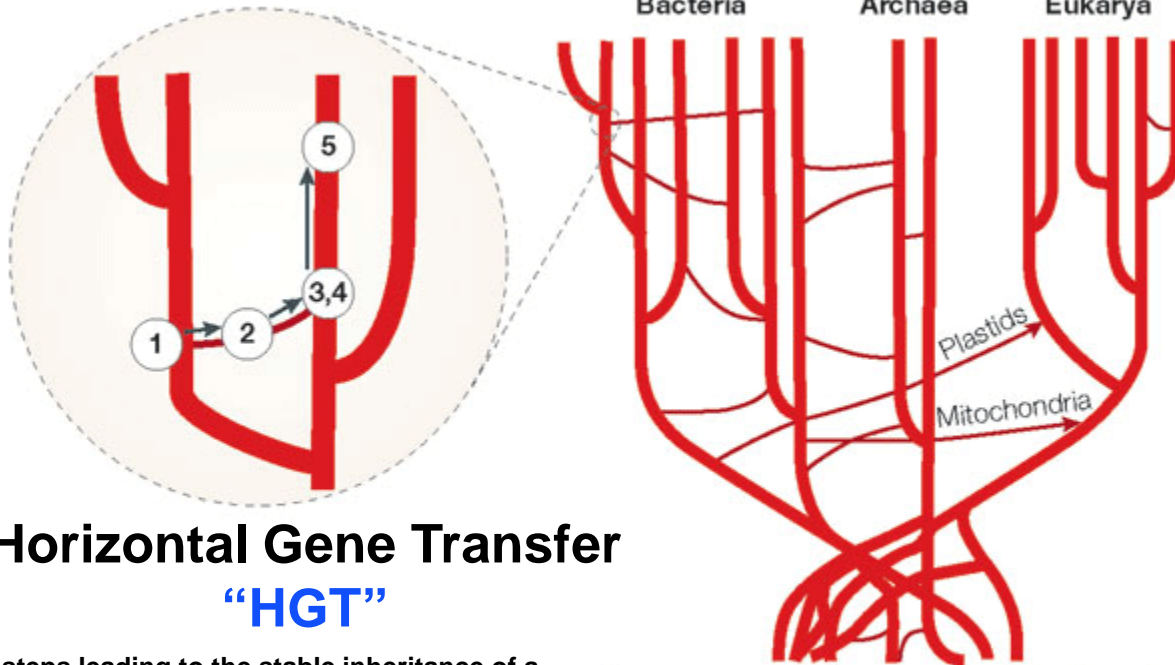
Woese, Carl. 2000. Interpreting the universal phylogenetic tree. PNAS. 97(15):8392-6. www.ncbi.nlm.nih.gov/pmc/articles/PMC26958/pdf/pq008392.pdf

Carl Woese: “Vertically generated and horizontally acquired variation could be viewed as the yin and the yang of the evolutionary process.

“Vertically generated variation is necessarily highly restricted in character; it amounts to variations on a lineage’s existing cellular themes.

Horizontal transfer, on the other hand, can call on the diversity of the entire biosphere, molecules and systems that have evolved under all manner of conditions, in a great variety of different cellular environments.

Thus, horizontally derived variation is the major, if not the sole, evolutionary source of true innovation.”



Horizontal Gene Transfer “HGT”

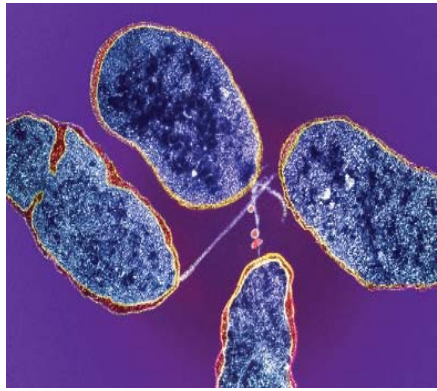
5 steps leading to the stable inheritance of a transferred gene in a new host. Figure Smets, Barth F. and Tamar Barkay. 2005. Horizontal gene transfer: perspectives at a crossroads of scientific disciplines. *Nature Reviews Microbiology* 3, 675-678 (Sep 2005).

Common ancestral community of primitive cells

Copyright © 2005 Nature Publishing Group

“The vast majority, between 88% and 98%, of the expansions of protein families [in eight studied prokaryote clades] are due to HGT. ... Xenologs [external transfers] have an average age of introduction that is twice that of paralogs [internal transfers]. Xenologs are therefore more persistent.” Treangen, Todd J. and Eduardo P. C. Rocha. 2011. Horizontal Transfer, Not Duplication, Drives the Expansion of Protein Families in Prokaryotes. *PLoS Genetics* 7:1, January.

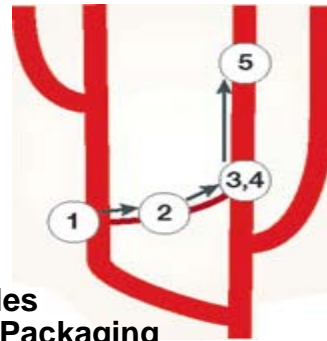
Pattern: Horizontal Gene/Meme Transfer



Available high variety cellular organisms

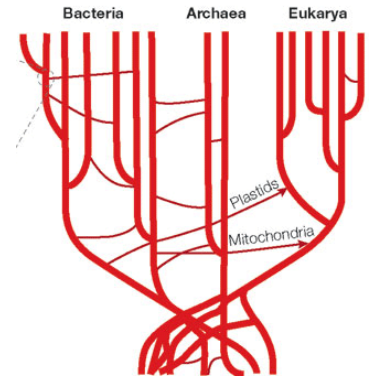


Two modular gene pools



Rules

1. Packaging
2. Transfer
3. Entry
4. Establishment
5. Inheritance



Innovative adaptation and evolution

Horizontal gene transfer speeds up innovative short-term adaptation and long-term evolution

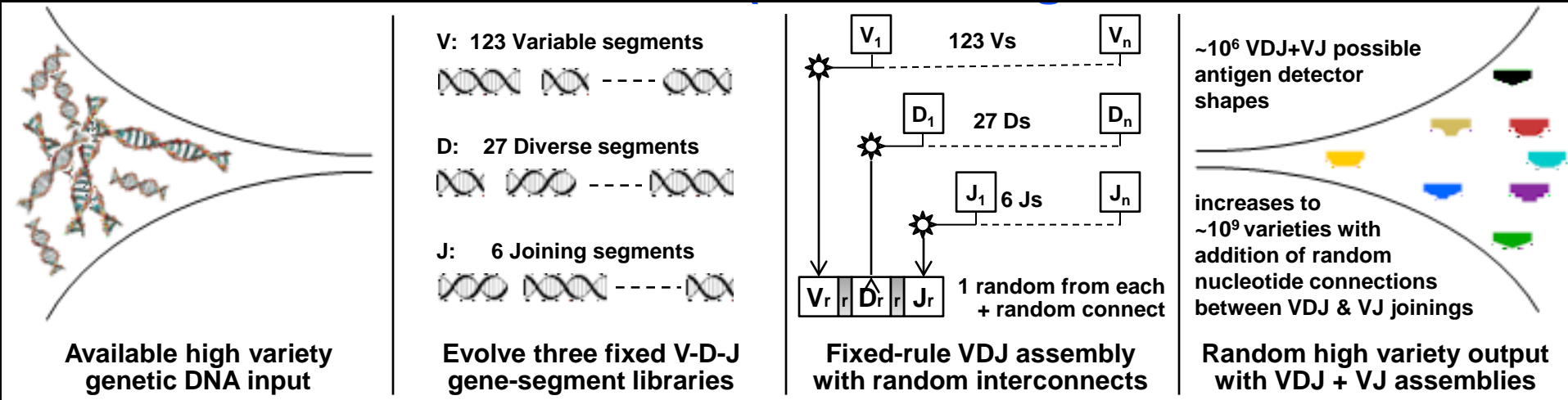
Context: When conditions deteriorate, it makes a lot of sense to try to scavenge DNA from your neighbors. Horizontal gene transfer facilitates a fast microbial adaptation to stress. Higher-than-suspected transfer rates among microbes living in nutrient-poor environments, where sharing genes may be key to survival, has been observed. Evidence indicates that organisms limit gene exchange to microbes on nearby branches of the family tree, probably because their chromosomes share certain characteristics. Genes appear to be exchanged between species with similar chromosomal structures (Pennise 2011).

Problem: Situational or environmental changes that threaten fitness or survival of the organism.

Forces: Short-term adaptability vs. long-term-evolvability, horizontal gene transfer speeds the development of new traits by a factor of 10,000 (Woese 2000, Pennise 2011).

Solution: Incorporate appropriate genetic material from other organisms that have developed compatible and useful situational fitness. Mobile genes don't just help a community survive, they also provide the grist for evolutionary innovations.

Pattern: Bow Tie Processor (assembler/generator/mediator)



Millions of random infection detectors generated continuously by fixed rules and modules in the "knot"

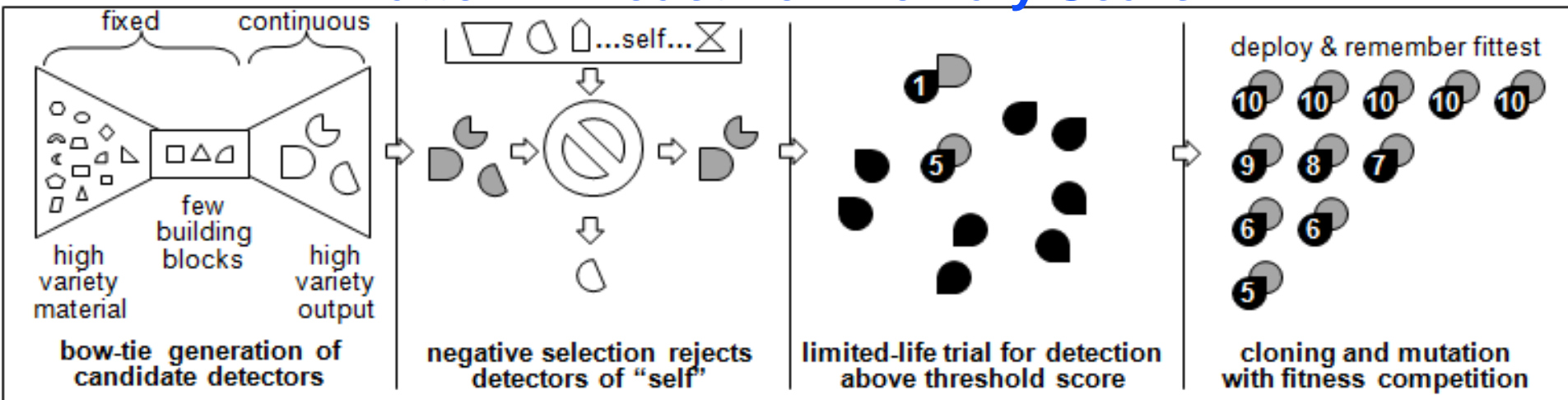
Context: Complex system with many diverse inputs and many diverse outputs, where outputs need to respond to many needs or innovate for many or unknown opportunities, and it is not practical to build unique one-to-one connections between inputs and outputs. Appropriate examples include common financial currencies that mediate between producers and consumers, the adaptable biological immune system that produces proactive infection detectors from a wealth of genetic material, and the Internet protocol stack that connects diverse message sources to diverse message sinks.

Problem: Too many connection possibilities between available inputs and useful outputs to build unique robust, evolving satisfaction-processes between each.

Forces: Large knot short-term-flexibility vs small knot short-term-controllability and long-term-evolvability (Csete 2004); robustness to known vs fragility to unknown (Carlson 2002).

Solution: Construct relatively small "knot" of fixed modules from selected inputs, that can be assembled into outputs as needed according to a fixed protocol. A proactive example is the adaptable immune system that constructs large quantities of random detectors (antigens) for unknown attacks and infections. A reactive example is a manufacturing line that constructs products for customers demanding custom capabilities.

Pattern: Proactive Anomaly Search



Speculative generation and mutation of detectors recognizes new attacks like a biological immune system

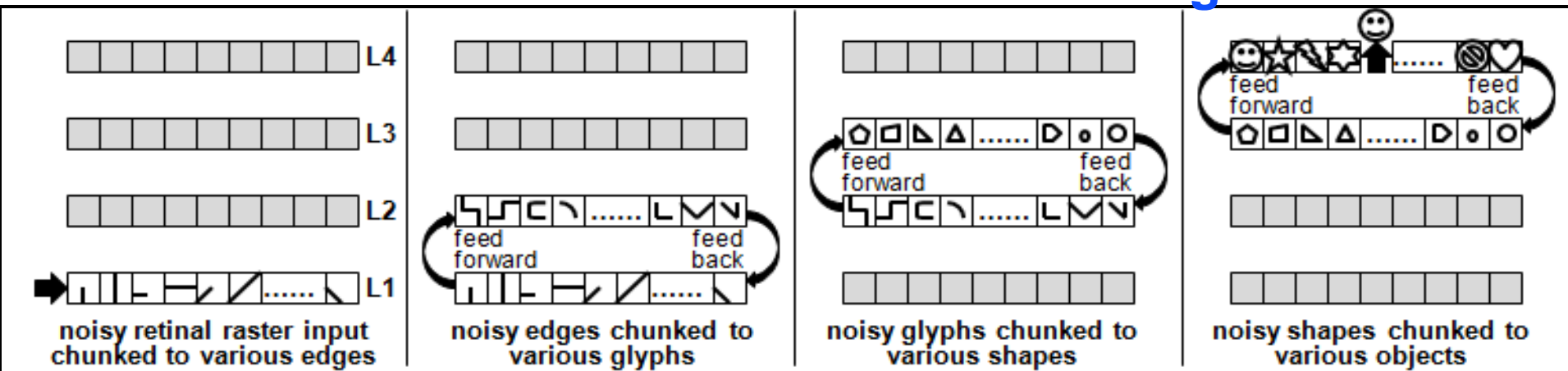
Context: A complex system or system-of-systems subject to attack and infection, with low tolerance for attack success and no tolerance for catastrophic infection success; with resilient remedial action capability when infection is detected. Appropriate examples include biological organisms, and cyber networks for military tactical operations, national critical infrastructure, and commercial economic competition.

Problem: Directed attack and infection types that constantly evolve in new innovative ways to circumvent in-place attack and infection detectors.

Forces: False positive tradeoffs with false negatives, system functionality vs functionality impairing detection measures, detectors for anything possible vs added costs of comprehensive detection, comprehensive detection of attack vs cost of false detection of self.

Solution: A high fidelity model of biological immune system antibody (detection) processes that generate high quantity and variety of anticipatory speculative detectors in advance of attack and during infection, and evolve a growing memory of successful detectors specific to the nature of the system-of-interest.

Pattern: Hierarchical Sensemaking



Four level feed forward/backward sense-making hierarchy modeled on visual cortex

Context: A decision maker in need of accurate situational awareness in a critical dynamic environment. Examples include a network system administrator in monitoring mode and under attack, a military tactical commander in battle, and the NASA launch control room.

Problem: A very large amount of low-level noisy sensory data overwhelms attempts to examine and conclude what relevance may be present, most especially if time is important or if sensory data is dynamic.

Forces: amount of data to be examined vs time to reach a conclusion, number of ways data can be combined vs number of conclusions data can indicate, static sensory data vs dynamic sensory data, noise tolerated in sensory data vs cost of low noise sensory data.

Solution: Using a bow-tie process, each level looks for a specific finite set of data patterns among the infinite possibilities of its input combinations, aggregating its input data into specific chunks of information. These chunks are fed-forward to the next higher level, that treats them in turn as data further aggregated into higher forms of information chunks. Through feedback, a higher level may bias a lower level to favor certain chunks over others, predicting what is expected now or next according to an emerging pattern at the higher level. Each level is only interested in a small number of an infinite set of data-combination possibilities, but as aggregation proceeds through multiple levels, complex data abstractions and recognitions are enabled.

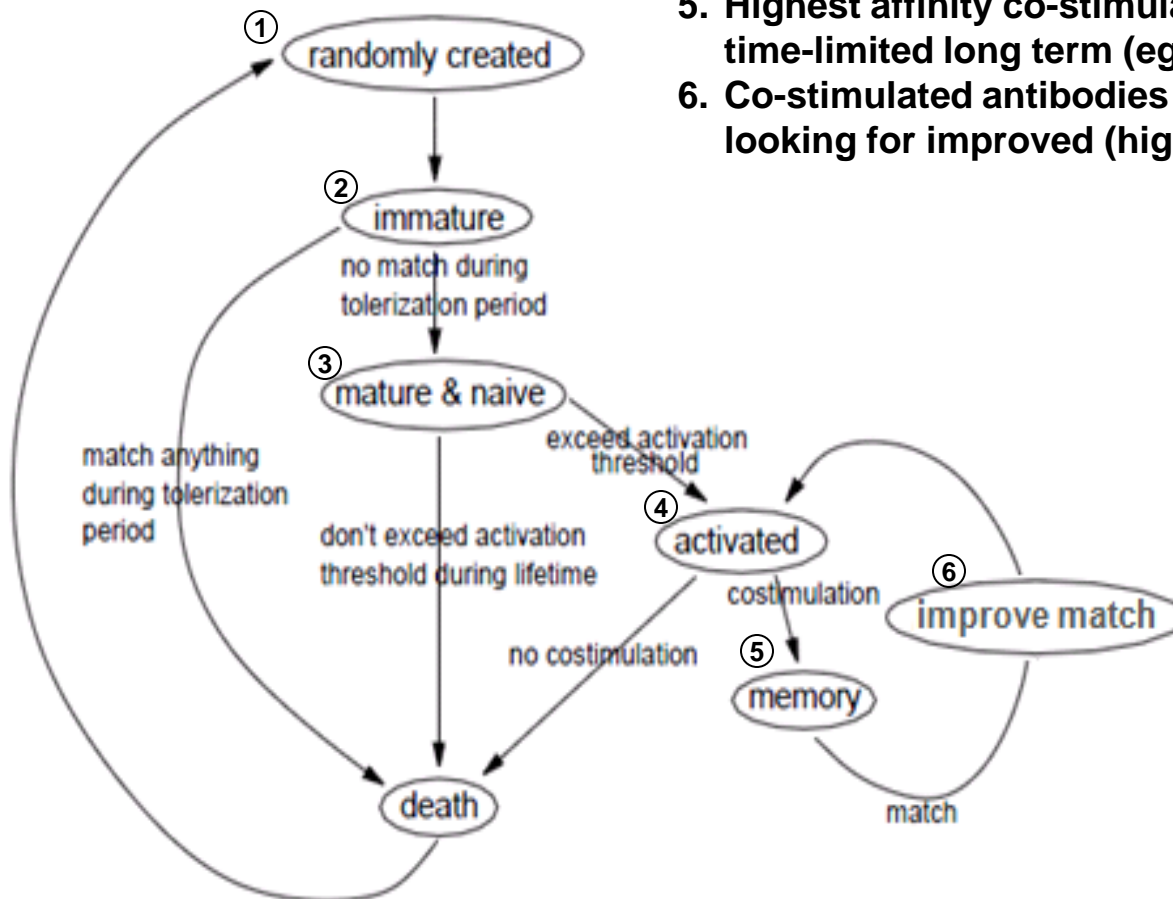
BIS Architecture

(Biological Immune System)

Antibody Creation & Life Cycle

General antibody life cycle: creation, false-positive testing, deployment efficacy or termination, mutation improvement, and long-term memory.

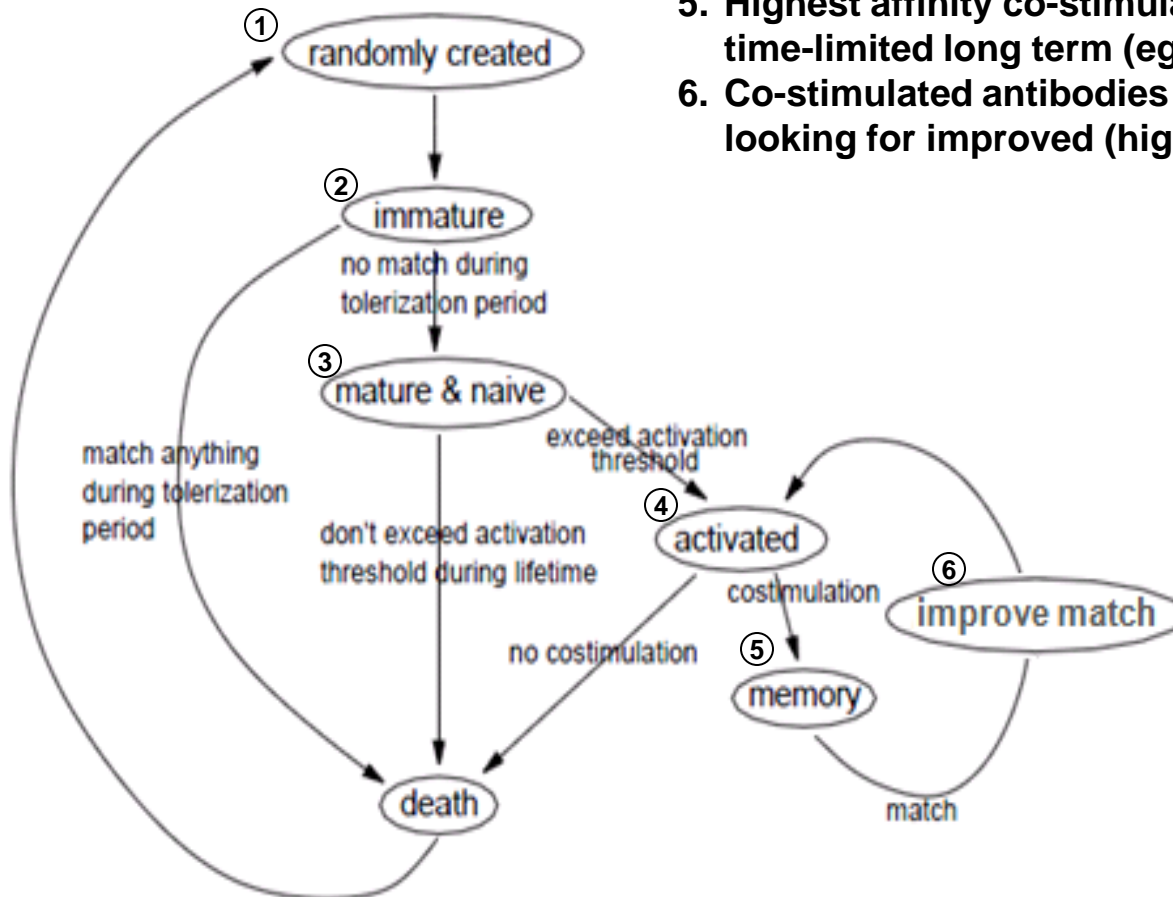
1. Candidate antibody semi-randomly created.
2. Tolerization period tests immature candidates for false-positive matches.
3. Mature & naïve antibodies put into time limited service.
4. Activated (B-cell) antibodies need co-stimulation (by T-cells) to ensure “improvement” didn’t produce auto-reactive result, non-activated & non-co-stimulated candidates die when time limit ends.
5. Highest affinity co-stimulated antibodies are remembered for time-limited long term (eg, many years, decades).
6. Co-stimulated antibodies are cloned with structured mutations, looking for improved (higher) affinity scores.



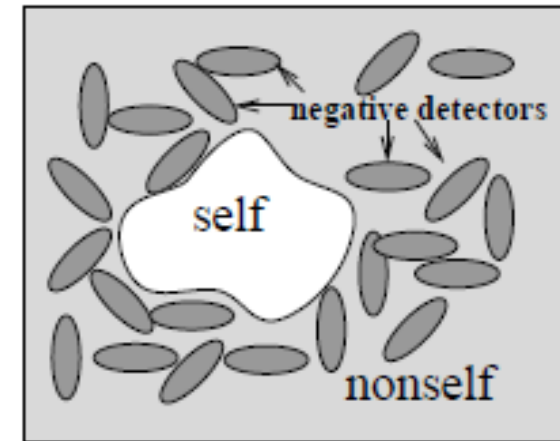
Antibody Creation & Life Cycle

General antibody life cycle: creation, false-positive testing, deployment efficacy or termination, mutation improvement, and long-term memory.

1. Candidate antibody semi-randomly created.
2. Tolerization period tests immature candidates for false-positive matches.
3. Mature & naïve antibodies put into time limited service.
4. Activated (B-cell) antibodies need co-stimulation (by T-cells) to ensure “improvement” didn’t produce auto-reactive result, non-activated & non-co-stimulated candidates die when time limit ends.
5. Highest affinity co-stimulated antibodies are remembered for time-limited long term (eg, many years, decades).
6. Co-stimulated antibodies are cloned with structured mutations, looking for improved (higher) affinity scores.



Shape/Pattern Space $\sim 10^9$



Self nonself discrimination: A universe of data points is partitioned into two sets – self and nonself. Negative detectors cover subsets of non-self. From (Esponda 2004)

SORNS Application

Self Organizing Resilient Network – Sensing

Proposed Basic Concept

- Explore advantages of new pattern processor
- Distribute collaborating detector agents at all network endpoints
- Artificial Immune System pattern detection a la Forest/Hofmeyr, et al.
- Hierarchical pattern detection a la Fink/Fulp, Hawkins/George, et al.
- Implement the work of others in a less-constraining technology that can better approach high fidelity natural-system performance

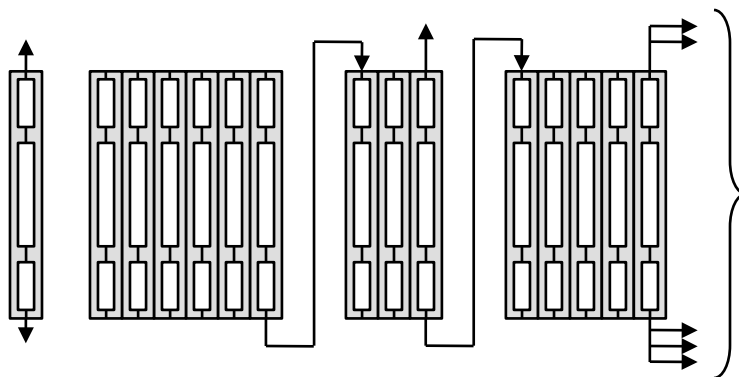
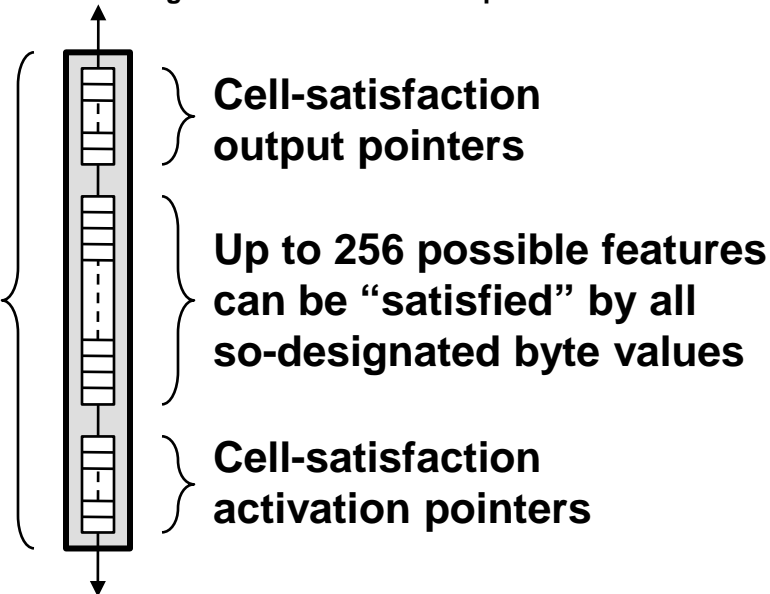
Reconfigurable Pattern Processor

Reusable Cells Reconfigurable in a Scalable Architecture

www.parshift.com/Files/PsiDocs/Pap090303-PatternRecognitionWithoutTradeoffs.pdf

**Independent detection cell:
content addressable
by current input byte**

**If active, and satisfied with
current byte, can activate
other designated cells
including itself**



**Individual detection cells are configured
into *detectors* by linking activation
pointers.**

an unbounded number of detector cells configured as finite state machines can extend indefinitely across multiple processors



All active cells have simultaneous access to current data-stream byte

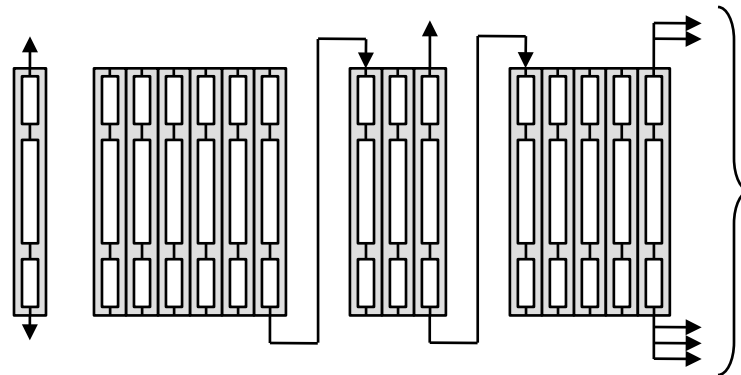
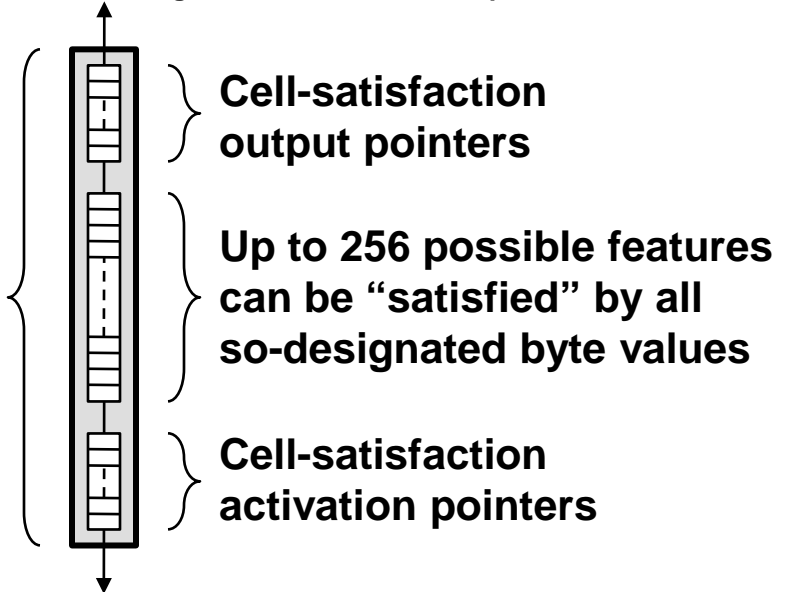
Reconfigurable Pattern Processor

Reusable Cells Reconfigurable in a Scalable Architecture

www.parshift.com/Files/PsiDocs/Pap090303-PatternRecognitionWithoutTradeoffs.pdf

Independent detection cell:
content addressable
by current input byte

If active, and satisfied with
current byte, can activate
other designated cells
including itself



Individual detection cells are configured
into *detectors* by linking activation
pointers.

Enables High Fidelity Modeling

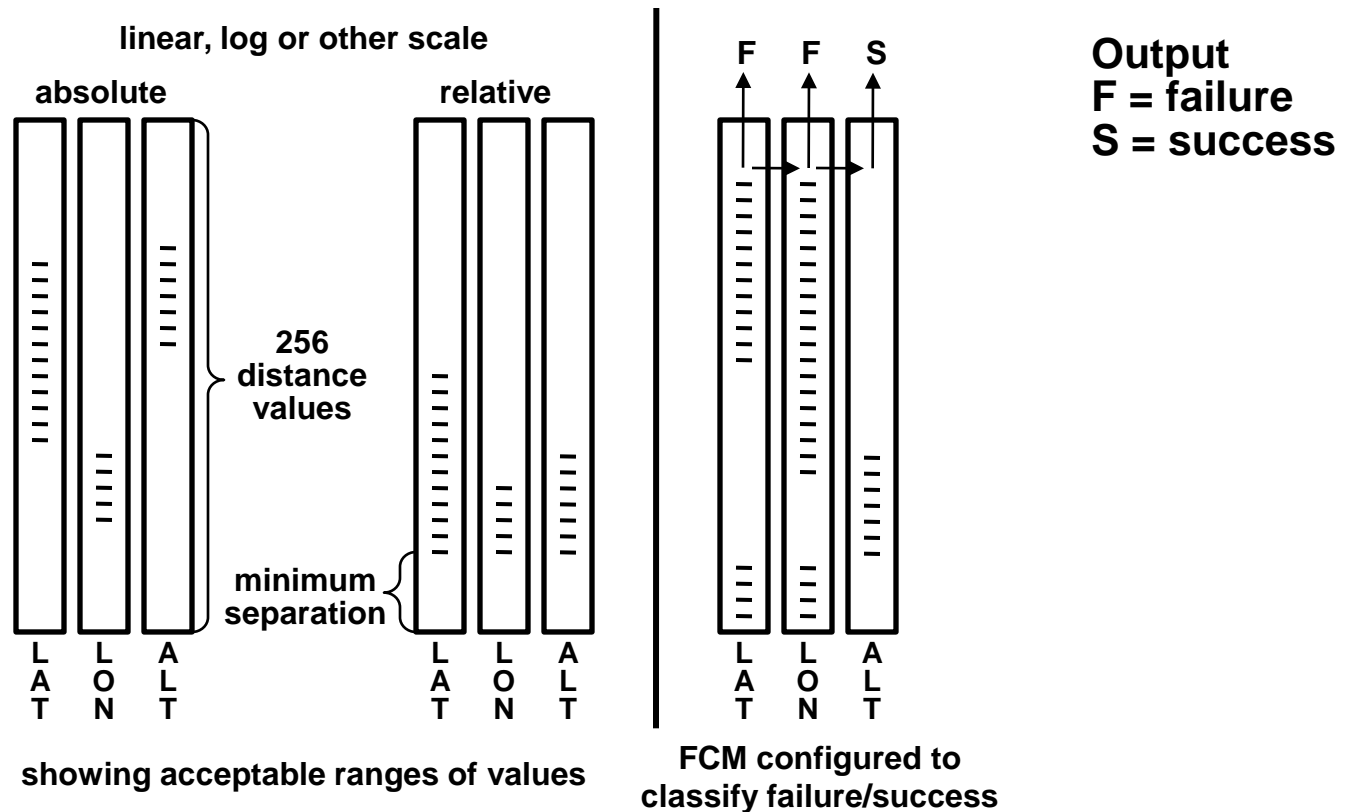
an unbounded number of detector cells configured as finite state machines can extend indefinitely across multiple processors



All active cells have simultaneous access to current data-stream byte

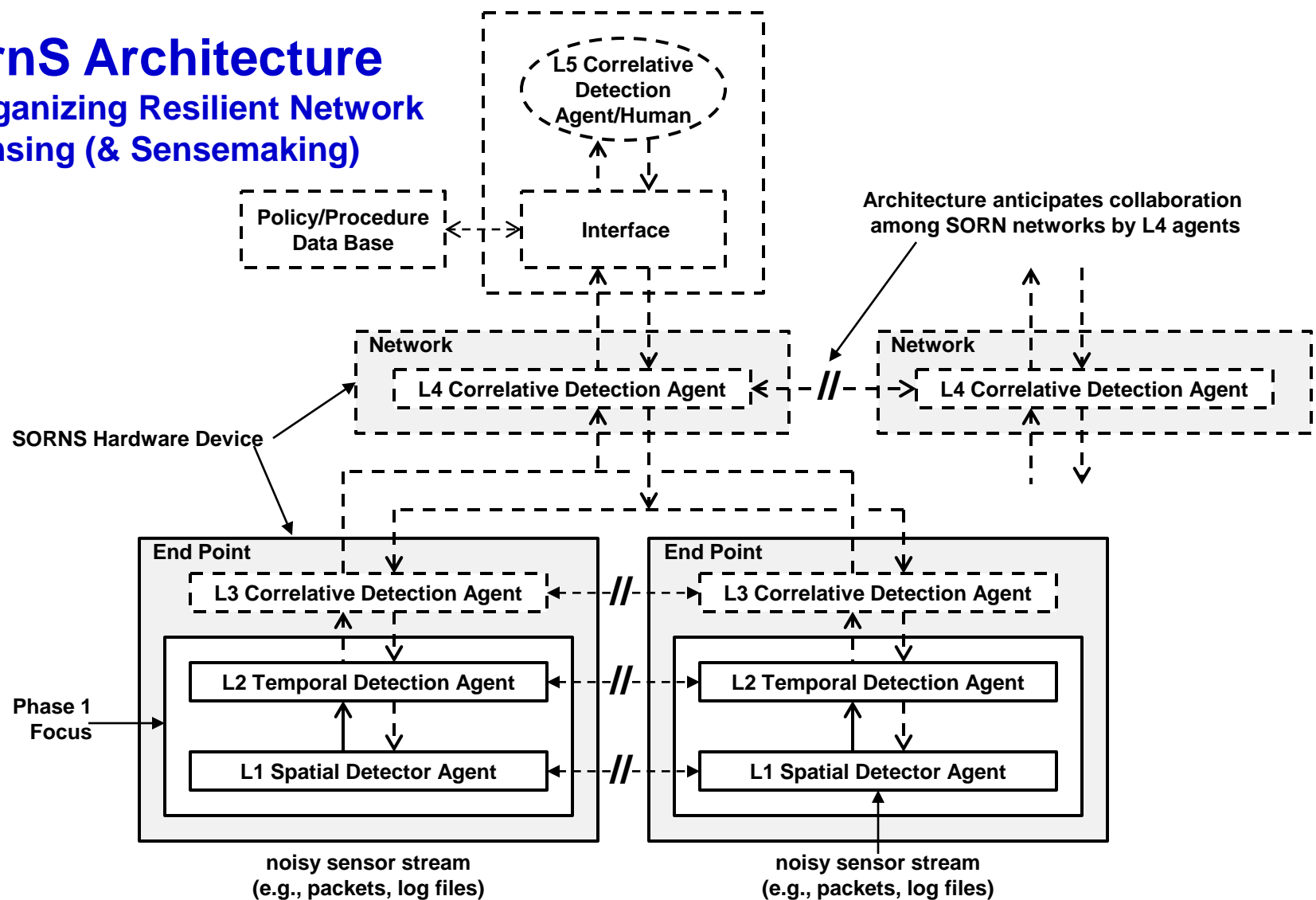
Value-Based Feature Example

A reference pattern example for behavior-verification of a mobile object.
Is it traveling within the planned space/time envelop?
Using GPS position data: Latitude, Longitude, Altitude.



Sorns Architecture

Self Organizing Resilient Network Sensing (& Sensemaking)



Multi-level hierarchy refines situational awareness with learning and sensemaking, supports remedial action agents (human/automated) with succinct relevant information.

Notes:

- For general collaborative hierarchy concept see (Haack 2009)
- For hierarchical feed-forward/backward pattern learning, prediction, and sense-making see (George 2009).
- For hierarchical learning of causal patterns spread as time-sequence events see (Hawkins 2010, Hawkins et al 2010).

Endpoint Detector Families – Application Specific



Detection philosophy:

**Automated learning of pattern features
within a fixed set of generic pattern structures.**

- Correlative:** n specific things happened
- Temporal:** n specific things happened in order
- Spatial:** n specific things happened in contiguous order

Proof-of-Concept

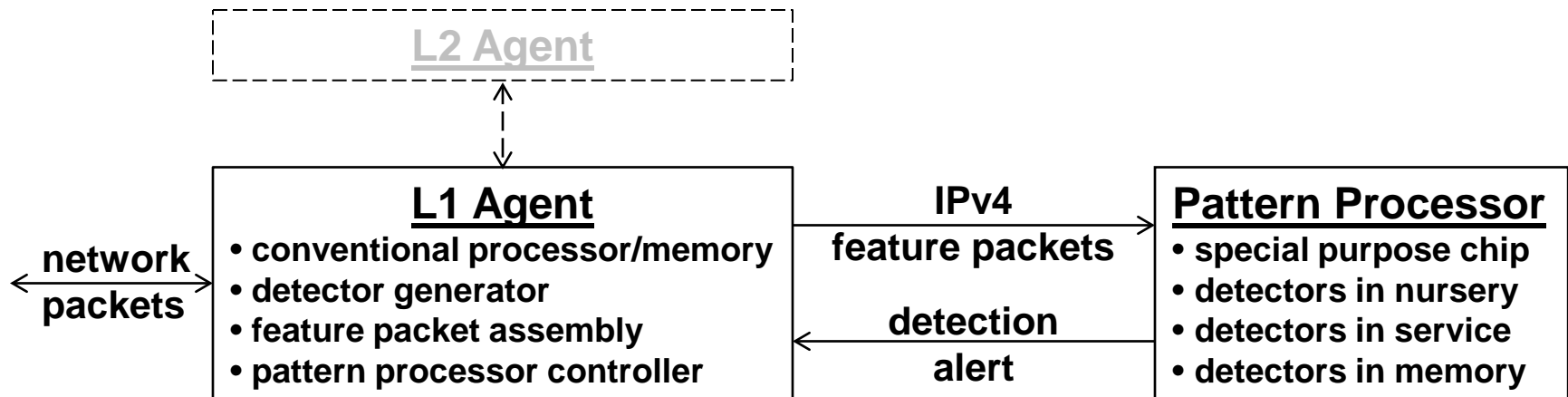
IPv4 packet-header detection

– single packet-header signature patterns (spatial connection category)

Three elements to a pattern signature: address – port – type

- **Address:** 4 bytes - Only the non-host address is of interest.
- **Port:** 2 bytes - Only the destination port is of interest.
- **Types:** 3 bits covers 8 types – (TCP, UDP, ICMP, other) x (incoming, outgoing)

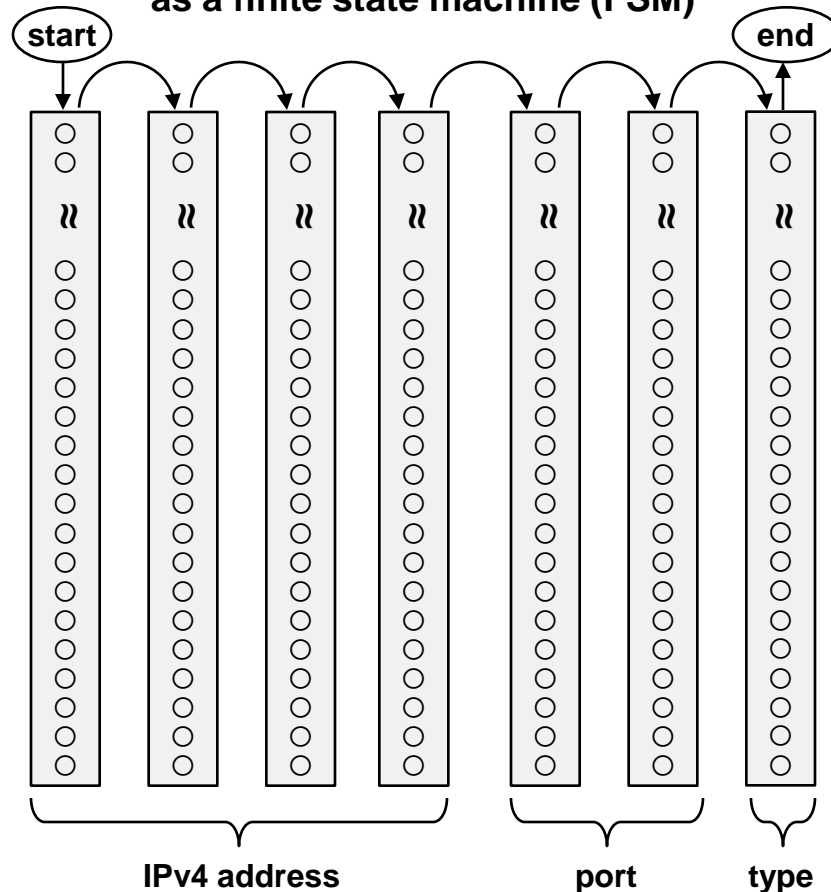
The L1-Agent preprocessor/controller selects relevant features from network packets and feeds them as condensed “feature packets” to the pattern processor



Feature Cells and Finite State Machines

(Illustrative example of pattern processor capability)

7 multi-feature detectors “connected”
as a finite state machine (FSM)



256-bit
associative
memory
multi-feature
detectors (MFD).

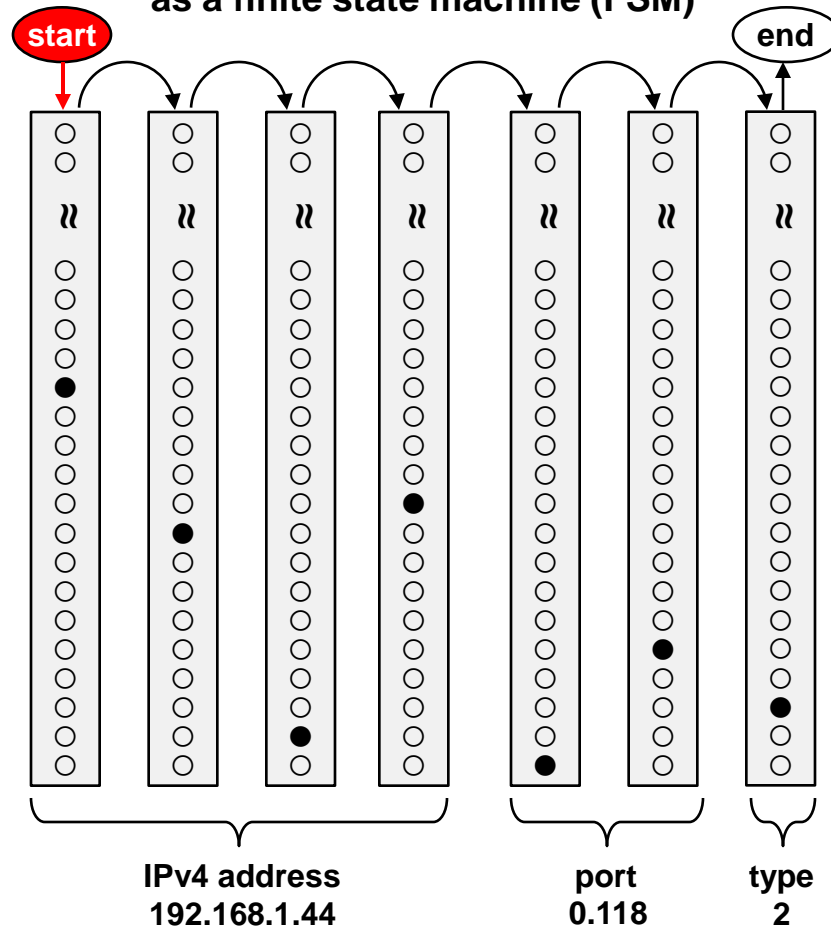
All active MFDs
are indexed by
the input
stream's current
byte value.

If the index finds
a set bit, the
next MFD is
activated and
looks at the next
stream byte,
else the process
dies.

Feature Cells and Finite State Machines

(Illustrative example of pattern processor capability)

7 multi-feature detectors “connected”
as a finite state machine (FSM)



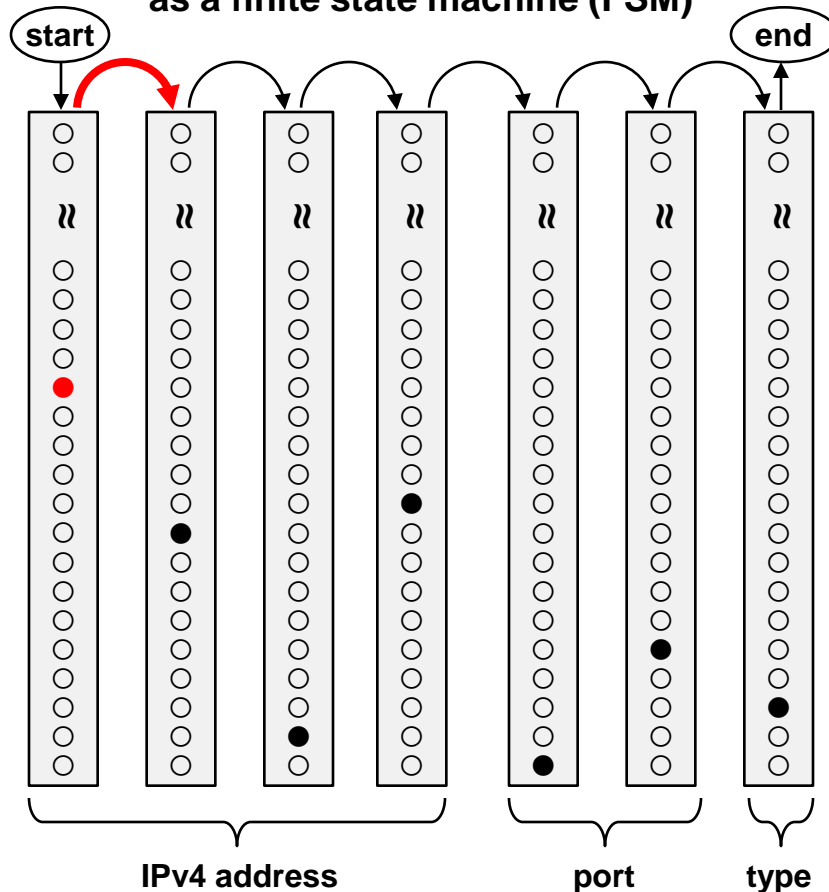
Loaded with 7 values

192.168.1.44, 0.118, 2

Feature Cells and Finite State Machines

(Illustrative example of pattern processor capability)

7 multi-feature detectors “connected”
as a finite state machine (FSM)



256-bit
associative
memory
multi-feature
detectors (MFD).

All active MFDs
are indexed by
the input
stream's current
byte value.

If the index finds
a set bit, the
next MFD is
activated and
looks at the next
stream byte,
else the process
dies.

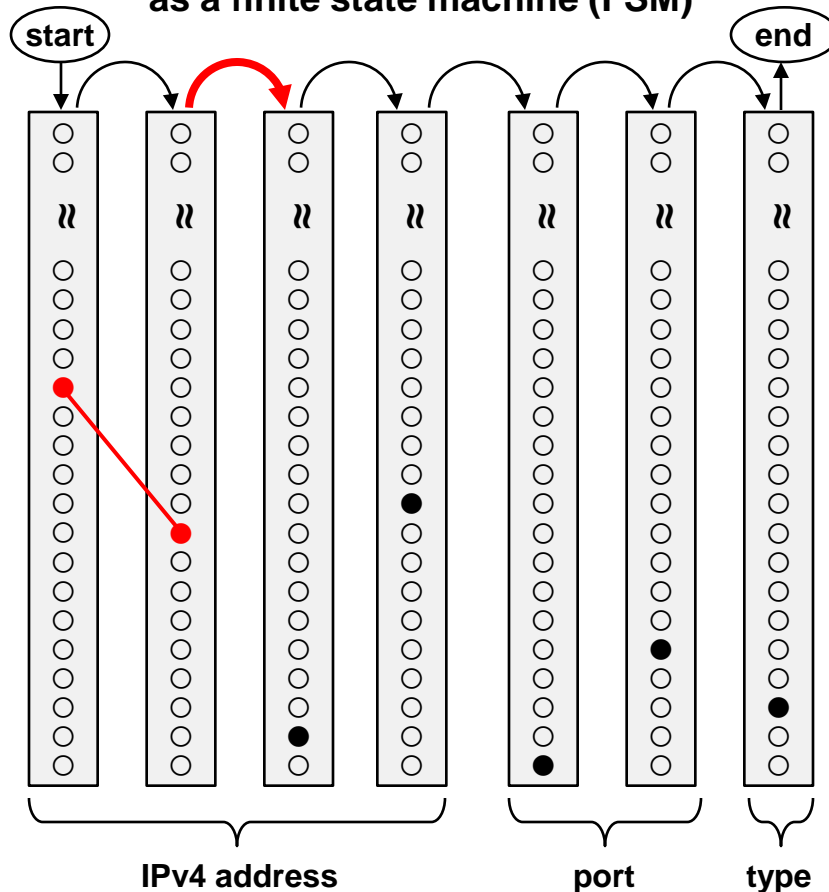
Processing Data Stream

192.168.1.44, 0.118, 2

Feature Cells and Finite State Machines

(Illustrative example of pattern processor capability)

7 multi-feature detectors “connected”
as a finite state machine (FSM)



256-bit
associative
memory
multi-feature
detectors (MFD).

All active MFDs
are indexed by
the input
stream's current
byte value.

If the index finds
a set bit, the
next MFD is
activated and
looks at the next
stream byte,
else the process
dies.

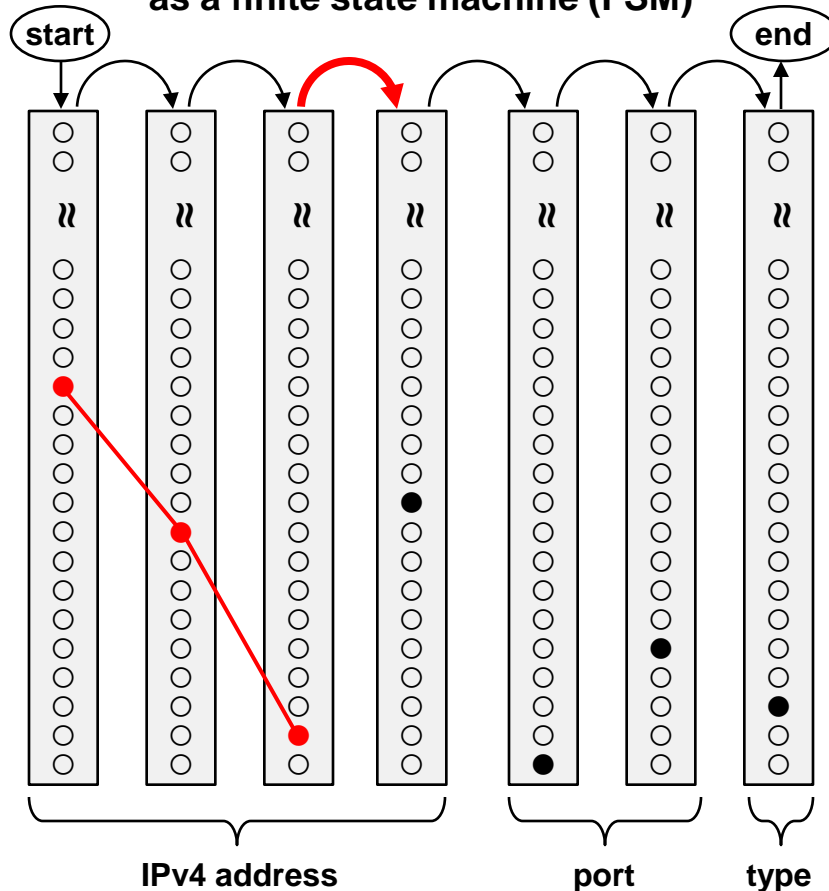
Processing Data Stream

192.168.1.44, 0.118, 2

Feature Cells and Finite State Machines

(Illustrative example of pattern processor capability)

7 multi-feature detectors “connected”
as a finite state machine (FSM)



256-bit
associative
memory
multi-feature
detectors (MFD).

All active MFDs
are indexed by
the input
stream's current
byte value.

If the index finds
a set bit, the
next MFD is
activated and
looks at the next
stream byte,
else the process
dies.

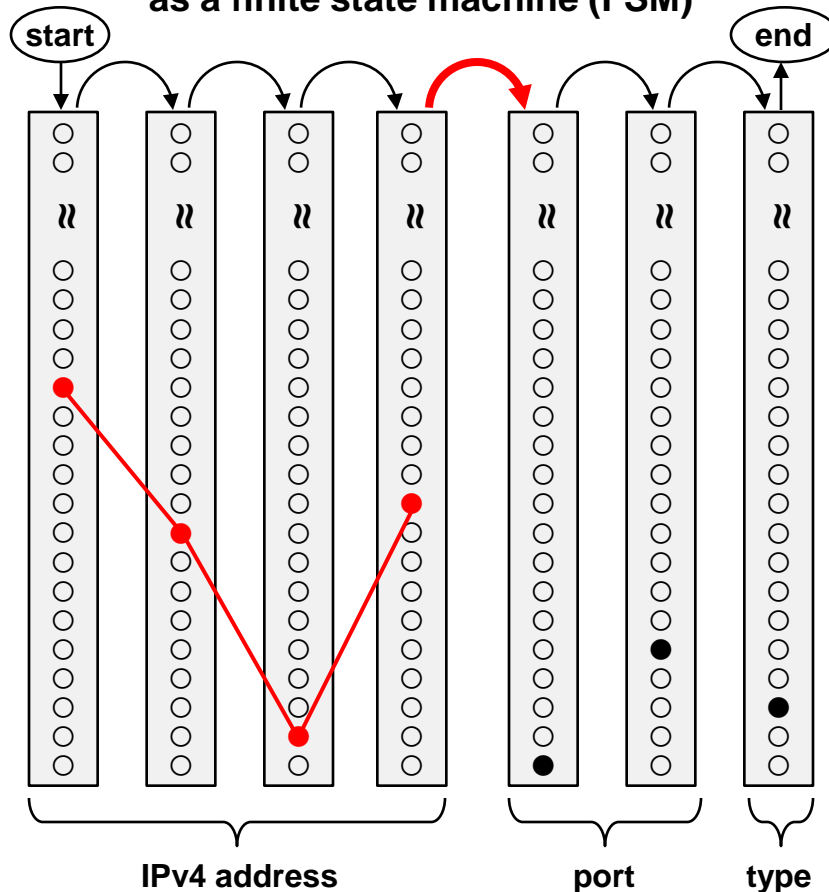
Processing Data Stream

192.168.1.44, 0.118, 2

Feature Cells and Finite State Machines

(Illustrative example of pattern processor capability)

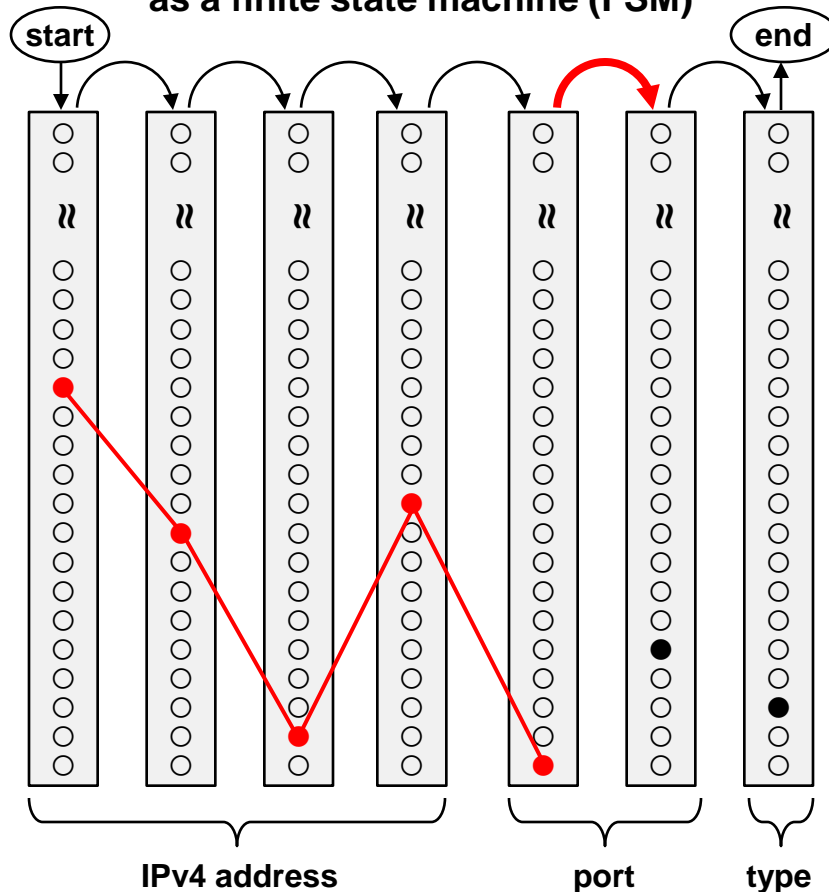
7 multi-feature detectors “connected”
as a finite state machine (FSM)



Feature Cells and Finite State Machines

(Illustrative example of pattern processor capability)

7 multi-feature detectors “connected”
as a finite state machine (FSM)



256-bit
associative
memory
multi-feature
detectors (MFD).

All active MFDs
are indexed by
the input
stream's current
byte value.

If the index finds
a set bit, the
next MFD is
activated and
looks at the next
stream byte,
else the process
dies.

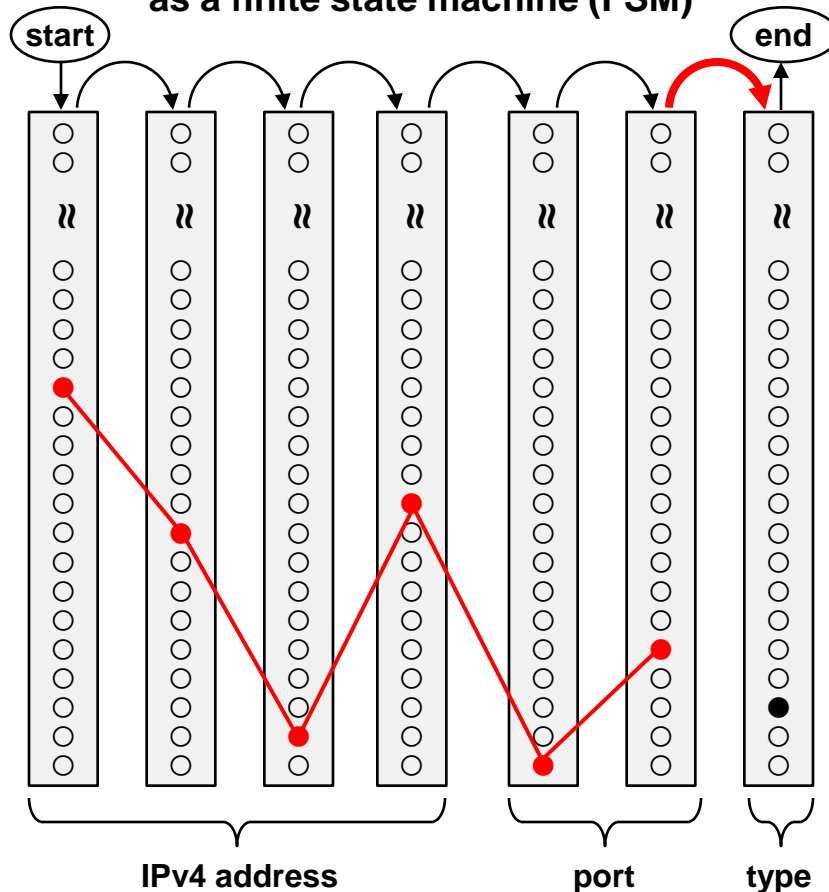
Processing Data Stream

192.168.1.44, 0.118, 2

Feature Cells and Finite State Machines

(Illustrative example of pattern processor capability)

7 multi-feature detectors "connected" as a finite state machine (FSM)



256-bit associative memory multi-feature detectors (MFD).

All active MFDs are indexed by the input stream's current byte value.

If the index finds a set bit, the next MFD is activated and looks at the next stream byte, else the process dies.

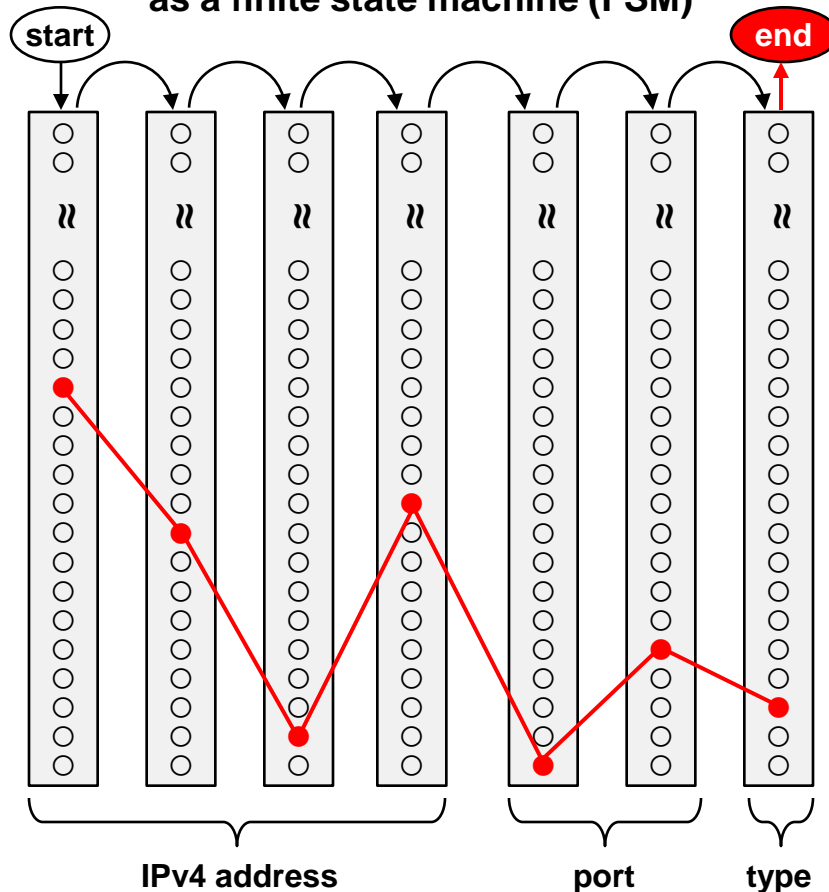
Processing Data Stream

192.168.1.44, 0.118, 2

Feature Cells and Finite State Machines

(Illustrative example of pattern processor capability)

7 multi-feature detectors “connected”
as a finite state machine (FSM)



256-bit
associative
memory
multi-feature
detectors (MFD).

All active MFDs
are indexed by
the input
stream's current
byte value.

If the index finds
a set bit, the
next MFD is
activated and
looks at the next
stream byte,
else the process
dies.

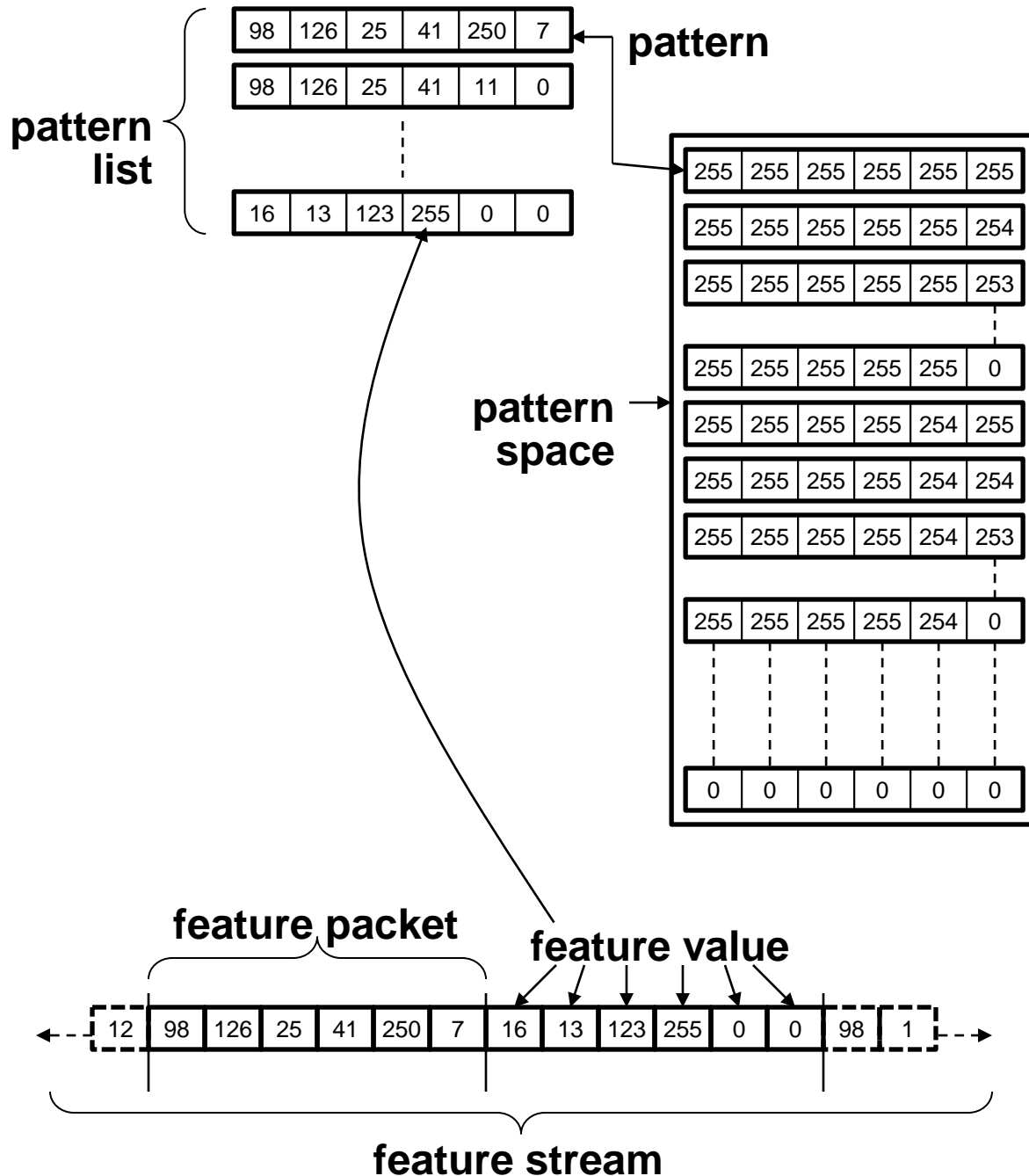
Processing Data Stream

192.168.1.44, 0.118, 2

Very Large Scale Anomaly Detector

Patent Pending

Fundamental Elements

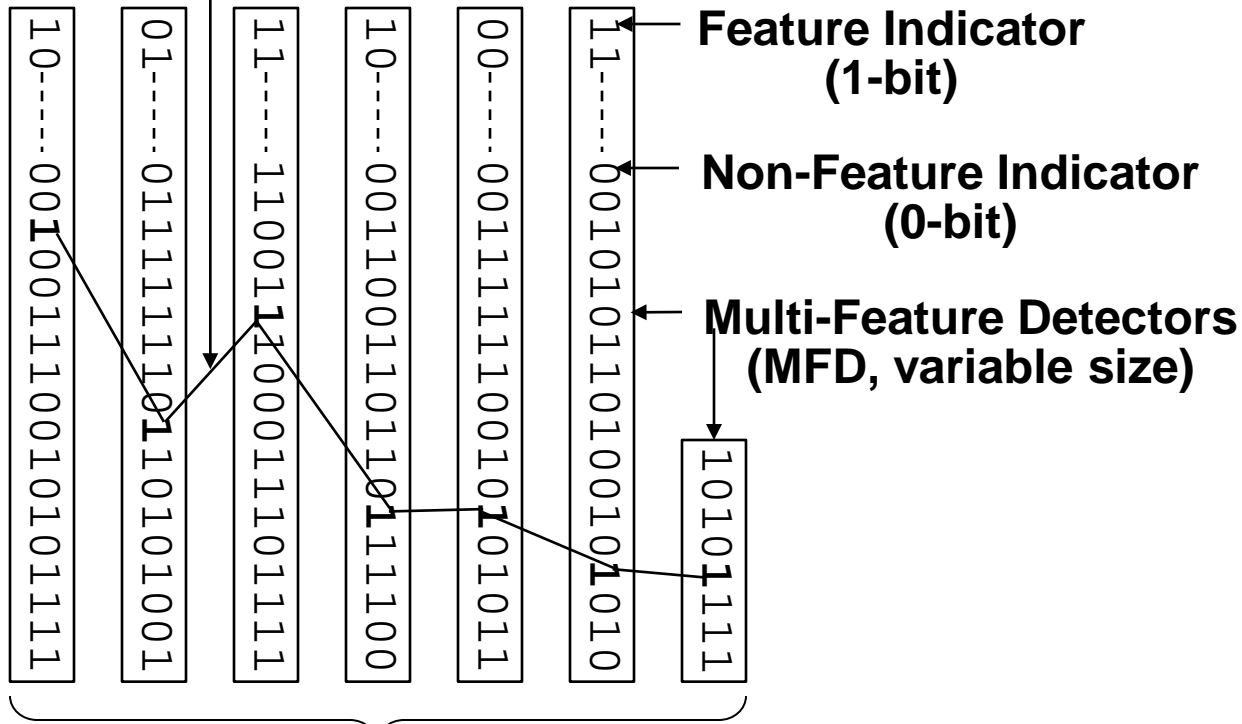


pattern space contains
 $256^6 = 2.81 \times 10^{14}$
unique patterns

for patterns consisting
of 6 feature values
with range 0-255

Gang Detector (GD)

Pattern (or Pattern Path)



Gang Detector

(eg, with seven multi-feature detectors)

Gang Detector (GD)

A GD is implemented as a 2 dimensional bit array, with each column corresponding to an MFD of independent size, but typically a max of 256 to accommodate associative addressing (indexing) by an 8-bit Feature Packet byte.

A Feature Indicator is a 1-bit in any or all of the possible index values.

One GD with all Feature Indicators present would have $256 \times 256 \times 256 \times 256 \times 256 \times 256 \times 8 =$

2.6×10^{15} unique Pattern Paths.

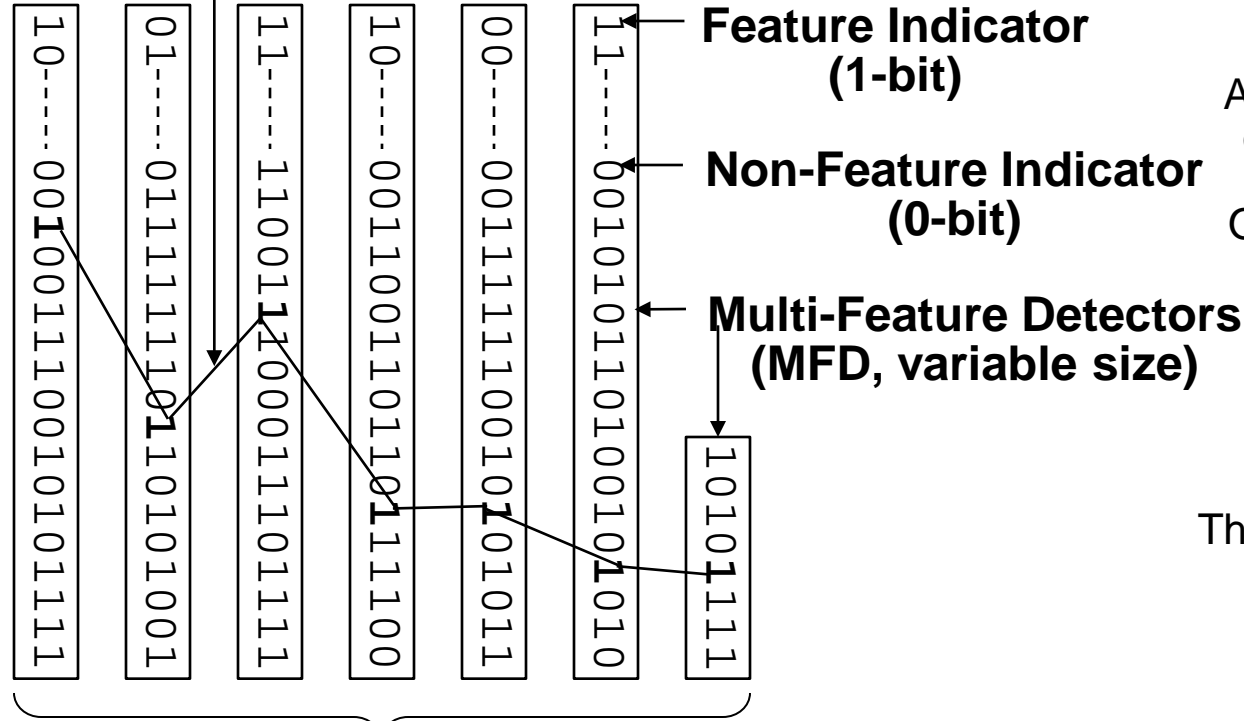
This many unique patterns would be represented in just $(6 \times 32) + 1 =$

193 8-bit data bytes.

If each of these patterns were in a pattern list, seven times the number of possible patterns in data bytes would be required =

$\sim 10^{16}$ data bytes in contrast.

Pattern (or Pattern Path)



Gang Detector (GD)

A GD is implemented as a 2 dimensional bit array, with each column corresponding to an MFD of independent size, but typically a max of 256 to accommodate associative addressing (indexing) by an 8-bit Feature Packet byte.

A Feature Indicator is a 1-bit in any or all of the possible index values.

One GD with all Feature Indicators present would have $256 \times 256 \times 256 \times 256 \times 256 \times 256 \times 8 =$

2.6×10^{15} unique Pattern Paths.

This many unique patterns would be represented in just $(6 \times 32) + 1 =$

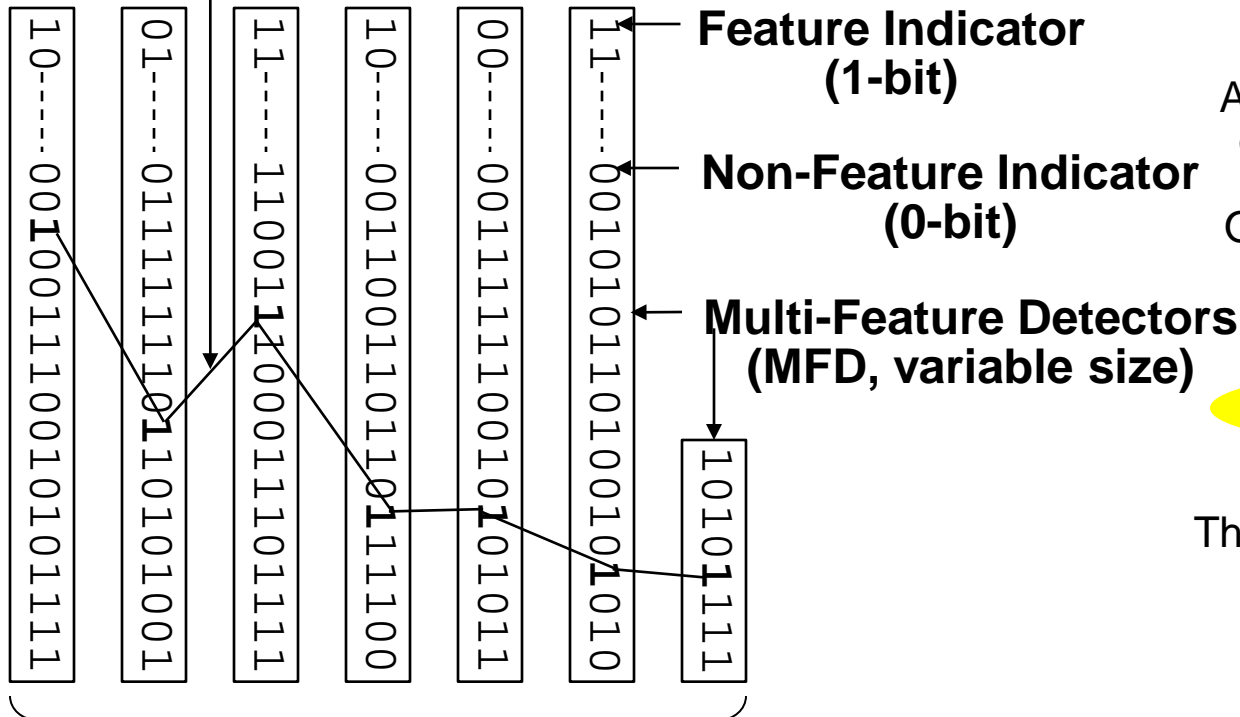
193 8-bit data bytes.

If each of these patterns were in a pattern list, seven times the number of possible patterns in data bytes would be required =

$\sim 10^{16}$ data bytes in contrast.

a unique benefit of the approach

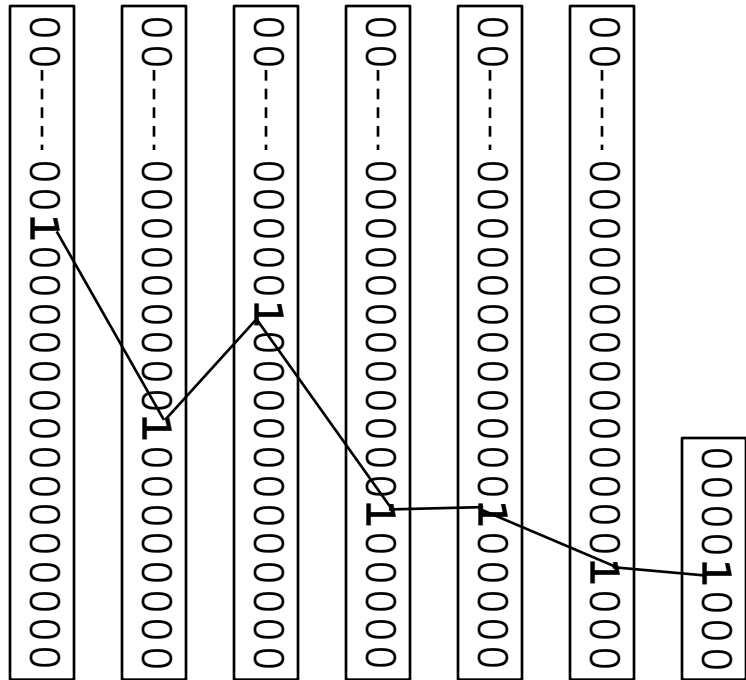
Pattern (or Pattern Path)



Gang Detector

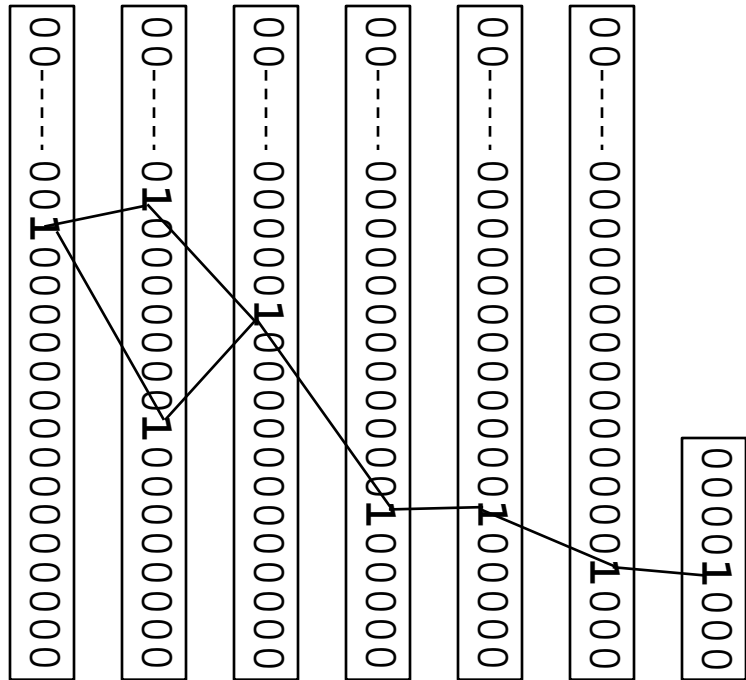
(eg, with seven multi-feature detectors)

Gang Detector (GD)



7 Feature Indicators = 1 Pattern (Path)

Gang Detector (GD)



7 Feature Indicators = 1 Pattern (Path)
8 Feature Indicators = 2 Patterns (Paths)

Gang Detector (GD)

Adding a single Feature Indicator increases the Patterns (Paths) by a factor of 2, an exponential increase.

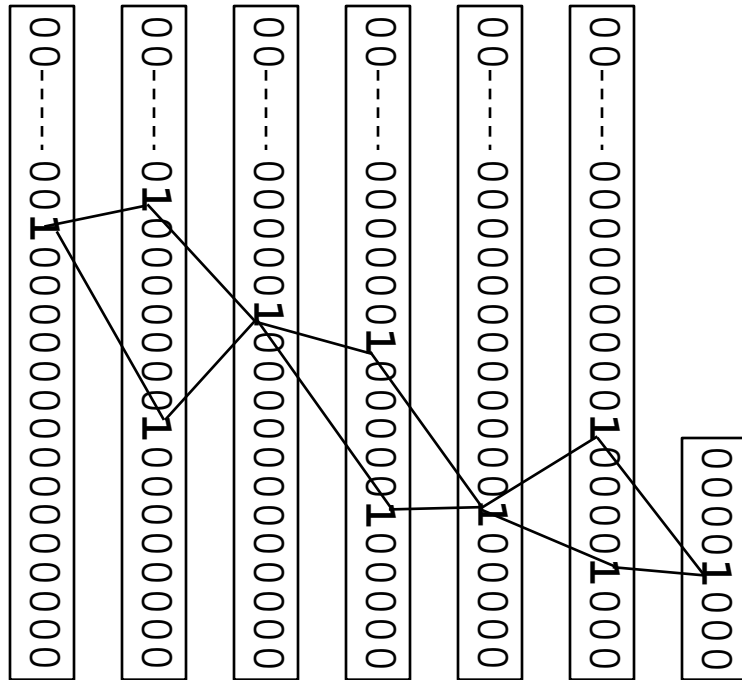
An application might create a new GD with the same percentage of Feature Indicators in every MFD.

If that were 50%, with six MFDs of size 256 and one of size 8, the total number of Patterns (Paths) upon creation would be

$$128 \times 128 \times 128 \times 128 \times 128 \times 128 \times 4 =$$

1.8×10^{13} patterns

Detectable at data stream feed speed independent of the number of patterns



- 7 Feature Indicators = 1 Pattern (Path)
- 8 Feature Indicators = 2 Patterns (Paths)
- 9 Feature Indicators = 4 Patterns (Paths)
- 10 Feature Indicators = 8 Patterns (Paths)

Gang Detector (GD)

Adding a single Feature Indicator increases the Patterns (Paths) by a factor of 2, an exponential increase.

An application might create a new GD with the same percentage of random Feature Indicators in every MFD.

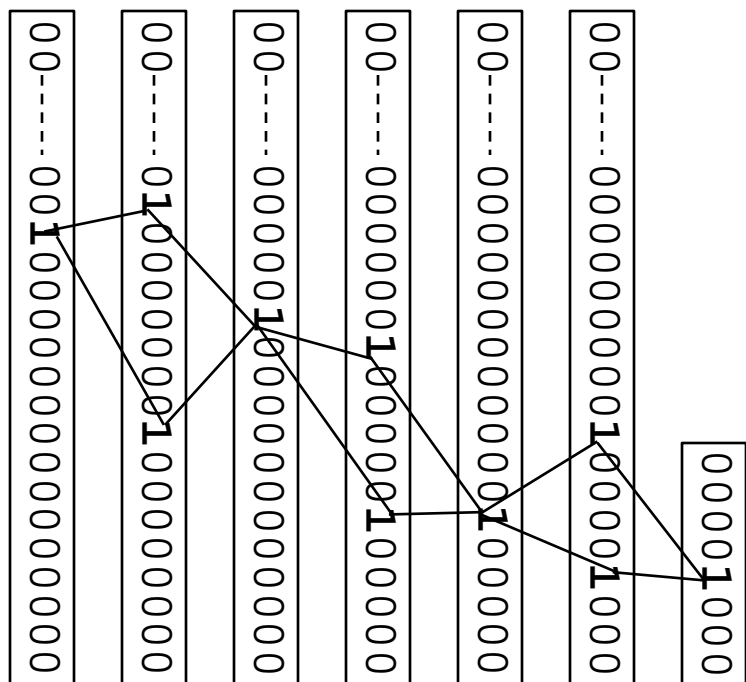
If that were 50%, with six MFDs of size 256 and one of size 8, the total number of Patterns (Paths) upon creation would be

$$128 \times 128 \times 128 \times 128 \times 128 \times 128 \times 4 =$$

1.8×10^{13} patterns

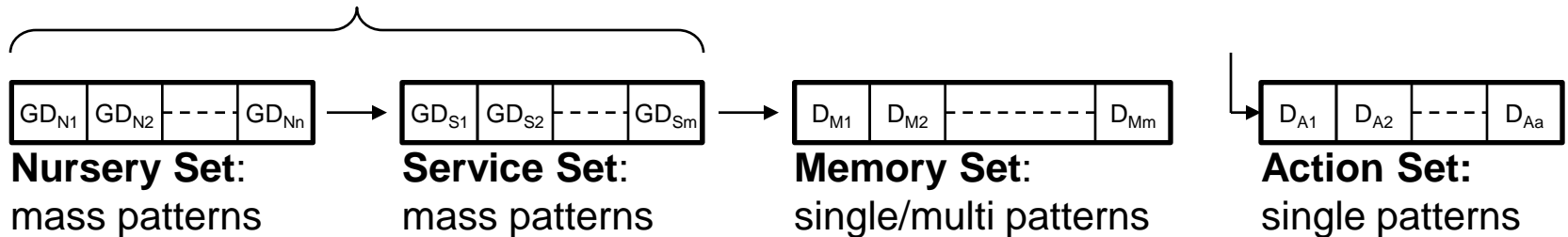
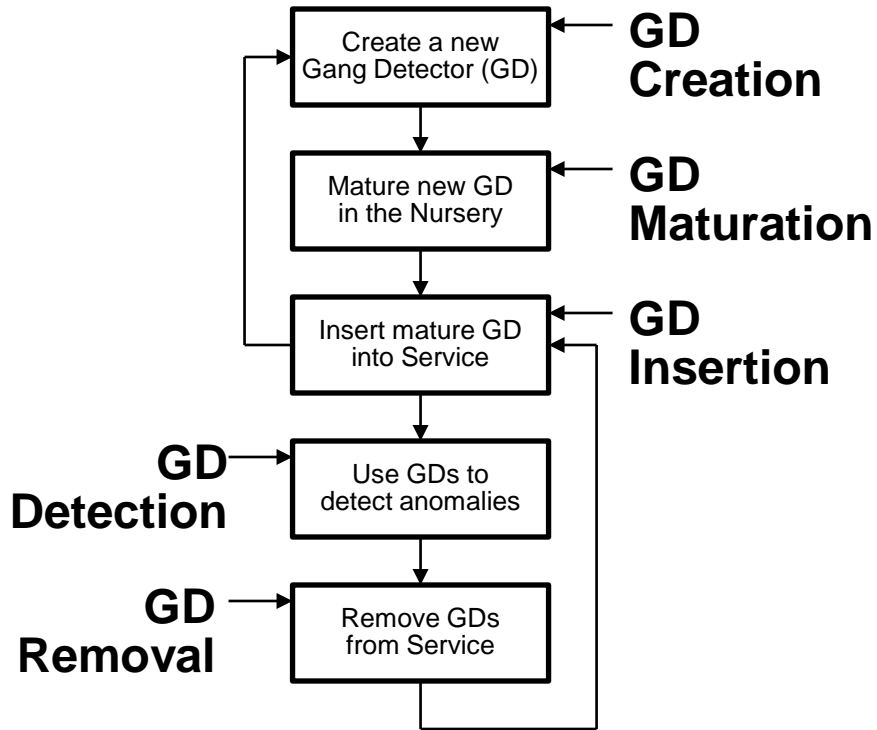
Detectable at data-stream feed-speed independent of the number of patterns

a unique benefit of the approach

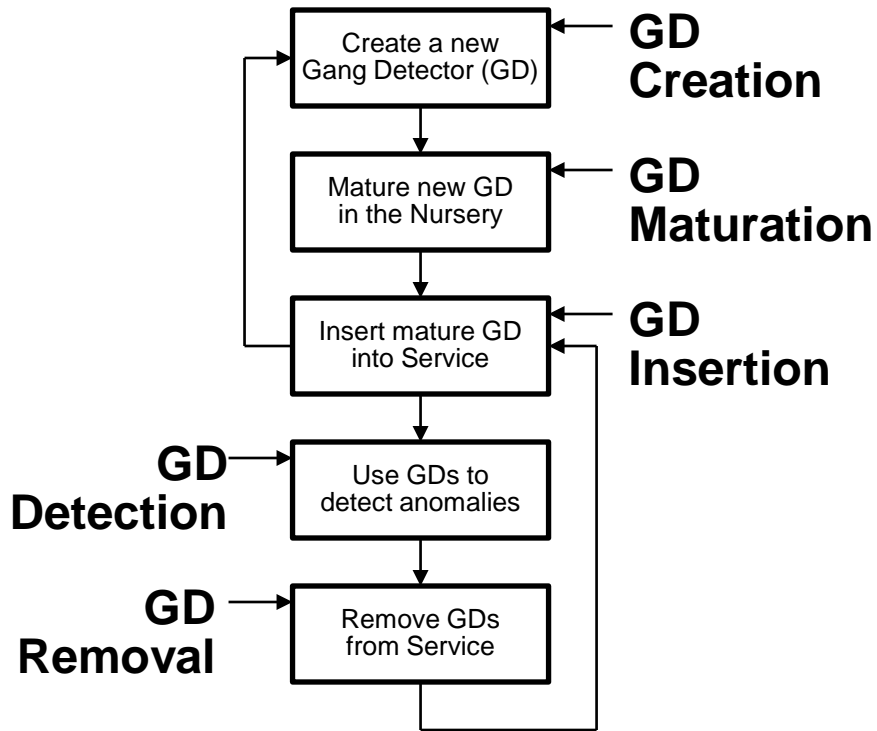


- 7 Feature Indicators = 1 Pattern (Path)
- 8 Feature Indicators = 2 Patterns (Paths)
- 9 Feature Indicators = 4 Patterns (Paths)
- 10 Feature Indicators = 8 Patterns (Paths)

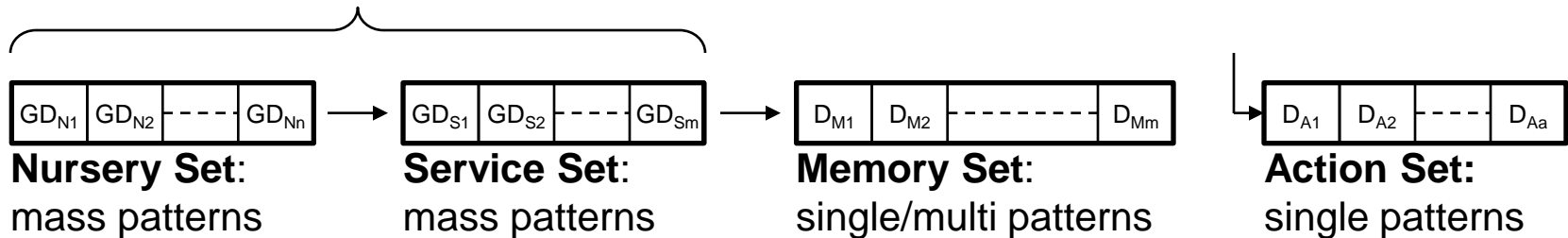
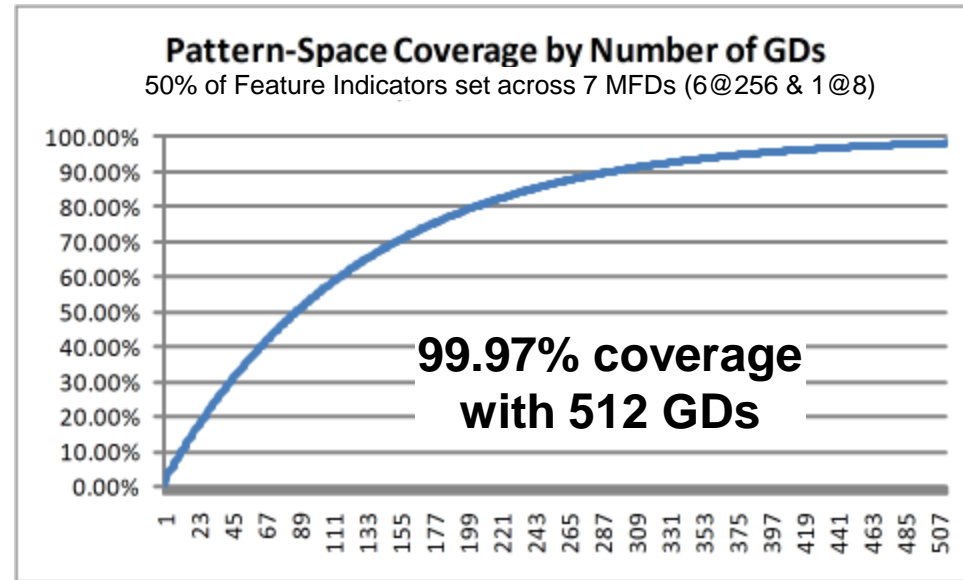
Detector Sets



Detector Sets

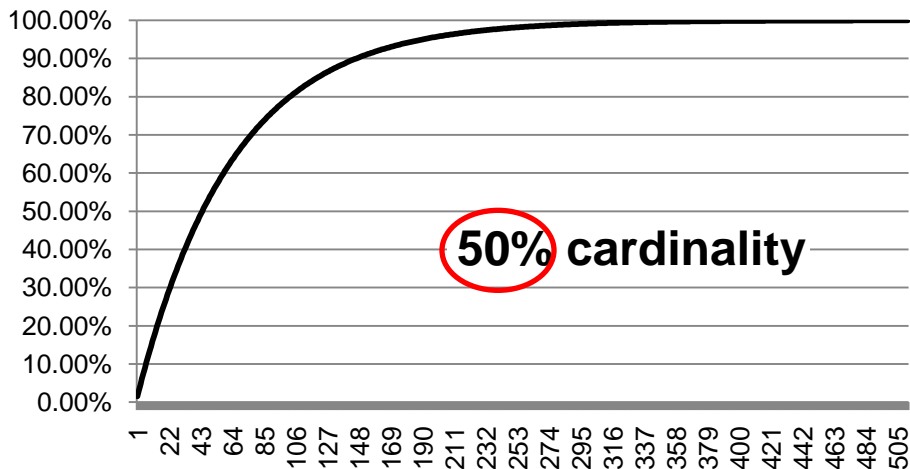


Multiple gang detectors covering slightly-overlapping portions of total pattern space collectively increase the total coverage of pattern space.

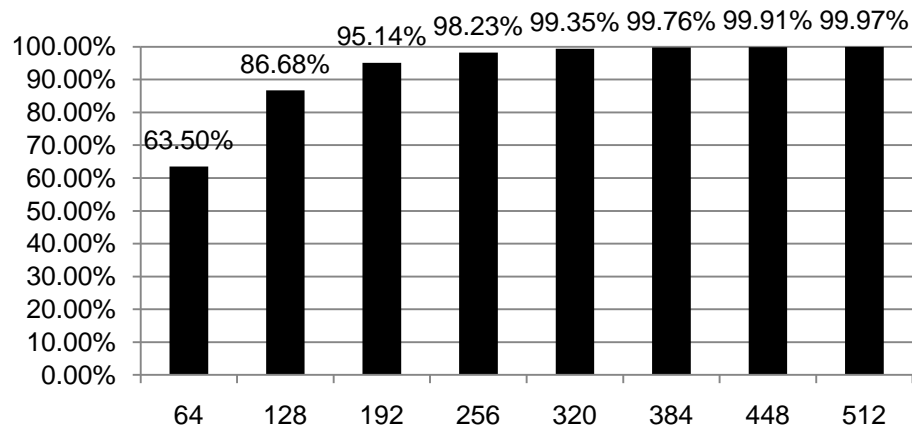


IPv4 Pattern-Space Coverage: **c=6**

Pattern-Space Coverage by Number of GDs

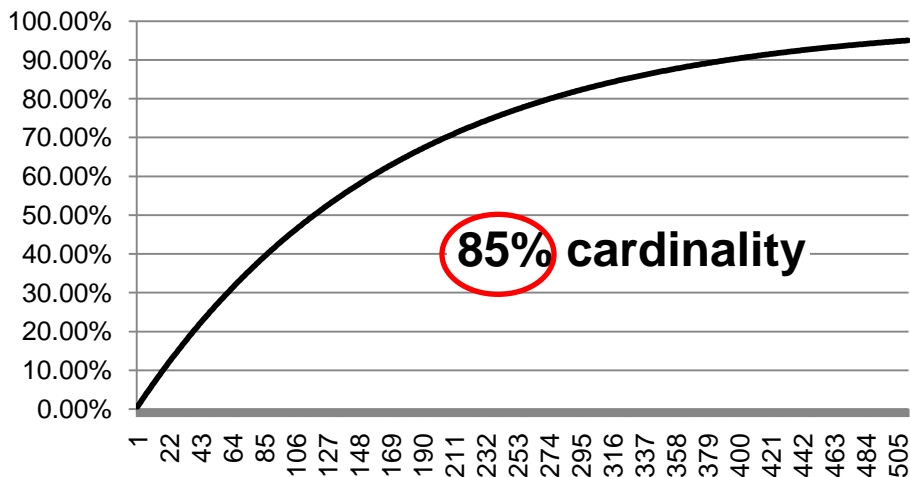


% Coverage by number of GDs

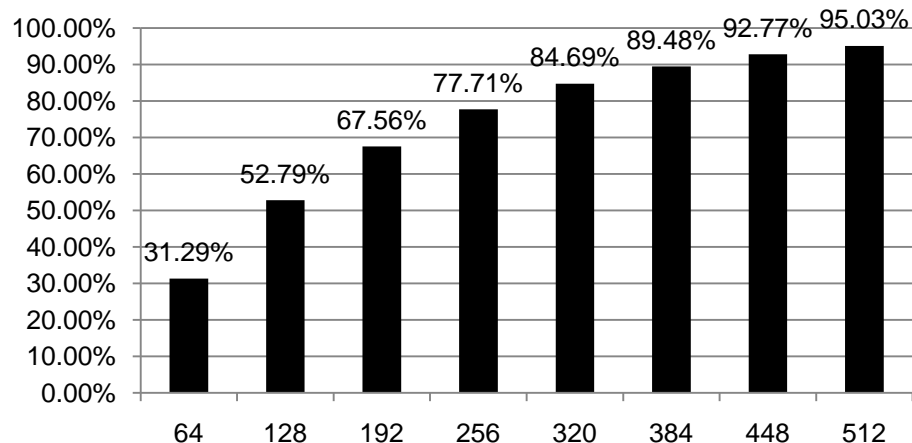


IPv6 Pattern-Space Coverage: **c=32**

Pattern-Space Coverage by Number of GDs



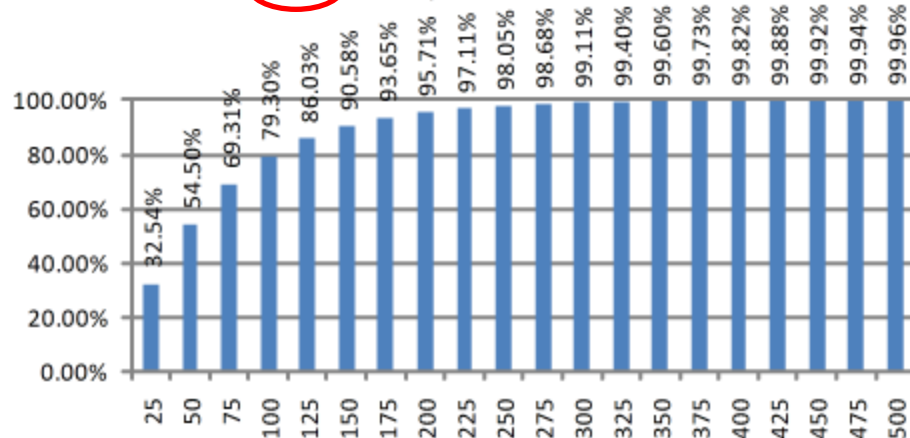
% Coverage by number of GDs



Coverage as Function of Cardinality

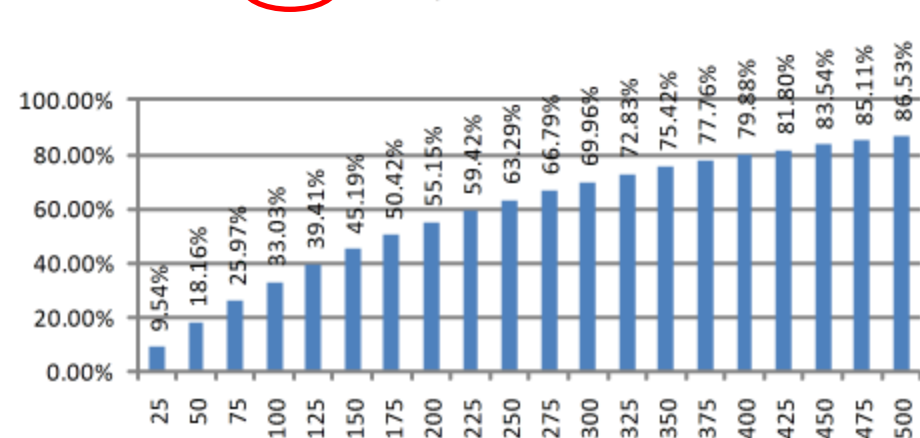
% Coverage by number of GDs

50% cardinality, n= 256 r= 128 c= 6



% Coverage by number of GDs

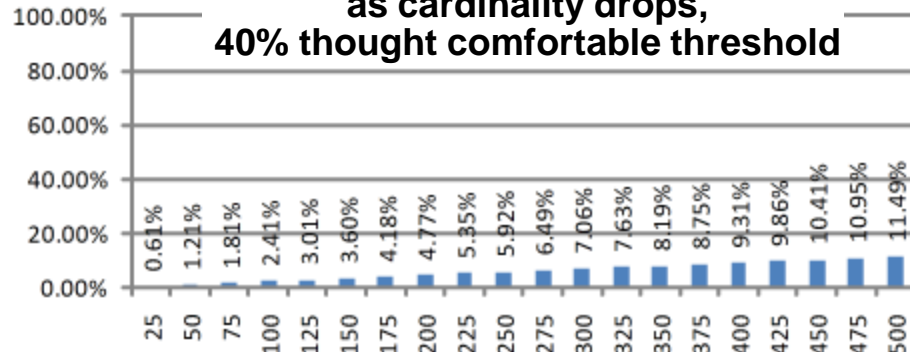
40% cardinality, n= 256 r= 102 c= 6



% Coverage by number of GDs

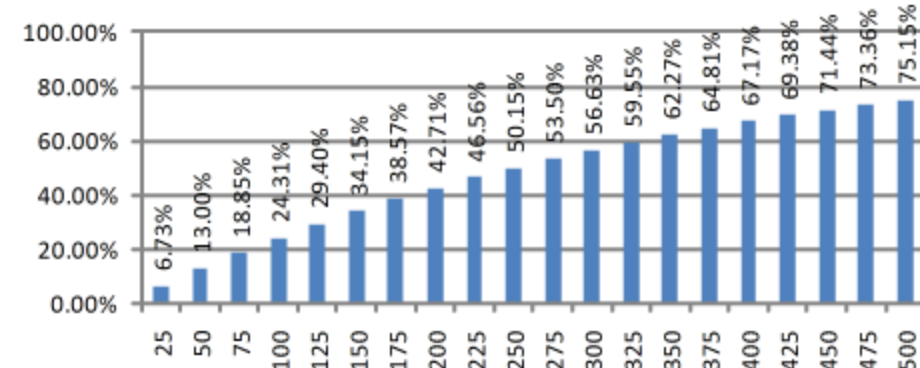
25% cardinality, n= 256 r= 64 c= 6

accelerating decline in coverage
as cardinality drops,
40% thought comfortable threshold



% Coverage by number of GDs

37.5% cardinality, n= 256 r= 96 c= 6



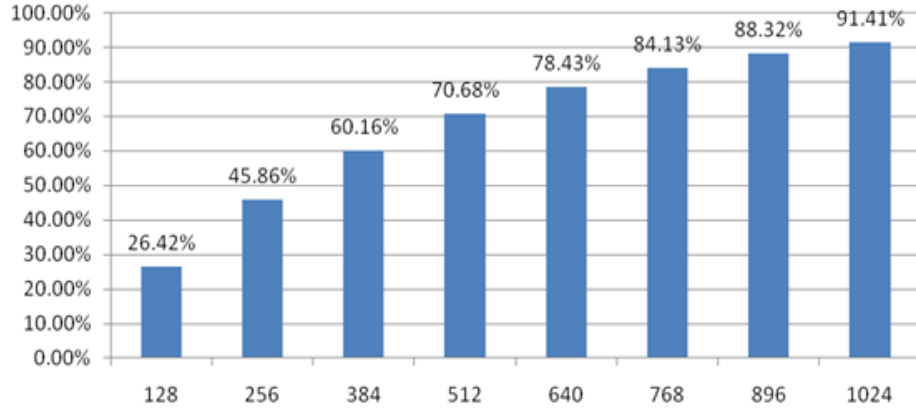
Cardinality losses justify the value of refresh-cycling the in-service GDs, and sharing results with other endpoint agents

Coverage of 32 MFDs Declines Fast

6 MFDs at 40% = 98.35% at 1024 GDs

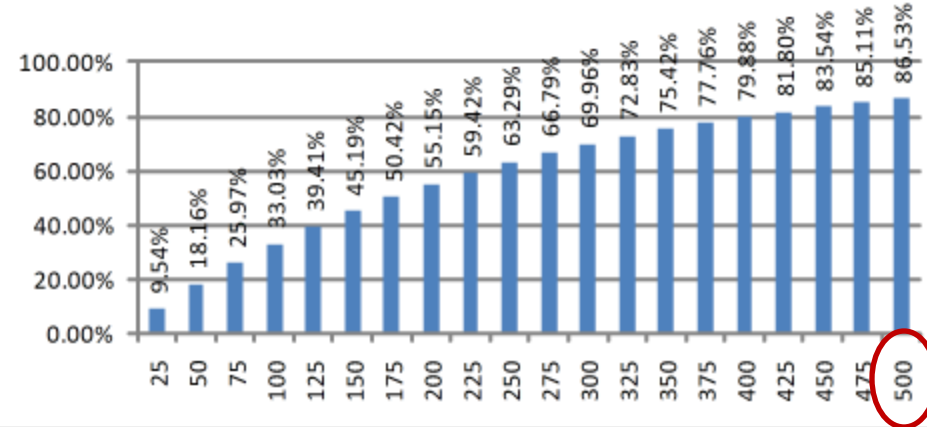
% Coverage by number of GDs

83% cardinality, n= 256 r= 212 c= 32



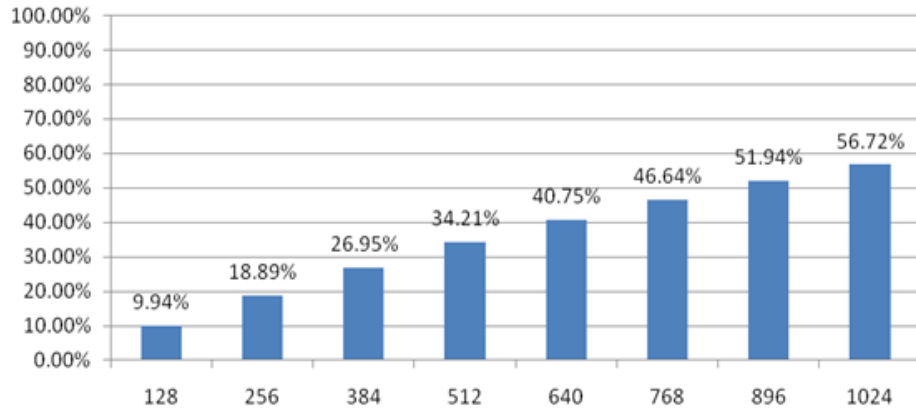
% Coverage by number of GDs

40% cardinality, n= 256 r= 102 c= 6



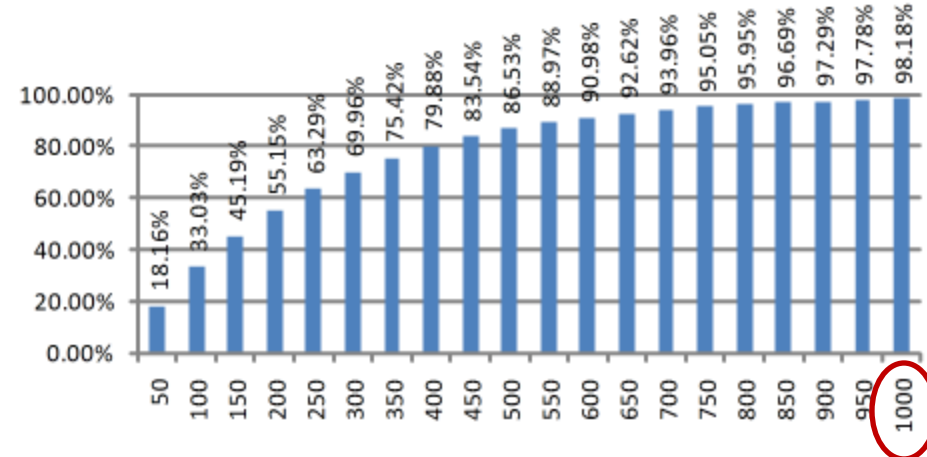
% Coverage by number of GDs

80% cardinality, n= 256 r= 205 c= 32



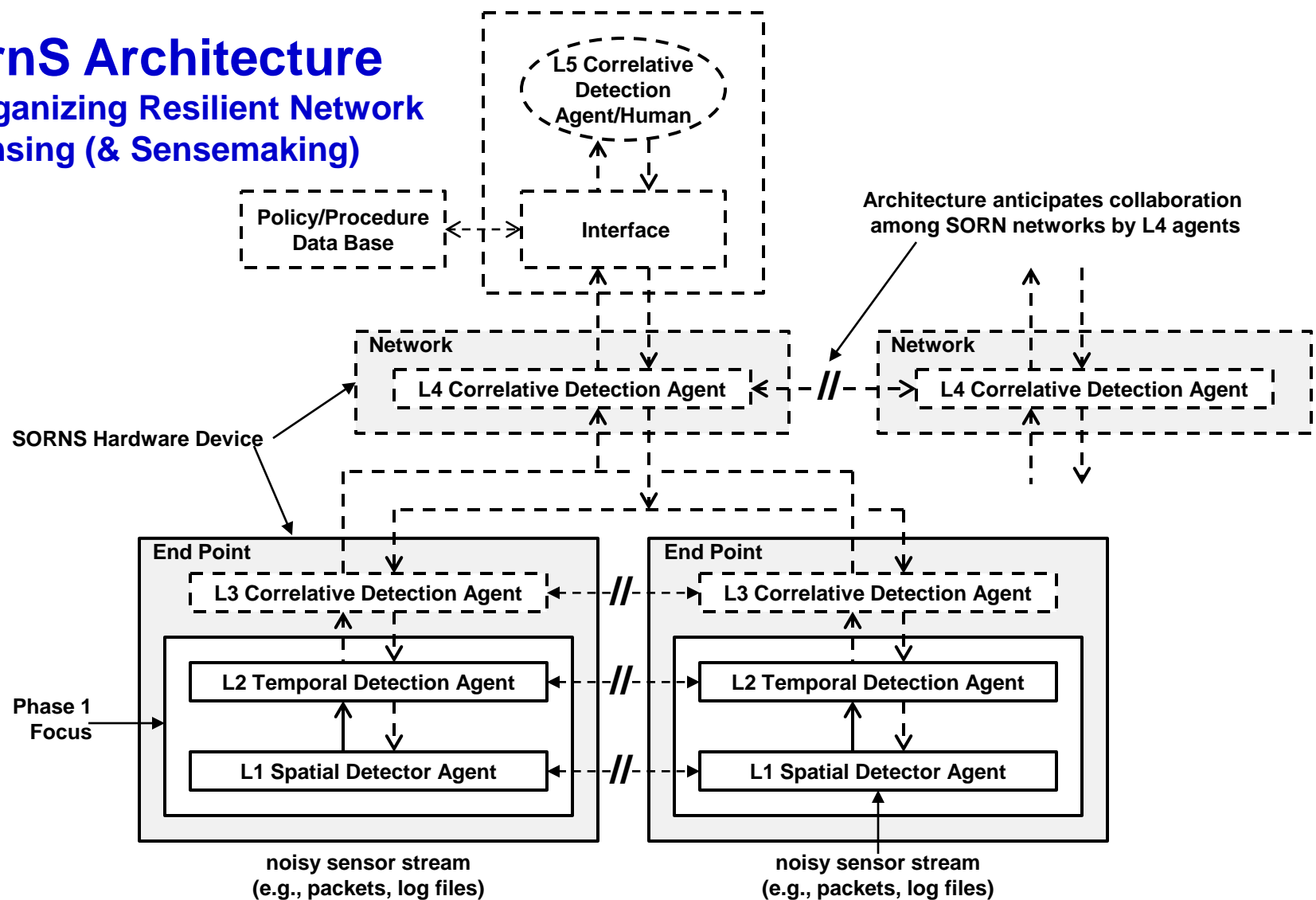
% Coverage by number of GDs

40% cardinality, n= 256 r=102 c= 6



Sorns Architecture

Self Organizing Resilient Network Sensing (& Sensemaking)



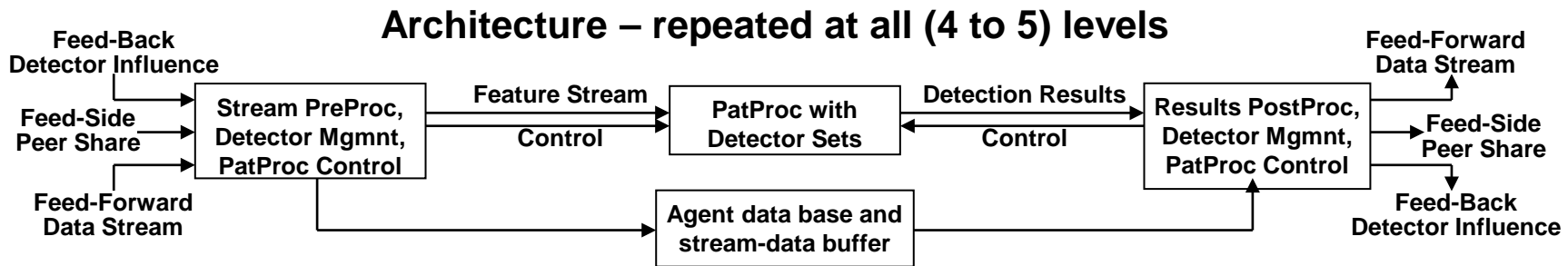
Multi-level hierarchy refines situational awareness with learning and sensemaking, supports remedial action agents (human/automated) with succinct relevant information.

Notes:

- For general collaborative hierarchy concept see (Haack 2009)
- For hierarchical feed-forward/backward pattern learning, prediction, and sense-making see (George 2009).
- For hierarchical learning of causal patterns spread as time-sequence events see (Hawkins 2010, Hawkins et al 2010).

General Architecture

Architecture blends immune system model with cortical hierarchy model.



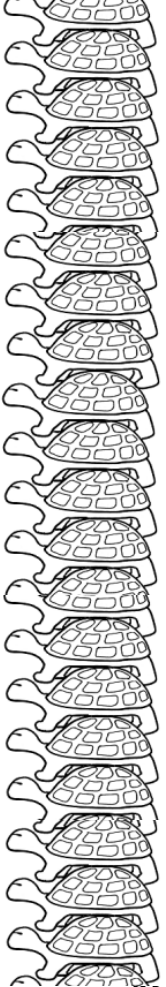
situational environment



immediate recognition



it's turtles all the way down



Immune system learning occurs by:

- augmenting the innate immune system population of detectors (e.g., black/white lists),
- negative selection self-tolerance training (e.g., nursery set),
- experience encounters (e.g., service set), and
- costimulation (e.g., memory-set).

Very Large Scale Anomaly Detector – Other App Domains

The SORNS example is a specific domain instance of the more general “normal vs. anomalous behavior” classification problem.

Of interest elsewhere, for instance:

- monitoring the operational behaviors of a swarm of unmanned autonomous weapons sent on a war fighting mission,
- video image monitoring of human behavior in terrorist target areas like airports,
- insider threat behavior,
- monitoring machine and process behaviors in factories (anti-Stuxnet),
- monitoring critical infrastructure operating behaviors, say for financial-market transactions,
- fraud, data mining, intelligence,

Perhaps most interesting,
the human brain appears to work on anomaly detection

- it appears to select and store (learn) pattern features for uniqueness and rarity
- children learn with attention to things that are different and new
- it doesn't learn immediately, but does recognize learned patterns immediately
- it can learn both by download (taught) and by speculative discovery (try this)

Next Steps

- Simulation-convergence on optimal parameters**
 - Endpoint feed-side collaboration**
 - Co-stimulation for reducing false positives**
 - Policy/procedure feed-down for false positive reduction**
 - Human interface for reports, alerts, and false positive reduction**
 - Seeding and black/white immediate-action list**
 - Correlative and temporal detector experimentation and convergence**
 - Detector-set learning algorithms**
 - Hierarchical stabilization learning and feedback influence algorithms**
-
- Get partners & potential commercializers involved**
 - Build hardware prototype and test bed**
 - Operational test at TBD**

Very Large Scale Anomaly Detector - Summary

Gang Detector (seven-feature pattern example)

- **Memory breakthrough: 193 bytes vs. 10^{16} bytes for pattern storage**
- **Coverage breakthrough: 512 GDs covers 99.97% of pattern space at 1 endpoint**
- **Good coverage does not require high coverage at any endpoint:**
 - **Endpoint multiples boost total coverage with same coverage curve**
 - **Endpoint detector cycling boosts total coverage with same coverage curve**
- **Custom anomaly detection – no two installations alike (in pattern content)**
- **Dynamically self-adaptable to local situation**

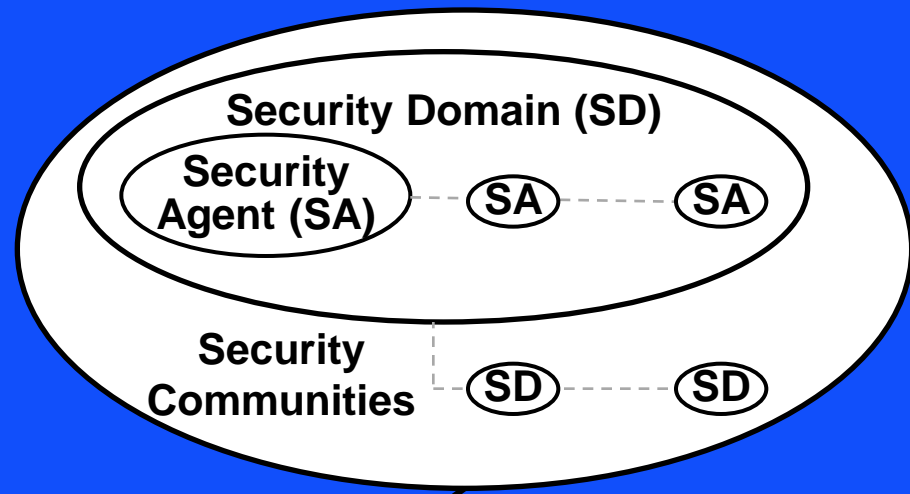
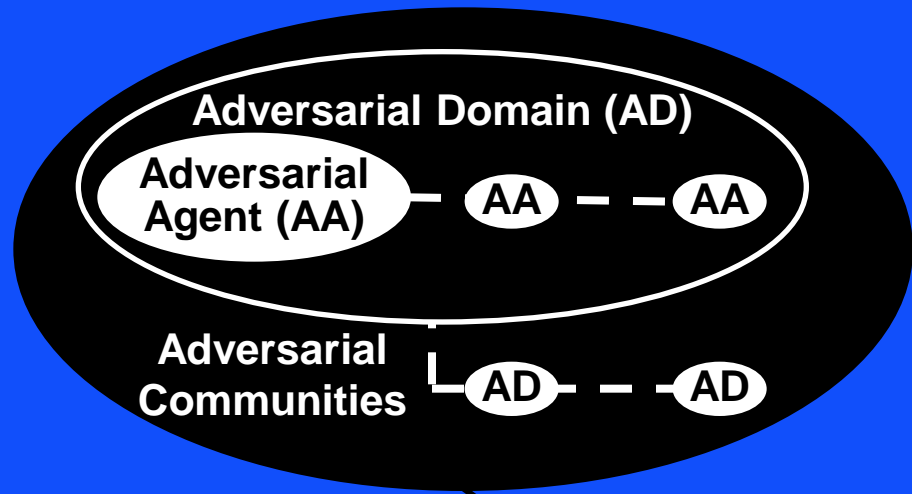
Pattern Processor without tradeoffs

- **Speed breakthrough: speed independent of number/size of patterns**
- **Capacity breakthrough: affordable massive pattern counts**

Potential Product Package

- **Transparent to endpoint: no latency or throttling of comm speed**
- **Transparent to network: no accommodations required**
- **Transparent to budget: no signature subscription**
- **Transparent to installer: bump on the wire installation**

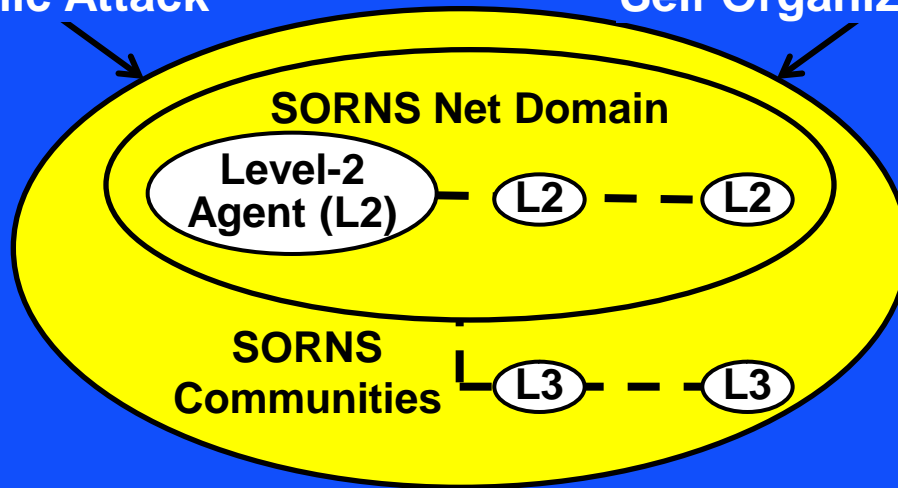
A Fairer Fight



Dynamic Attack

Self Organizing Systems

Dynamic attack includes human and systemic adaptive control – unable to know the detection detail and capability.



Anomaly Sensors collaborate within a network, dynamically adjusting to local situations, and can also collaborate among networks.

Just one self-organizing security example of many more to come

Participation Opportunities...

1. **Contribute a Self Organizing Security pattern (and/or Join INCOSE SSE WG)**
2. **Get Involved with SORNS Project:**
 - a) **Soft Red Team: project progress briefs**
 - b) **Consider being an Operational Test Site**
 - c) **Consider being a Commercialization Candidate**
3. **Be on the Pattern Processor Technology Intro Tour (Date TBD: capability/applications briefing)**

Contact: Rick Dove, dove@parshift.com

References

- Carlson, Jean and John Doyle. 2000. Highly Optimized Tolerance: Robustness and Design in Complex Systems, *Physical Review Letters* 84 (11): 2529–2532, 13 March.
- Carlson, Jean and John Doyle. 2002. Complexity and Robustness. *PNAS* 99: 2538–2545, 19 February.
- Csete, Marie and John Doyle. 2010. Bow Ties, Metabolism and Disease, *TRENDS in Biotechnology* 22(9), September 2004. www.cds.caltech.edu/~doyle/CmplxNets/Trends.pdf
- Dove, Rick. 2009. Pattern Recognition without Tradeoffs: Scalable Accuracy with No Impact on Speed. Proceedings of Cybersecurity Applications & Technology Conference For Homeland Security, Wash. D.C., March 2009. www.parshift.com/Files/PsiDocs/Pap090303-PatternRecognitionWithoutTradeoffs.pdf
- Dove, Rick. 2009. Embedding Agile Security in System Architecture. *Insight* 12 (2): 14-17. International Council on Systems Engineering, July. www.parshift.com/Files/PsiDocs/Pap090701Incese-EmbeddingAgileSecurityInSystemArchitecture.pdf
- Dove, Rick. 2010. Pattern Qualifications and Examples of Next-Generation Agile System-Security Strategies, IEEE International Carnahan Conference on Security Technology (ICCST), San Jose, CA, USA, 5-8 October. www.parshift.com/Files/PsiDocs/PatternQualificationsForAgileSecurity.pdf
- Dove, Rick. 2010. Self Organizing Resilient Network Sensing Using Patterns from Natural and Adversarial Processes, working paper, www.parshift.com/Files/PsiDocs/PatternsForResilientNetworkSensing&Sensemaking.pdf
- Dove, Rick. 2010. Illuminating Next Generation Agile Security Patterns. SERC Security Research Roadmap Workshop, March 31-April 1, Washington, D.C. www.parshift.com/Files/PsiDocs/Pap100331SERC-IlluminatingNextGenAgileSecurityPatterns.pdf
- Dove, Rick. 2010. Pattern Qualifications and Examples of Next-Generation Agile System Security Strategies. IEEE International Carnahan Conference on Security Technology (ICCST), San Jose, CA, USA, 5-8 Oct. www.parshift.com/Files/PsiDocs/PatternQualificationsForAgileSecurity.pdf
- Dove, Rick 2011. Patterns of Self-Organizing Agile Security for Resilient Network Situational Awareness and Sense-Making. Proceedings ITNG 2011 conference, Las Vegas, NV April 11-13. www.parshift.com/s/110411PatternsForSORNS.pdf
- Forrest, S., Perelson, A. S., Allen, L., and Cherukuri, R., Self-Nonself Discrimination in a Computer, In Proceedings IEEE Symposium on Research in Security and Privacy, Oakland, CA., May 16–18, 1994.
- Forrest, S., Balthrop, J., Glickman, M. and Ackley, D.. K. Park and W. Willins Eds. *The Internet as a Large-Scale Complex System*, Oxford University Press, 2005.
- George, Deleep. 2009. How the Brain Might Work: A Hierarchical and Temporal Model for Learning and Recognition. PhD thesis, Stanford University.
- Haack, Jereme N., Glenn A. Fink, Wendy M. Maiden, David McKinnon, and Errin W. Fulp. 2009. Mixed-Initiative Cyber Security: Putting Humans in the Right Loop. www.cs.wfu.edu/~fulp/Papers/mims09f.pdf
- Hawkins, Jeff. 2010. Video talk of “Hierarchical Temporal Memory and FDR (Fixed-sparsity Distributed Representations),” University of British Columbia, Vancouver, March 18. Video: www.youtube.com/watch?v=TDzr0_fbnVk.
- Hawkins, Jeff, Subutai Ahmad, Donna Dubinsky, and Numenta Employees. 2010. Hierarchical Temporal Memory Including HTM Cortical Learning Algorithms. Version 0.2, December 10. www.numenta.com/htm-overview/education.php
- Hofmeyr S. and S. Forrest. 2000. "Architecture for an Artificial Immune System." *Evolutionary Computation* 7(1), Morgan-Kaufmann, San Francisco, CA, pp. 1289-1296. http://cs.unm.edu/~forrest/publications/hofmeyr_forrest.pdf
- Pennisi, Elizabeth, Researchers Trade Insights About Gene Swapping, *Science*, Vol. 305:334-335, 16 July, 2004.
- Smets, Barth F. and Tamar Barkay. 2005. Horizontal gene transfer: perspectives at a crossroads of scientific disciplines. *Nature Reviews Microbiology* 3, 675-678 (September 2005).
- Woese, Carl. 2000. Interpreting the Universal Phylogenetic Tree. *PNAS*. 97(15):8392-6. www.ncbi.nlm.nih.gov/pmc/articles/PMC26958/pdf/pq008392.pdf
- Zhang, C., Zhang, J., Liu, S., and Liu, Y., Network Intrusion Active Defense Model Based on Artificial Immune System. Fourth International Conference on Natural Computation, Jinan, China, October 18-20, 2008.