



E2xB: a Domain-specific string-matching algorithm for Intrusion Detection



An IST Project

<http://www.ist-scampi.org/>



Evangelos Markatos
markatos@ics.forth.gr

<http://www.ics.forth.gr/~markatos>

Institute of Computer Science (ICS)

Foundation for Research and Technology – Hellas (FORTH)

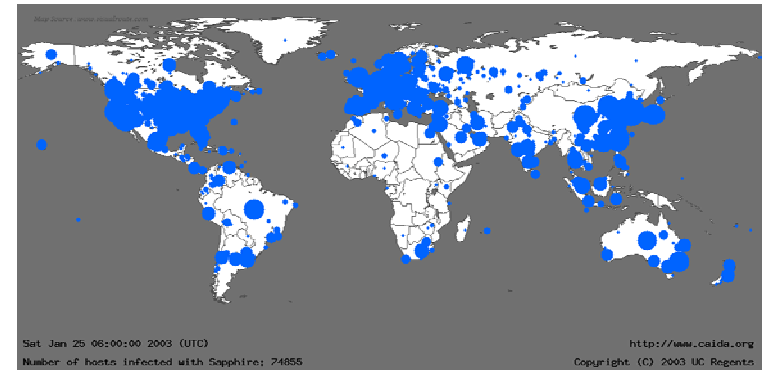
In cooperation with

Kostas Anagnostakis, Spiros Antonatos, and Michael Polychronakis

- What is Intrusion Detection?
- What is the Problem we work on?
 - High-speed string searching for IDS
- E2xB string-searching algorithm
- Implementation in SNORT
 - Open-source Intrusion Detection System
- Performance Results
- Summary and Conclusions



- Computers are vulnerable to cyberattacks
 - Intrusion attempts (e.g. “su root”)
 - Worms (self-replicating malicious programs)
- Network-based Intrusion Detection Systems
 - Provide early warning
 - Of such cyberattacks
 - Even for home computers
 - Poorly administered



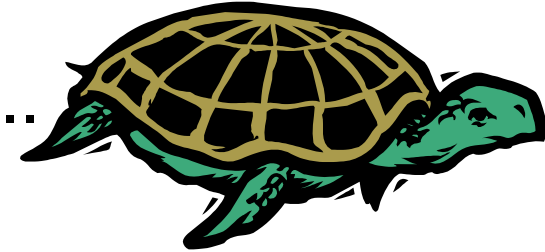
- Network Intrusion Detection Systems

- are based on a set of signatures:

- Each cyberattack has a signature:
 - CODE-RED signature: “GET /default.ida?NNNNNNNNNNNN”
 - PERL exploit (“backdoor”) signature: “GET /perl.exe”
- Each packet is checked against every signature
 - If it matches,
 - » *an alert is generated*
 - » *the packet is logged*
 - » *the packet may be dropped (Intrusion Prevention Systems)*



- Network Intrusion Detection is slow...
- Fisk and Varghese studied “Snort”:
 - a popular open-source Intrusion Detection System
 - A single SNORT system can effectively monitor traffic **up to 60 Mbps** (on 733 MHz Pentium III)
- Our results [CCN 02] (1.7 GHz processor):
 - Snort’s efficiency depends on the traffic:
 - 101 Mbps (eth0.dump)
 - 134 Mbps (eth0.dump7)
 - 53 Mbps (eth2.dump2)



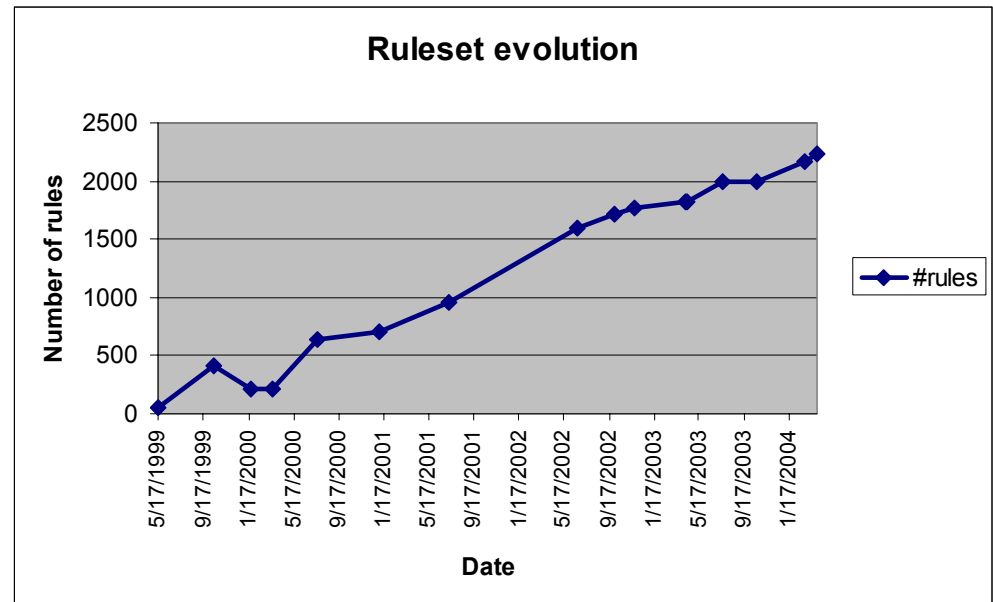
SNORT can process no more than **150 Mbps**
while at the same time
networks deliver **2.5-10 Gbps**

How are we going to close this GAP?

Study IDS performance overheads and
provide better algorithms to reduce these
overheads



- In string matching:
 - Trying to match signatures against packet payload:
 - 31%-81% of NIDS processing time
- Things are probably going to get worse:
 - Set of signatures is growing
 - may induce more caches misses



- Boyer-Moore (single-signature)
 - Shifts the pattern based on mismatch prefix
 - Bad-character and good-suffix shift
- Aho-Corasick (multi-signatures)
 - Builds automaton accepting set of signatures
 - Certain nodes indicate (multiple?) signature matching
- Setwise Boyer-moore (multi-signatures)
 - Boyer-Moore adapted to multiple signatures
 - uses trie data-structure
- Snort 1.x uses hybrid (Fisk and Varghese - FVh)



Exclusion-Based Matching



An IST Project

<http://www.ist-scampi.org/>

- Observations:
 - **Most of the time the system is not under attack**
 - i.e. the current packet does not contain any (attack) signatures
 - Even when the system is attacked, it is usually **attacked by a single attacker**
 - i.e. the current packet does not contain most (attack) signatures
 - Even when the system is attacked by multiple attackers, **each packet contains a single attack**
 - i.e. the current packet does not contain most (attack) signatures
- Overall,
 - Most of the times
 - the current packet does not contain most (any) of the signatures
 - most of the signatures are not sub-strings of the current packet
 - When searching for a signature in a packet, the answer is usually NO: i.e. the packet does not contain the signature



Exclusion-Based Matching



An IST Project

<http://www.ist-scampi.org/>

- Fact:
 - **Common case:** the signature is not a sub-string of the input packet
- Opportunity:
 - *How can we make the common case fast?*
- Idea:
 - Find an algorithm which:
 - If a signature is **NOT** a sub-string of the input packet (i.e. common case) it replies **fast**
 - If a signature is a sub-string of the input packet (i.e. rare case) speed is not very important

- Check if signature S is not in the input P
- Basic idea
 - If one character in signature S does not appear in the input packet P then S is not in P
- Example:

P:	T	E	S	T	P	A	C	K	E	T
S:	E	V	R							

Input packet: "ABCABBF"

A	1
B	1
C	1
D	0
E	0
F	1

•
•
•

- **Problem:**
 - How can we decide quickly whether a character exists in input packet P ?
- **Solution: Pre-processing**
 - For all characters in packet P mark a corresponding element in a bitmap
 - If P contains the j_{th} character, the j_{th} element of the bitmap is marked

Input packet: "ABCABBF"

A	1
B	1
C	1
D	0
E	0
F	1

- Search
 - e.g: signature "AFE"
 - Check signature against the bitmap
- Positive answer case
 - e.g. signature: "ABF"
 - "false positives"
 - Falls back to existing string searching algorithms

•
•
•

A	45
B	45
C	45
D	32
E	30
F	54

•
•
•

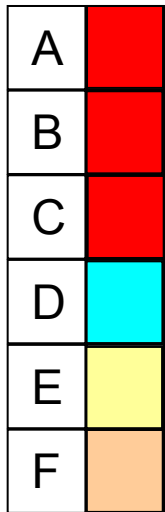
- For each packet
 - The bitmap needs to be reset!
 - Induces high overhead!
- Idea:
 - Each place in the bitmap that should have been “1” is marked with the current packet’s ID (identification) instead
 - “1’s” correspond to current packet’s ID
 - “0’s” correspond to old packet’s IDs
- BitMAP becomes ColorMAP

 Current packet ID

   Older packet’s IDs

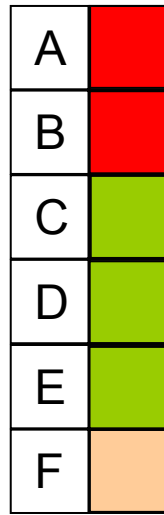
Evangelos Markatos

Start state



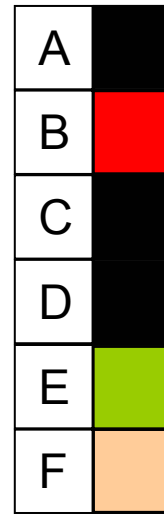
•
•
•

Input packet: "CDE"



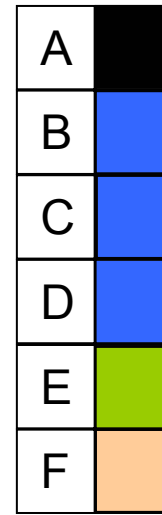
•
•
•

Input packet: "ACD"



•
•
•

Input packet: "BCD"



•
•
•

How long is a “character”?

A	
B	
C	
D	
E	
F	

8-bit characters

AA	
AB	
AC	

16-bit characters

Input Packet: “ABAAB”

•
•
•

BA	
BB	
BC	

•
•
•

- Longer characters
 - Better accuracy – fewer false positives
 - Higher storage requirements

- Implementation in Snort-1.9.0
- NIDS Host
 - P4 at 1.7 GHz , 512 MB main memory
 - 8KB L1 data cache, 256 KB L2 cache
 - Linux operating system
- DEFCON packet traces
 - Contain numerous network attacks
 - Real payload
- Web traces (real payload)
- NLANR traces (synthetic payload)
- All snort signatures used (1243)



An IST Project

Traffic trace files



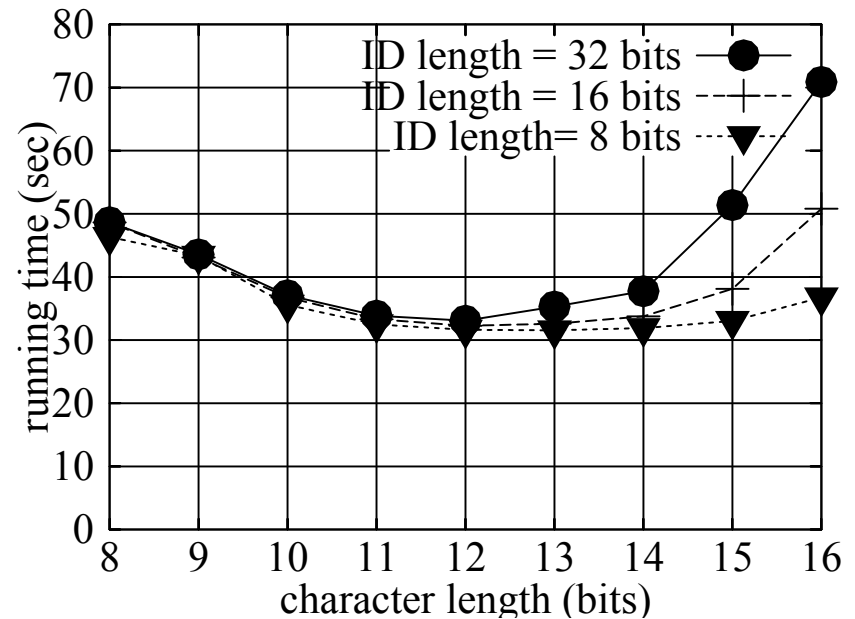
<http://www.ist-scampi.org/>

Trace Name	Average packet size	Source
eth0.dump2	835	DEFCON
eth0.dump4	1481	DEFCON
eth0.dump8	582	DEFCON
Webtrace	761	ics.forth.gr
IND	703	NLANR
MRA	760	NLANR
UCNET	422	U of Crete
UCNET II	413	U of Crete

- How long should a character be?
- Large characters
 - reduce false matches
 - Increase data structures size (and running time)

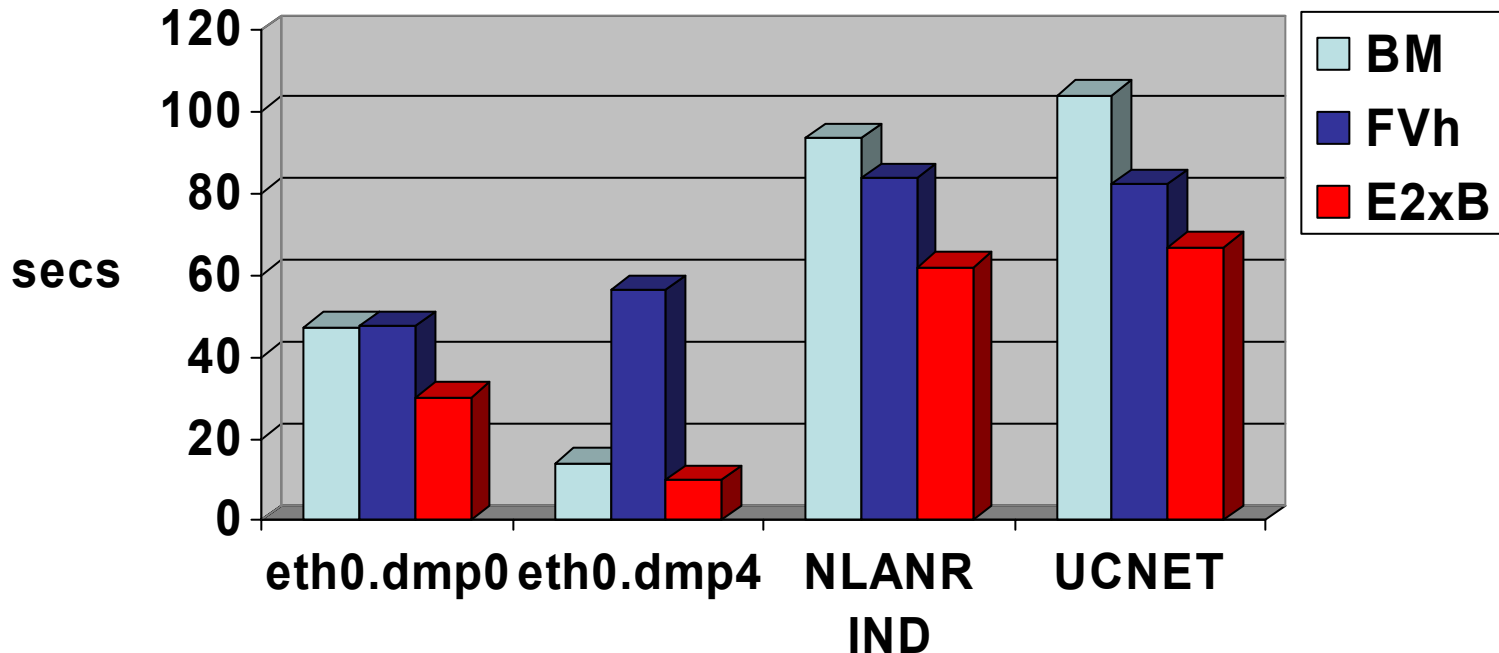


Evangelos Markatos



FORTH

Snort's completion time



- E2xB is always better
 - It says NO quickly!

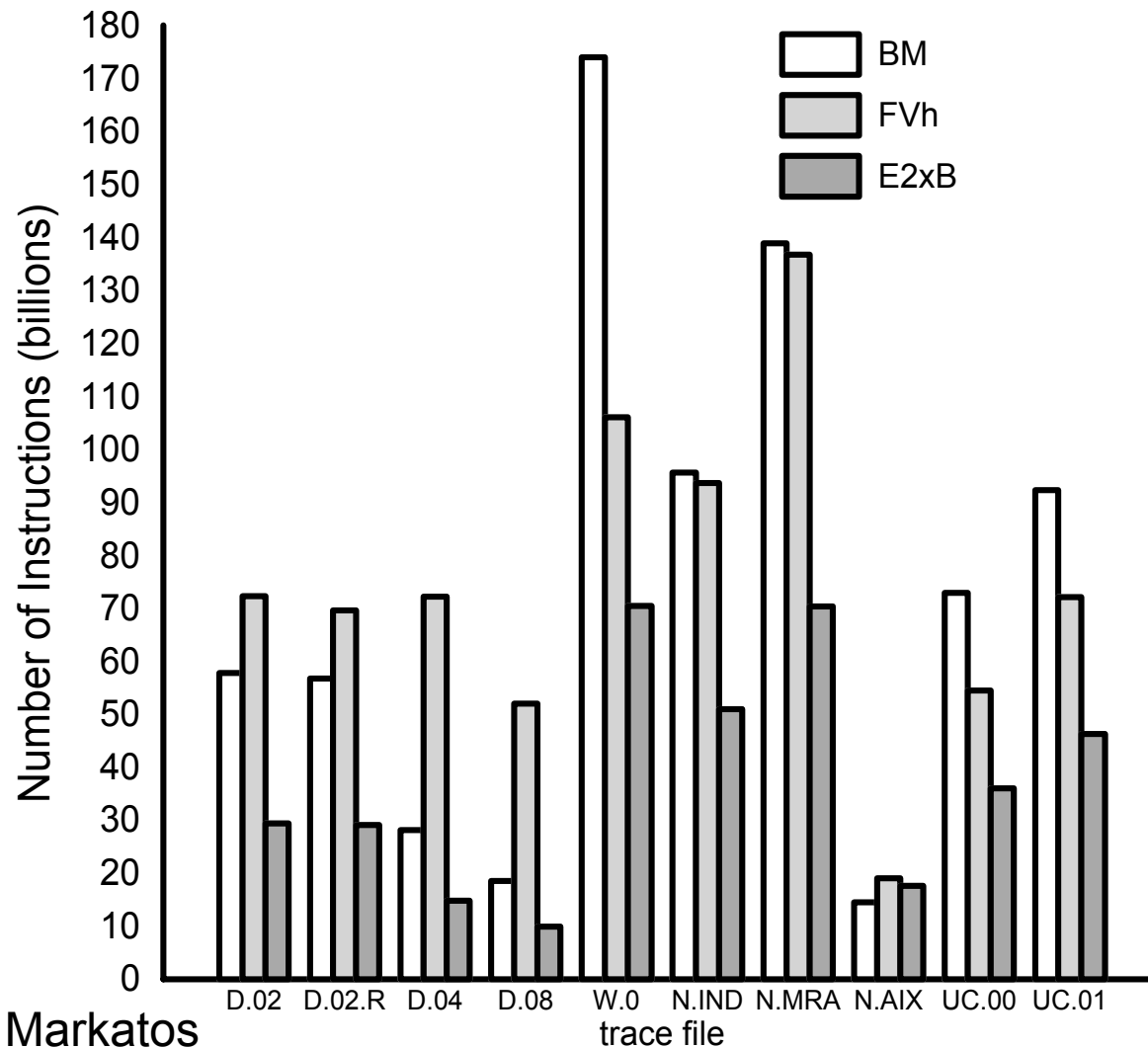


Executed Instructions

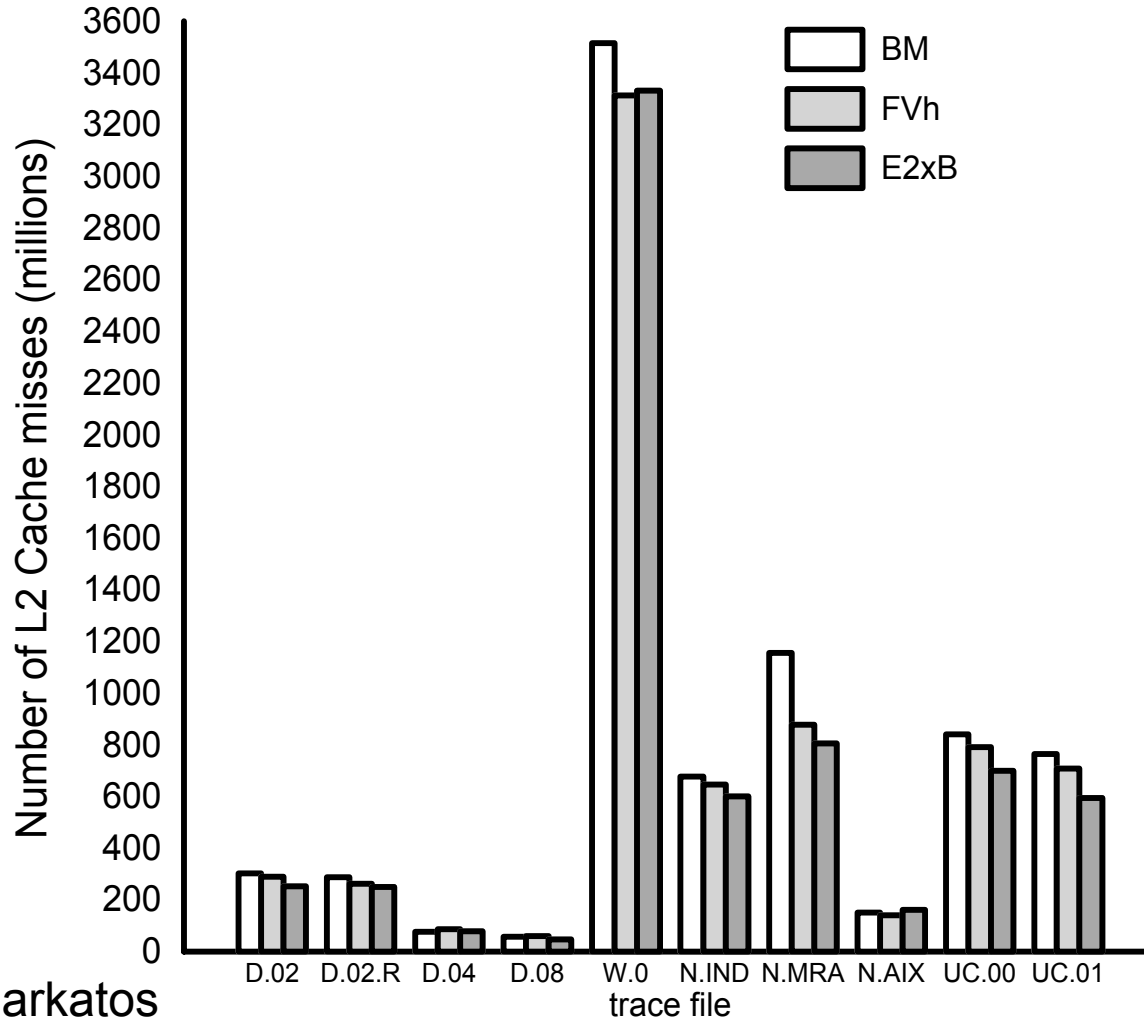


An IST Project

<http://www.ist-scampi.org/>



L2 Misses



- Reverse-logic algorithm
- Implemented in Snort
- Evaluated in realistic conditions
- E2xB improves NIDS performance
 - Quickly decides if pattern is NOT in input
- Fast IDS will become increasingly important in the near future!



E2xB: a Domain-specific string-matching algorithm for Intrusion Detection



An IST Project

<http://www.ist-scampi.org/>



Evangelos Markatos

markatos@ics.forth.gr

<http://www.ics.forth.gr/~markatos>

Institute of Computer Science (ICS)

Foundation for Research and Technology – Hellas (FORTH)

In cooperation with

Kostas Anagnostakis, Spiros Antonatos, and Michael Polychronakis

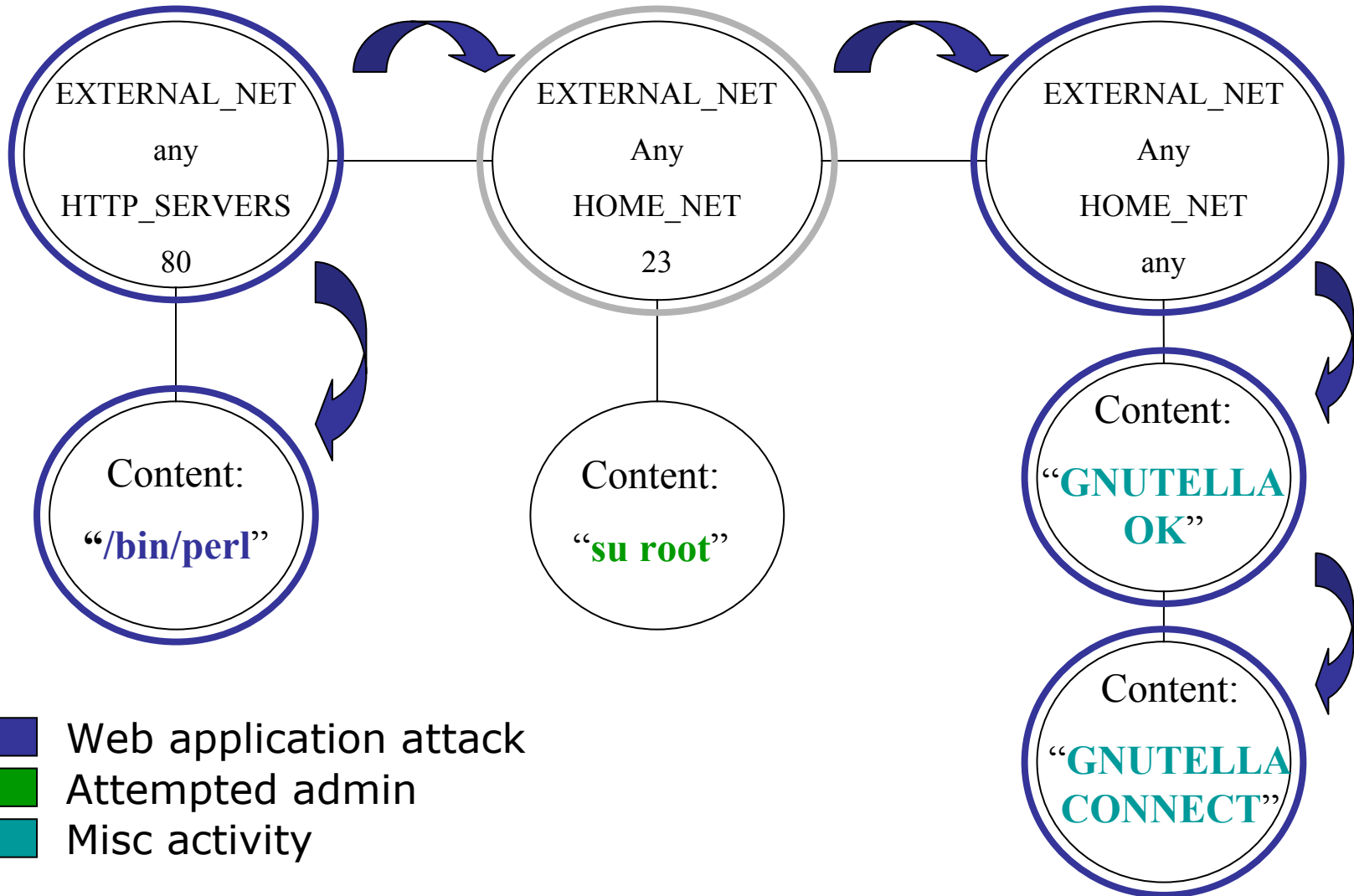


An IST Project

Back-up Slides



<http://www.ist-scampi.org/>



- Web application attack
- Attempted admin
- Misc activity

