

Using Contextual Information to Enhance Intrusion Detection Systems

François Gagnon

Ph.D. Student

`fgagnon@sce.carleton.ca`

`www.nmai.ca`

Joint work with

Canada's Communication Research Center

`www.crc.ca`

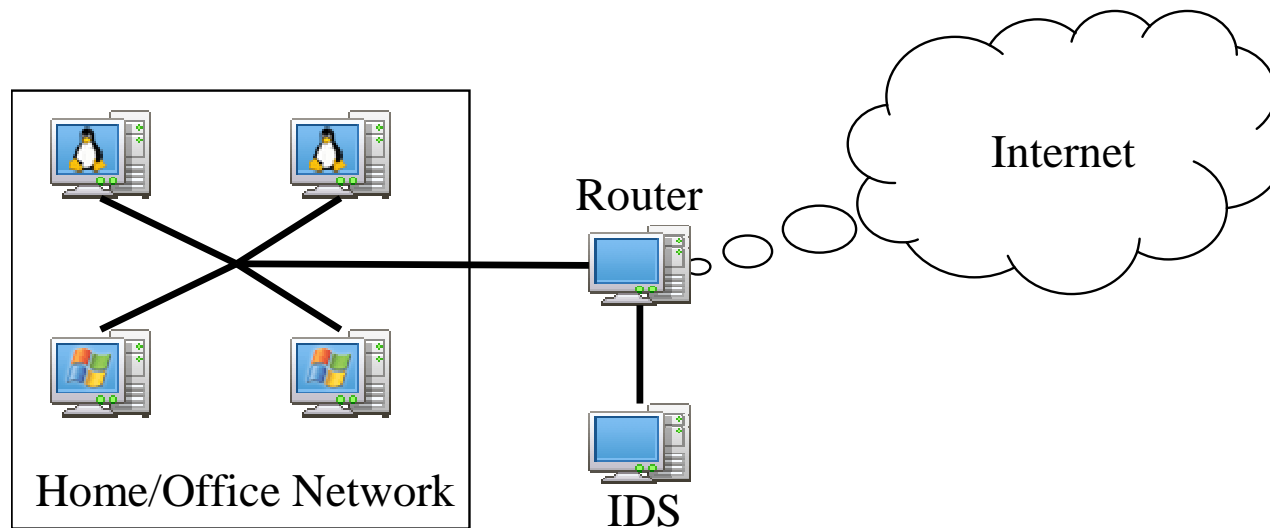


Outline

- Intrusion Detection Background
- Context in Intrusion Detection
- Experiment Setup
- Potential of Target Configuration Information
- Using Current Tools for Context Gathering
- Conclusion
- Future Work

Intrusion Detection Background

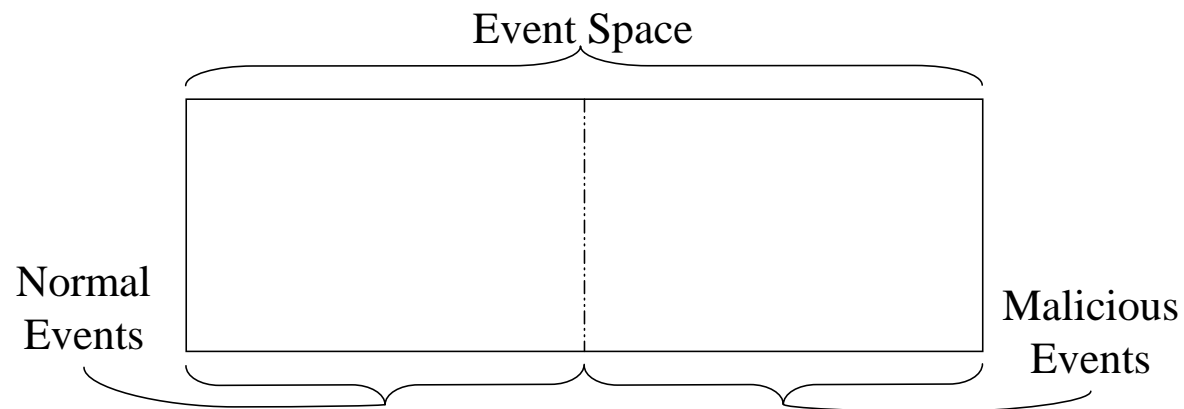
What is an IDS ?



Signature-based IDSes generate too many non-critical alarms.

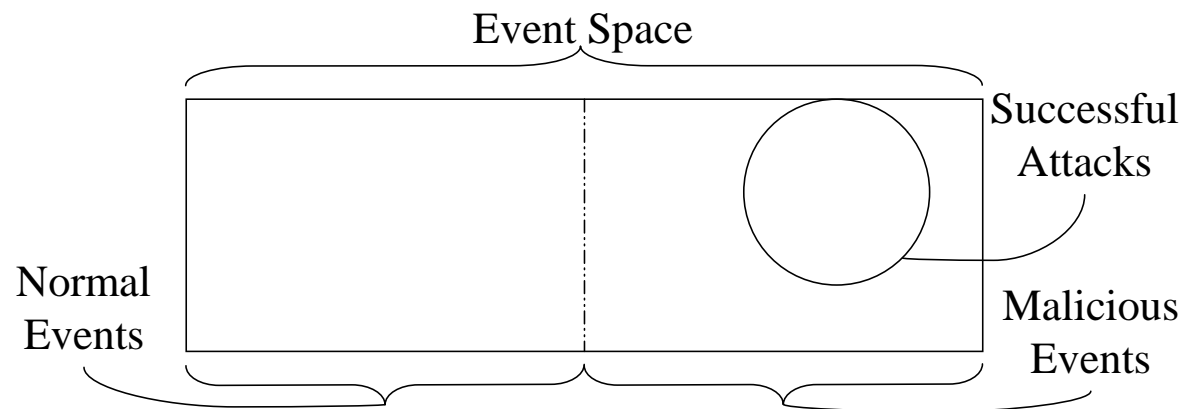
Intrusion Detection Background

Non-critical alarms are not indicative of a plausible threat.



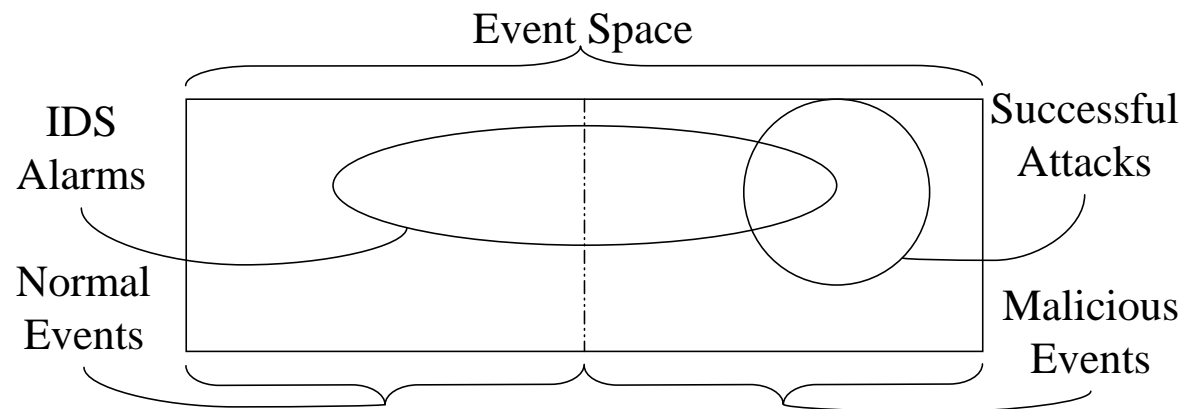
Intrusion Detection Background

Non-critical alarms are not indicative of a plausible threat.



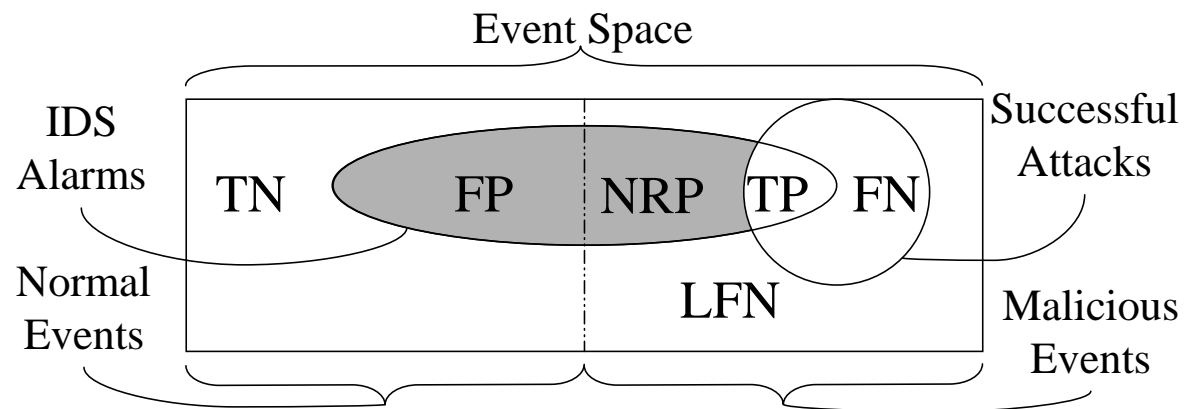
Intrusion Detection Background

Non-critical alarms are not indicative of a plausible threat.



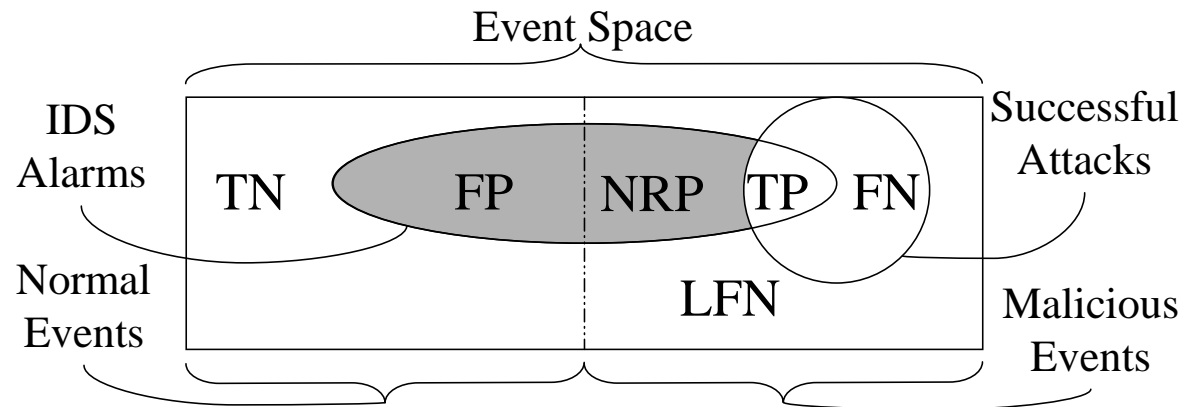
Intrusion Detection Background

Non-critical alarms are not indicative of a plausible threat.



Intrusion Detection Background

Non-critical alarms are not indicative of a plausible threat.



Non-Critical Alarm: A false positive (alarm related to normal background traffic) or a non-relevant positive (alarm related to an unsuccessful attack attempt).

- They pose two problems:
 - Distract security officers from real threats.
 - Prevent automatically blocking attacks.

Context in Intrusion Detection

Considering the context surrounding an event helps deciding whether the corresponding alarm is critical.

Four elements of IDS Context:

- Network-related information (e.g., topology, protocol specification)
- **Target configuration information** (OS and App)
- Vulnerability assessment
- Attack side effects (e.g., target reaction)

Contextual Approach Properties

	Network Information	Target Configuration	Vulnerability Assessment	Attack Side Effect
Information Variability	Static	Nearly Static	Nearly Static	N/A
Gathering Timeframe	a priori	a priori	a priori	a posteriori
Gathering Mode	Manual & passive	Passive & active	Active	Passive & active
Intrusive	N/A	Variable	Yes	No
Safe	N/A	Yes	Not entirely	Yes

Experiment Setup

- Using freely available dataset from CRC
- 92 exploits, 95 targets, 4,857 traces (1 trace = 1 attack attempt)
- Well-documented
 - Target OS and App
 - Attack result (success/failure)
- IDS alarms (Snort):
 - Target IP
 - Attack id (BID securityfocus.org)

Windows 9x NetBIOS NULL Name	
BID:	1163
Vuln:	Microsoft Windows 98 Microsoft Windows 95

Attack BID + target OS/App => decide whether the alarm is critical

Attack result => verify our decision

Four Algorithms

ContextOS:

- (1) by default, assign A;
- (2) if the target OS is listed as non-vulnerable for this exploit, then assign NC;
- (3) if the target OS is not listed as vulnerable for the BID and
 - (3.1) the products listed as vulnerable are all OSes, then assign NC.

ContextApp: considers only application

ContextOSApp: considers both OS and App

ContextOSDeduction: considers only OS and deduce some App info from OS
(e.g., Microsoft IIS cannot run on a Linux computer)

Performance Measures

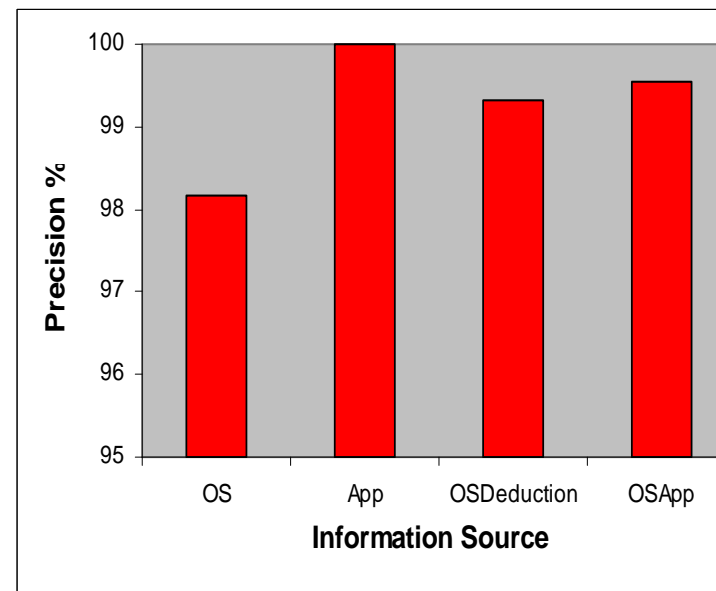
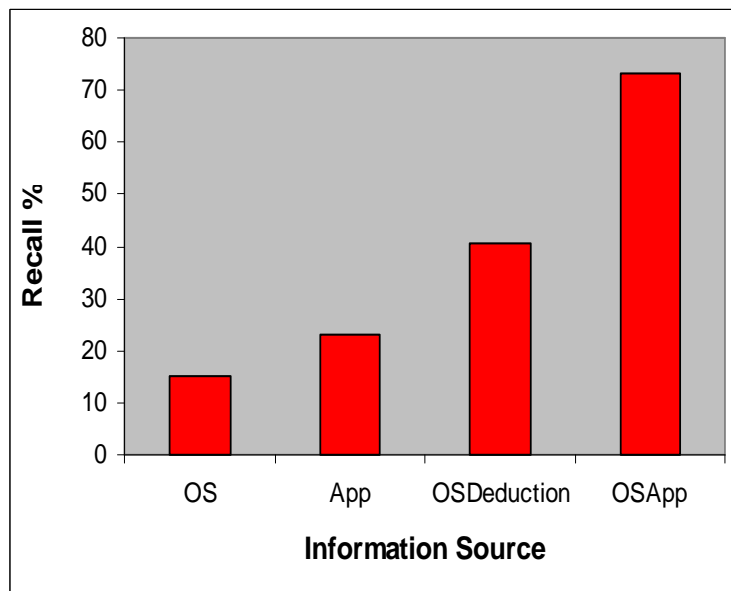
Using the usual information retrieval performance measures.

$$\textit{Precision} = \frac{\# \text{ non-critical alarms classified as NC}}{\# \text{ of alarms classified as NC}}$$

$$\textit{Recall} = \frac{\# \text{ non-critical alarms classified as NC}}{\# \text{ of non-critical alarms}}$$

Potential of Target Configuration Information

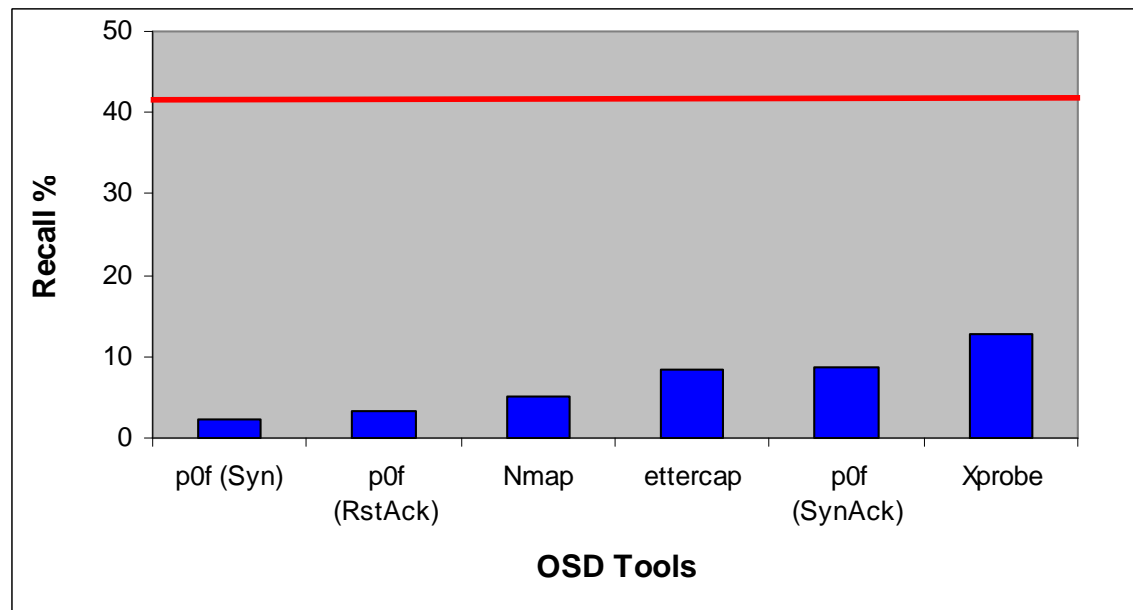
Assuming we know the exact target configuration (OS and App).



- Errors (precision decrease) are due to missing entries on securityfocus.
- Target configuration is very helpful to filter non-critical alarms (73%).
- Unrealistic assumption.

Current Tools for Context Gathering - OS

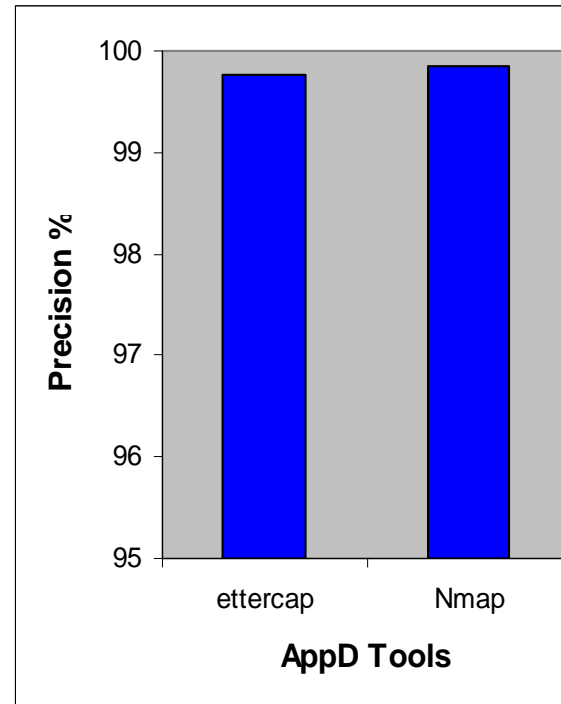
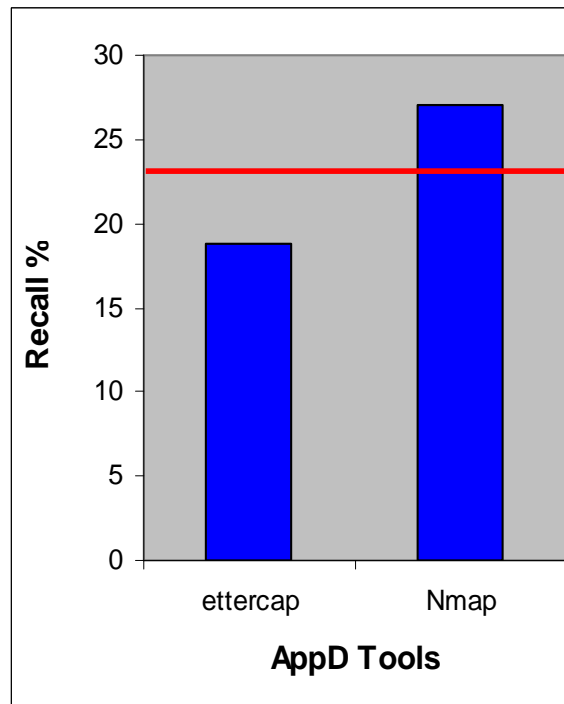
Using OS discovery tools to gather the context information.
Using the ContextOSDeduction algorithm.



- Current OSD tools are not nearly good enough (13% vs 40%).

Current Tools for Context Gathering - App

Using App discovery tools to gather the context information.
Using the ContextApp algorithm.



- How can Nmap be better than the ideal case (27% vs 23%)?

Current Tools for Context Gathering - App

Suppose the target application (Microsoft IIS FTP) is vulnerable to the attack, but the attack fails anyway (thus it is non-critical):

- The alarm is classified A by ContextApp with exact knowledge.
- This means 0/1 for recall.

Suppose Nmap thinks the target application is wuftp (not vulnerable):

- The alarm is now classified NC by ContextApp with Nmap.
- This means 1/1 for recall.

Those mistakes should result in a decrease of precision for Nmap (successful attack misclassified as NC).

The dataset does not have enough successful attacks.

Conclusion

- Target configuration is extremely useful for IDS context:
 - Filtering 73% of non-critical alarms.
 - Not filtering critical alarms.
- OSD tools are not adequate to gather the required contextual information (they achieve only 1/3 of potential).
- New OS discovery tools are required.
- The dataset does not contain enough successful attack to illustrate the mistakes made by discovery tools.

Limitation: Possibility for evasion.

Future Work

- Compare the effectiveness of the different IDS context elements.
- Enhance the dataset to expose context gathering mistakes.
- Develop a new OS discovery approach (HOSD).
- Prevent an attacker from fooling the system.