Flow Dependency Discovery and Analysis

Samer Mehri and Olivier Festor

Team MADYNES - INRIA Nancy - Grand Est

October 30, 2008

Outline

1. Introduction
2. Existing approaches
3. Our contribution
4. Evaluation and analysis
5. Conclusion and future work

Introduction

Dependencies study is important for :

- fault and configuration management
- improving the performance of systems and networks.

Problem statement

Existing approaches have limits:

- network resource consumption
- legal issues
- high pre-existing knowledge base

Our objective

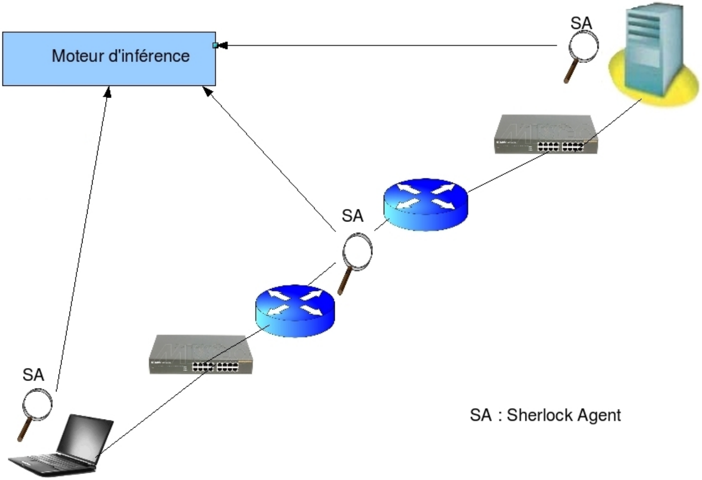
Adapt existing approaches to easily obtainable data (IPFix/Netflow records).

Existing Approaches

- Sherlock
- Kachima et al.
- Active Dynamic Discovery (ADD)

Sherlock

- Towards Highly Reliable Enterprise Network Services Via Inference of Multi-level Dependencies.
- Paramvir Bahl, Ranveer Chandra, Albert Greenberg, Srikanth Kandula,David A. Maltz, Ming Zhang.
- SIGCOMM'2007, August 27 -31, 2007

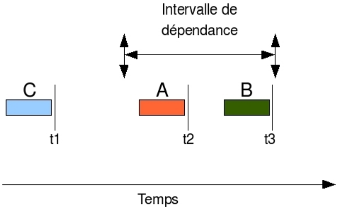

Moteur d'inférence
SA
SA
SA
SA : Sherlock Agent

Service discovery

- (IPdest,IPSource,Portdest,PortSource)

Dependency discovery

- Service B is considered to be depending on A if at least one invocation of A appears before B with an "interesting" probability during a given time interval named dependency interval.

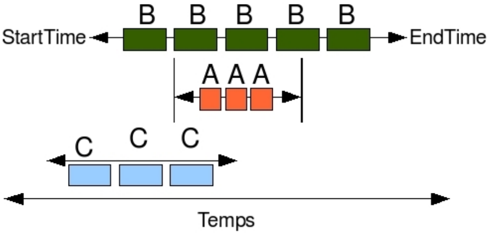


B dépend de A

Kachima et al.

- Network-based Problem Detection for Distributed Systems
- Hisashi Kashima, Tadashi Tsumura, Tsuyoshi, Takahide Nogayama, Ryo Hirade, Hiroaki Etoh Takeshi Fukuda
- 21st International Conference on Data Engineering, Japan , 5-8 Avril 2005

- A technique based on the history D of start and stop times of invocations and services execution.



B dépend de A

- Requires deep-packet inspection and semantics knowledge of monitored services.

Active Dedendency Discovery (Keller & Kar, 2001

- Active approach
- injection of a single fault in one entity causes the interruption of one or several applications $---\triangleright$ dependecy between the faulty entity and the affected entities.

Comparison table

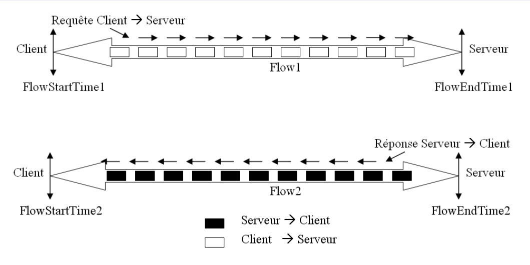
Technique	Dependency Graph	Input Data	Architecture	Mode
Sherlock	Physical + Service levels	IP packet headers	centralized and distributed	Passive
Kachima et al.	Service level	IP headers + service payload	distributed	Passive
ADD	Physical and Services layers	Fault injection results	distributed	Active

Our Approach

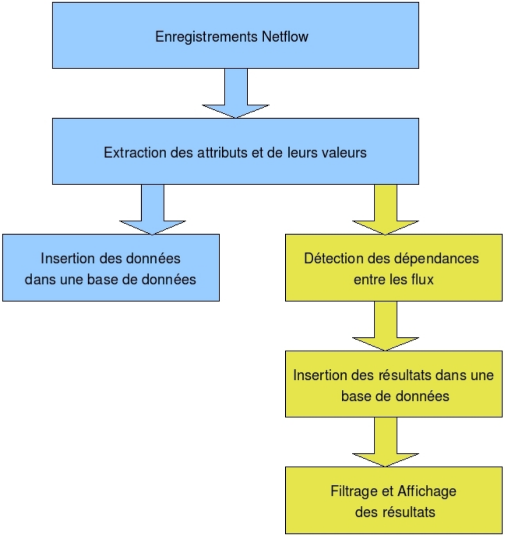
A technique to discover dependencies based on the analysis of flow records only.

Considered fields

FlowStartTime
FlowEndTime
SourceIP
DestinationIP
SourcePort
DestinationPort
Protocol



Functional Requirements



Enregistrements Netflow

Extraction des attributs et de leurs valeurs

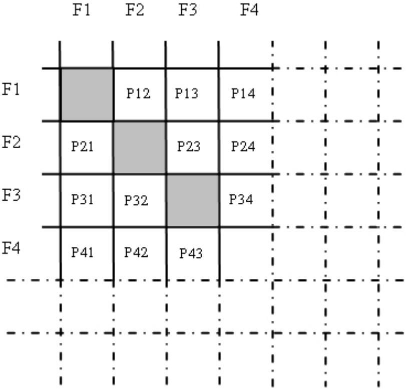
Insertion des données dans une base de données

Détection des dépendances entre les flux

Insertion des résultats dans une base de données

Filtrage et Affichage des résultats

Dependency Frequency Matrix



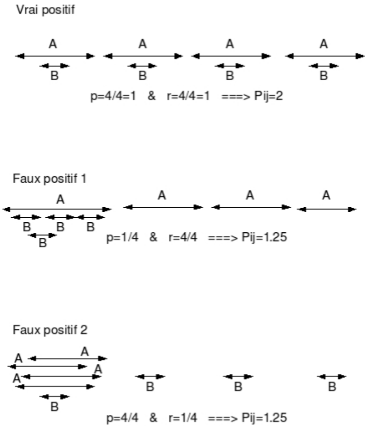
	F1	F2	F3	F4
F1		P12	P13	P14
F2	P21		P23	P24
F3	P31	P32		P34
F4	P41	P42	P43	

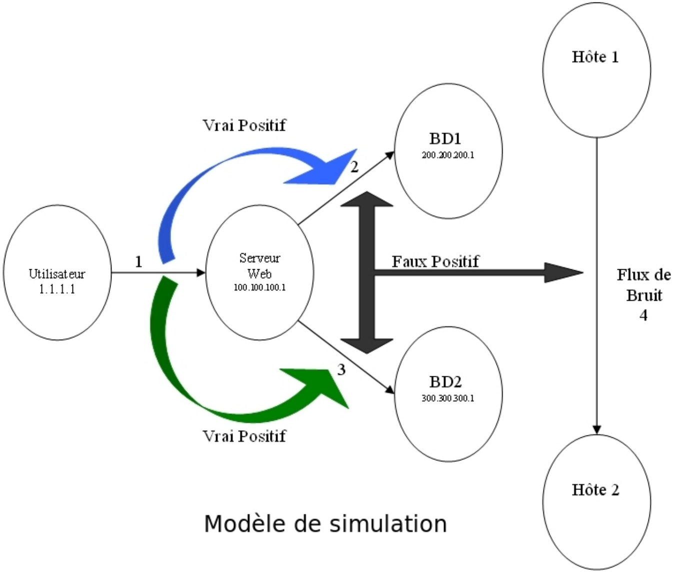
$P_{ij}/2$ is the probability that flow Fi depends on flow Fj.

$P_{ij} = max(p, r) + p * r$ given p=$\frac{\#(F_i/F_i>F_j)}{\#F_i}$ et r=$\frac{\#(F_j/F_i>F_j)}{\#F_j}$.

$P_{ij} = max(p, r) + p * r$ given p=$\frac{\#(F_i/F_i{>}F_j)}{\#F_i}$ et r=$\frac{\#(F_j/F_i{>}F_j)}{\#F_j}$.

True vs False positives


Vrai positif
A
A
A
A
B
B
B
B
p=4/4=1 & r=4/4=1 ===> Pij=2
Faux positif 1
A
A
A
A
B
B
B
B
p=1/4 & r=4/4 ===> Pij=1.25
Faux positif 2
A
A
A
A
B
B
B
B
p=4/4 & r=1/4 ===> Pij=1.25



Modèle de simulation

Simulation Parameters

Parameters	Description
ExecTime	Flow living time
NbFlux	Number of generated flows or generated instances
NbHotes	Nof hosts generating traffic
%BD1	Probabilitof redirection towards DB1.
%BD2	Probability of redirection twads DB2.
(SST) et (SET)	SimulationStartTime and SimulationEndTime.

Two environments

1. Loaded network: dependecy flows occupy 90% of simulation time.
   - 500 generated flows
   - Execution time of flows follows a normal law N(100ms,10)
   - Simulation duration 60s
2. Unloaded networks: dependent flows occupy 15% of simulation time.
   - 100 generated flows
   - Execution time of flows follows a normal law N(100ms,10)
   - Simulation duration 60s

No? Future work

- Real evaluation
- Investigation of a hybrid approach
- Simulation on different flow models

We are looking for students to join us on this topic, and several others ;-)