國立交通大學

網路工程研究所

碩士論文

低儲存空間消耗的錄製真實流量

與回復有效狀態的重播真實流量技術

Low-Storage Capture and Loss-Recovery Stateful Replay

of Real Flows

研 究 生：鄭宗寰

指導教授：林盈達　教授

中 華 民 國 九 十 九 年 六 月

低儲存空間消耗的錄製真實流量

與回復有效狀態的重播真實流量技術

Low-Storage Capture and Loss-Recovery Stateful Replay of Real Flows

研 究 生：鄭宗寰　　Student: Tsung-Huan Cheng

指導教授：林盈達　　Advisor: Dr. Ying-Dar Lin

國立交通大學

網路工程研究所

碩士論文

A Thesis

Submitted to Institutes of Network Engineering
College of Computer Science
National Chiao Tung University
in partial Fulfillment of the Requirements
for the Degree of
Master
In

Network Engineering

June 2009

HsinChu, Taiwan, Republic of China

中華民國九十九年六月

低儲存空間消耗的錄製真實流量

與回復有效狀態的重播真實流量技術

學生: 鄭宗寰 **指導教授: 林盈達**

國立交通大學網路工程研究所

摘要

網路產品在真實流量上仍會遇到許多實驗室模擬網路流量測試所無法找到的問題，而這些問題可藉由重播真實流量的測試來找到。由於真實流量由許多真實使用者所產生，在錄製時會快速消耗儲存空間使得錄製時間無法很長，與漏錄影響重播的準確性。在重播時要追蹤流量的有效狀態以應付待測物對流量的反應，並且要能很快的重製事件的發生以便開發者除錯或尋找原因。因此本論文以(N, M, P)錄製機制針對每條連線只錄製連線的前 N 位元與錄製剩餘 P 個封包的前 M 幾位元來節省儲存空間，達到節省 87%的儲存空間但保留 99.74%的攻擊事件。並且實作實作 SocketReplay 重播工具以回復漏錄重播追蹤 TCP 串流使得漏錄對觸發事件的數量成比例的下降而不會驟降，有效狀態的重播使待測物認為流量是真實的，選擇性的重播以漸增方式階段性尋找造成事件的少數流量，達到重製事件僅需從千條的連線中挑出幾十條連線重製攻擊或病毒的事件。

關鍵字: 流量重播、流量錄製、真實流量、測試、缺陷

Low-Storage Capture and Loss-Recovery Stateful Replay of Real Flows

Student: Tsung-Huan Cheng **Advisor: Dr. Ying-Dar Lin**

Department of Computer Science

National Chiao Tung University

Abstract

Model-based traffic generated in the laboratory might not trigger some device defects found only by replaying traffic flows. However, capturing real flows might result in high storage cost and capture loss; the latter affects the accuracy of replay. Replaying real flows should be accurate and also stateful enough to adapt to device reaction. It should reproduce a defect efficiently in helping developers to identify the flows triggering the defect. Therefore, this work first presents the *(N, M, P)* capture scheme to capture *N* bytes per flow of data and *M* bytes of *P* packets after the N bytes. This scheme reduces 87% storage cost while retaining 99.74% of attack traffic. Next we develop a tool named SocketReplay with the mechanisms of loss-recovery, stateful replay, and selective replay to track TCP sequence numbers to identify capture loss, recover these incomplete flows, follow the TCP/IP protocol behavior, and incrementally select flows to replay. Numerical results show that SocketReplay retains the accuracy and statefulness in triggering device defects and could reduce replayed flows from thousands to tens.

Keywords: traffic replay, traffic capture, real flows, testing, defects

Content

List of Figures

List of Tables

Chapter 1 Introduction

Generating network traffic as if hosts interact with each other for testing new network applications, systems, and protocols is highly demanded for network research community, developer and tester. Currently, there are two main approaches, model-based traffic generation and trace-based traffic replay, to generate network traffic. The model-based approach generates network traffic according to protocol specifications and it is easy to configure the desired parameters such as request formats for each test case. The trace-based approach replays packet traces captured in real environments called real flows and it includes more realistic and varied network conditions and behaviors such as P2P, video streaming, on-line game and proprietary protocols which is hard to model. Therefore, using trace-based traffic replay can trigger many unexposed device bugs and alerts that are difficultly discovered by adopting model-based traffic generation.

The intuitive solution is to replay traffic as captured such as Tcpreplay [1]. However, the quality of replay might be low due to insufficient efficiency and accuracy. Efficiency means how fast the traffic can be replayed while accuracy represents whether replay is meaningful on testing DUT (Device Under Test). Many previous works had contributed to raise the efficiency of traffic replay [2] and improve its accuracy [3-8]. Some studies tried to replay network layer or transport layer accurately [4-6] while the others tried to replay application layer accurately [7, 8]. However, these existing studies all require the complete trace of packets to guarantee the accuracy of traffic replay.

In large-scale networks, the miss in capturing, called capture loss throughout the thesis, may happen due to limited I/O speed of network card, memory, or disk,

causing that the previous approaches are not very suitable to the environments. On the other hand, storing all captured traffic needs a lot of storage. Therefore, an appropriate method for large-scale networks is required to conquer the problems of capture loss and huge requirements of storage. Currently, the study of [9] showed how to capture packets in 10Gbps by splitting traffic and tuning parameters. Also, some studies [10-11] proposed a method to only capture partial traffic by using the heavy-tail nature of traffic to reduce storage cost.

In this thesis, we design and implement a tool named SocketReplay, which can provide an effective way to capture and replay large-scale network traffic. SocketReplay mainly includes four features: (1) *low-storage capture* records partial network traffic according to types of concerned traffic so that storage cost can be significantly reduced; (2) *loss-recovery* recovers incomplete connections due to capture loss so that a complete TCP stream can be replayed; (3)*stateful replay* mimics TCP/IP protocols and replay payloads so that TCP semantics can be maintained; and (4) *selective replay* reproduces the abnormal events, such as bugs or alerts, with minimal replaying traces so that the events can be analyzed efficiently.

The rest of this thesis is organized as follows. Chapter 2 shows the background issues of capture and replay in large-scale environments and related work. Chapter 3 and Chapter 4 describe the design and implementation of our proposed method SocketReplay, respectively. Chapter 5 displays evaluation of our work. Finally chapter 6 concludes this work and gives some future directions.

Chapter 2 Background

2.1 Issues of traffic capture in large-scale environment

The quality of traffic capture affects the quality of experiments based on these