On Campus Beta Site: Architecture Designs, Operational Experience, and Top Product Defects

Ying-Dar Lin and I-Wei Chen, National Chiao Tung University
Po-Ching Lin, National Chung Cheng University
Chang-Sheng Chen and Chun-Hung Hsu, National Chiao Tung University

ABSTRACT

Testing network products in a beta site to reduce the possibility of customer found defects is a critical phase before marketing. We design and deploy an innovative beta site on the campus of National Chiao Tung University, Hsinchu, Taiwan. It can be used for developers to test and debug products, while maintaining network quality for network users. To satisfy the needs of developers, we set up environments and mechanisms, such as a variety of test zones for multiple types of products or systems under test (SUTs), remote control, degrees of traffic volume, and traffic profiling. For network users, we set up mechanisms of failure detection, notification, and recovery. The beta site network users are all volunteers. Test results show that beta site testing is good for finding stability and compatibility defects. The period starting from the beginning of a test until the next defect is found is called the time to fail (TTF). We call it converged if the TTF exceeds four weeks, and the convergence ratio is the percentage of SUTs that reach convergence. We find that the TTF increases with longer test duration, meaning that product quality improves through beta site testing. However, the convergence ratios are only 7 and 20 percent for test durations of one month and one year, respectively, meaning that few products operate faultlessly for a long duration. The convergence ratios also indicate that it takes much more time to enhance product quality to be converged. Therefore, if considering both marketing timing and product quality, one month is our suggested minimum TD for low-end and short-life-cycle products. However, we recommend one year as the minimum TD for high-end and long-life-cycle products.

INTRODUCTION

Network products undergo and are required to pass a series of tests during development [1]. For testing, products can be deployed in an operational network with common users in addition to functionality-driven tests in the laboratory. Such a testing network is called a beta site, and testing through the beta site is called beta site testing or field testing. Beta site testing can help discover product defects in the early stage and effectively reduce the number of defects found by customers (*customer found defects* [CFDs]) after the sale [2]. In our beta site we can test multiple product types at different levels in the network, from the link to the application level.

STIMULATING PRODUCT DEFECTS: LABORATORY OR FIELD?

Laboratory testing is not adequate for product testing; it is often difficult, expensive, time-intensive, and labor-intensive for laboratory testing to emulate the diversity and complexity of real-world networks, and to stimulate potential product problems. For example, in the application layer, peer-to-peer (P2P) applications, video streaming services, and online games are getting popular. These applications have proprietary protocols that can undermine the establishment of a comprehensive testbed using commodity PCs or commercial traffic generators. In the network and transport layers, various operating systems (OSs) and configurations, perhaps using IPv6, diversify TCP/IP behavior. In the link layer, various protocols (e.g., fast/gigabit/10G Ethernet, wireless LAN [WLAN], third generation [3G], and WiMAX) and network interface cards from many vendors also diversify link behavior. In terms of test coverage [3, 4], the exponentially growing number of combinations of all these force us to design a huge and ever growing number of test cases to cover the statements, branches, and paths in the code that are most likely to be executed in order to find potential bugs. Exhaustive coverage appears to be impossible. Therefore, we must complement laboratory testing with field testing to fill the gap — and the gap is getting large.

0163-6804/10/$25.00 © 2010 IEEE

Developers desire the convenience of testing and debugging, while network users demand network availability. A bias toward either party might either incur strong protests from users or sacrifice the quality of the beta site testing — a balance between them is necessary.

Balancing Two Conflicting Parties: Developers vs. Network Users

The design of a beta site can be viewed from two perspectives: that of the product developer and that of network users. Developers desire the convenience of testing and debugging, while network users demand network availability. A bias toward one party might either incur strong protests from users or sacrifice the quality of the beta site testing; hence, a balance between them is necessary.

Evaluation: Time to Fail and Convergence

We assess the testing methodology in this study based on the results for an operational campus beta site over a period of one year. We tally the numbers of critical problems found, and evaluate whether the product quality is improved. We also count the number of volunteers participating in the beta site over time as a measure of network quality. We use a time-to-fail (TTF) measure as a metric for evaluating improvements in product quality. TTF represents the period starting from the beginning of a *test round* until the *next* defect is found. The defect must be of the same severity level, and we only count critical level defects such as product crashes. For example, if a product is tested from time t_i, and defect $defect_j$ is found at time t_j, we say that the TTF of $defect_j = (t_j - t_i)$. After developers fix $defect_j$ and start another test round, if the TTF of the next defect is longer than the TTF of $defect_j$, we assume that the product quality is improving. Usually there will be several test rounds in the test project of a product. The *test duration* means how long a product has been tested from the first test round to a specific test round. If TTF ≥ 4 weeks, we call the system under test (SUT) *converged* and define the *convergence ratio* as the percentage of SUTs that reach convergence.

Related Work

Some research established testing environments with a large number of computers [5–7] using virtual LAN (VLAN) technology and a set of automatic modules for test engineers to install as testbeds for analyzing the results. There are also studies that emphasize generating test traffic based on a good traffic model [8, 9]. Some traffic models use the attributes gained from analyzing real world traffic traces to generate more realistic traffic. In this study, we would rather establish a beta site on the campus dormitory network. This beta site is more realistic than the above-mentioned approaches that use artificial test beds or traffic, since the products are tested in an operational campus network. Another set of research captures [10, 11] and replays [12–14] network traffic for product testing. Several institutes offer captured network traffic to academia for network research [15]. Because replaying is helpful for *reproducing* SUT problems encountered in the real world, we include replaying techniques in our beta site infrastructure.

The rest of the article is organized as follows. We discuss the needs of product developers and network users. We then describe our methodology for satisfying these needs. We then illustrate our standard operating procedure (SOP) in product testing and give two case studies. We then enumerate the major product defects found and some defect statistics, thereby evaluating the benefits to developers. Moreover, we also explain two practical cases in detail. The final section concludes this work and points to future work.

Concerns in Building the Beta Site

From the point of view of the developer the possible concerns are as follows.

Variety of Test Zones

Due to the variety of network products (end-user software, access points, L2 Ethernet switches, residential gateways, security appliances, L3 switches, core routers, etc.), the beta site must have an array of environments that satisfy multiple types of network products.

Remote Control

Developers need to check the status of an SUT when the test is ongoing, and proceed to debug as soon as a problem is found. Therefore, the beta site must provide remote control so that developers can remotely and quickly check the SUT and execute administrative functions, such as reconfiguration, firmware upgrade, reboot, and debug. For debugging, developers must be able to capture log files or network traces.

Degrees of Traffic Volume

The traffic volume in the testing area must be adaptive to the processing capability of the SUT. If the volume is much smaller than the maximum throughput of an SUT, it does not fully test the processing power of that SUT; on the contrary, if the volume is much larger, packet loss would occur, and potential product problems might not be triggered. Even though the products are of the same type and level, their processing power might differ significantly. For example, the maximum throughput of some enterprise level products reaches 500 Mb/s, but that of others might reach 1 Gb/s. The beta site must provide classified and adaptive traffic volume to fit the capability of the SUT.

Traffic Profiling

Before and during product testing in a beta site, we should thoroughly study and understand the beta site traffic, so that we can prepare the testing well and summarize comprehensive information for developers for debugging. The information includes the average throughput, maximum throughput, distribution of application protocols, average number of connections, maximum number of connections, average connection rate, and maximum connection rate, and so on.

From the point of view of network users, the most important concerns are availability and volunteering.

Availability and Volunteering

Network users are very sensitive to network quality. To increase their tolerance to network quality, we let users decide whether to join the

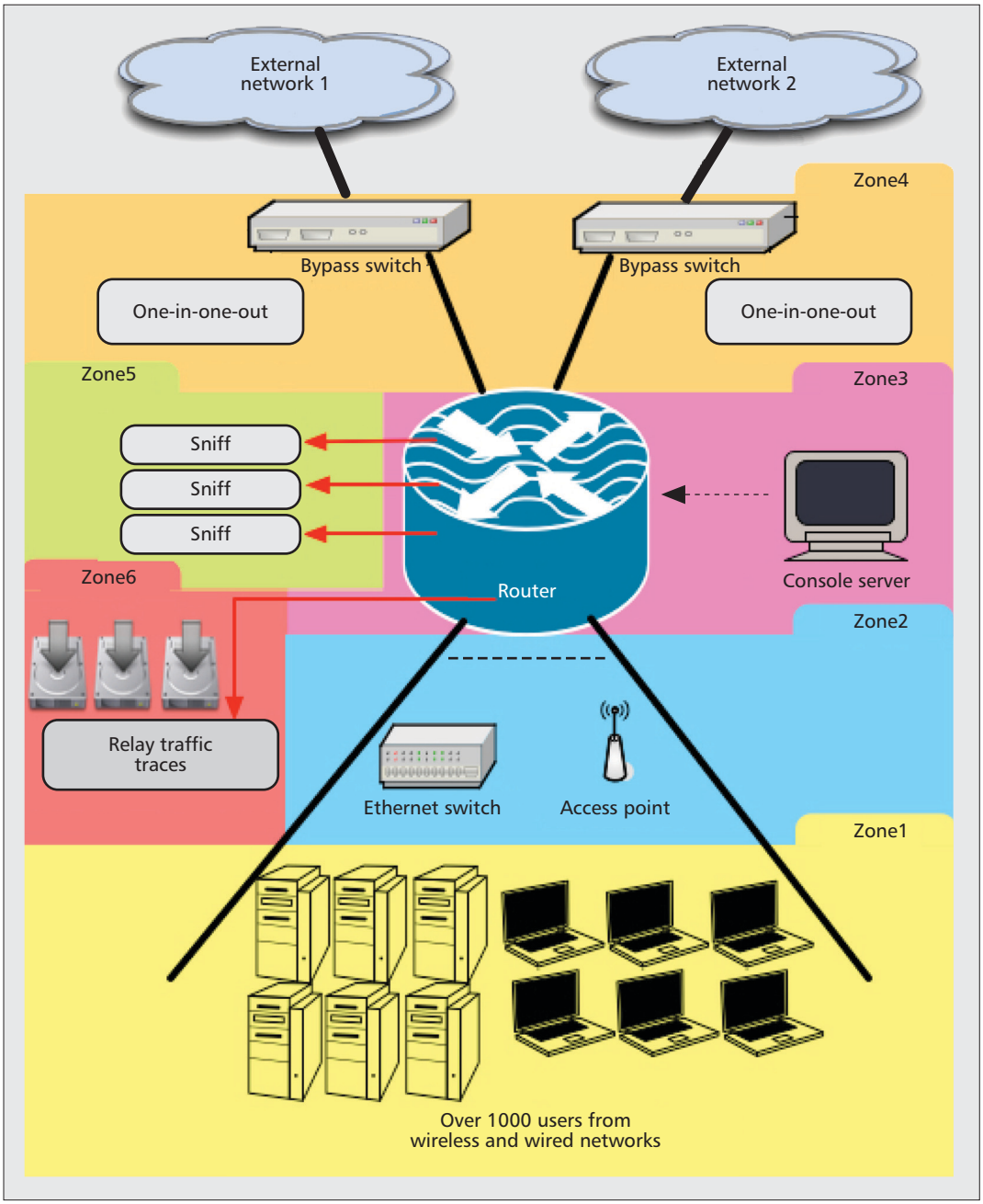


Figure 1. *Coverage and structure of the beta site.*

We use different replay tools alternately to utilize each of their specialized functions. Overall, dozens of network product types can be tested on our beta site. Despite the broad range of SUT types, some core testing principles and mechanisms apply to all of them.

beta site. Before their participation, they must clearly understand their rights and obligations, as well as the advantages and disadvantages of being on the beta site. They need to realize that the beta site might be more problematic than an ordinary operational network. The beta site is acceptable as long as problems can be solved quickly; therefore, it must support automatic failure detection, notification, and recovery to reduce troubles that annoy the users.

Architecture Designs

To Satisfy Concerns from Developers

Solutions to the First Concern: Variety of Test Zones — Figure 1 shows the beta site network topology, which provides six test zones. Users are in zone 1, where we can test software such as anti-virus and malware detection. We install the SUTs on the computers with various applications and operating systems mainly to determine if compatibility problems exist. In zone 2 we test Ethernet switches and 802.11 a/b/g/n wireless access points. Various network adapters on the user side test for interoperability problems. In zone 3 we test the stability of core routers. In zone 4, we test network security products in the transparent mode that receive traffic from one interface and transmit traffic to another interface (i.e., *one-in-one-out*) for their functionality, performance, and stability. These SUTs may include firewalls, unified threat managers (UTMs), intrusion prevention systems (IPSs), and P2P management systems. In zone 5

To encourage students to participate in the beta site, we provide incentives, protect their privacy, and facilitate the automation process to make it convenient to join the beta site. The incentives include an advanced and fast network environment.

we test products for network forensics in the *sniff* mode for their performance and stability. In zone 6 we test the stability of residential gateways with *replayed* traffic. The replay tools include tcpreplay, tomahawk, avalanche, and traffic IQ, as well as our own tools. We use different replay tools alternately to utilize each of their specialized functions. Overall, dozens of network product types can be tested on our beta site. Despite the broad range of SUT types, some core testing principles and mechanisms apply to all of them. They are discussed in the following paragraphs.

Solutions to the Second Concern: Remote Control — The beta site allows the developers to access the SUT from the Internet, such as through the Secure Shell (SSH), without physically staying at the beta site. Besides, developers can also use the Internet to access the debugging environment for capturing log files or network traces. In zone 1, we develop and deploy the *beta agent* on user machines with users' agreement. It receives remote commands to push out useful information. Because the debug information available through the SUT network port might be limited, the beta site also provides a *remote console* that developers can use to access the SUT from a console port (e.g., via RS-232). Moreover, if the SUT crashes during testing or debugging, the beta site provides *remote power* for the developers to use to reboot the SUT. Both the remote console and remote power are offered to all zones except zone 1.

Solutions to the Third Concern: Degree of Traffic Volume — We deploy an auxiliary router in zone 3, as illustrated in Fig. 2. The redirect switch redirects traffic from part of its downstream ports that we select to this router using the VLAN technology. For example, in Fig. 2, traffic passing through the left (circled) area can be redirected to the right (circled) area. The traffic to the auxiliary router will also go through the original router to the external networks. In this architecture we can select the ports to be redirected by traffic volume to adjust the amount of traffic at the link between the two routers; therefore, that link provides different degrees of traffic volume for the one-in-one-out SUTs. Furthermore, the auxiliary router can be replaced by other SUTs, such as core routers and chassis-based switches, and tested with varying traffic volume. In zone 5 the amount of traffic into the SUT can be adjusted by traffic filtering. In zone 6 the replay tools can control the *speed* in replaying traffic.

Solutions to the Fourth Concern: Traffic Profiling — The beta site utilizes various packages to profile the traffic at different protocol layers. A multirouter traffic grapher (MRTG) can provide traffic volume statistics at a given point in time. It also shows the maximum and minimum network utilization, as well as when they occur. Network top (NTOP) can analyze various portions of layer 3 and 4 protocol messages, packet sizes, transmission types, TCP/UDP port numbers, and so on, according to the protocol header fields. Some other packages can analyze the application headers and even the payload of various applications (e.g., P2P file sharing, gaming, VoIP, video streaming). Among these are *L7-filter*, *IPP2P*, *Ourmon*, and *Panabit*. After exercising these tools on the beta site, we have a general idea about its traffic profile: the bandwidth utilization is more than 1 Gb/s at the peak hour and 300 Mb/s on average. The number of maximum concurrent connections is more than 50,000, and the connection rate is about 2000/s. P2P traffic, like BitTorrent, Thunder, Gnutella, and eDonkey, dominate over others, such as Web, email, FTP, and IPTV.

To Satisfy Concerns from Network Users

Solutions to the Fifth Concern: Volunteering — To encourage students to participate in the beta site, we provide incentives, protect their privacy, and facilitate the automation process to make it convenient to join the beta site. The incentives include an advanced and fast network environment, such as Gigabit Ethernet or 802.11n WLAN, security protection, loose restrictions on management (e.g., allowing more bandwidth for P2P applications), strong problem solving support teams, and new network services such as live TV streaming. To protect the privacy of network traffic, all participants, including industrial developers and academic researchers, must sign a nondisclosure agreement (NDA). To automatically redirect network traffic to the beta site, as shown in Fig. 3, we use a *port-based* VLAN, associating the ports of the switch located in the students' dormitory (the *dorm switch*) to the beta site VLANs that the participants connect to. When the traffic with tagged VLAN identifiers is forwarded to the redirect switch, the redirect switch redirects the traffic to the beta site based on the VLAN identifiers. With VLAN, beta site administrators do not need to plug and unplug wires at Ethernet switches to connect users to the beta site.

Solutions to the Sixth Concern: Auto Failure Detection, Notification, and Recovery — When a problem occurs during product testing, we want to know it as early as possible to solve it and maintain network quality. Figure 4 illustrates how zone 4 in Fig. 1 works, where we use bypass switches for failure detection, notification and recovery. A bypass switch is a special network device with special functions, not a general-purpose router/switch. Network ports A and B on the bypass switch are connected to the external network and internal network, respectively; and monitor ports 1 and 2 are on the two sides of the SUT. When the SUT operates normally, the bypass switch is in normal mode, and the traffic path from internal to external is internal network → port B → port 1 → SUT → port 2 → port A → external network. Once the SUT has encountered a problem and the bypass switch detects it, the switch operates in the bypass mode. At this time, the traffic path becomes internal network → port B → port A → external network: the SUT is bypassed and network availability is recovered. At the same time, the bypass switch sends an email message to notify the administrators.

The bypass switch periodically sends *heartbeat* packets to check whether the SUT can forward the heartbeat packets back to the bypass switch, and this period is determined by a configurable *timer*. When the number of lost heartbeat packets exceeds a threshold decided by a configurable *counter*, the bypass mode is turned on. Moreover, we set up some PCs in zone 1 to run our own testing programs, which can imitate common network user behavior, such as browsing well-known web sites, fetching files by P2P, and watching network TV. We call these PCs *beta clients*. Beta clients can simultaneously monitor the network status and regularly report to the administrators by email. Besides the automatic methods, we establish a website for end users to post their feedback. End-user feedback is one way to report the occurrence of product defects and usually provides useful debug information.

Operational Experience

Standard Operating Procedure

Figure 5 illustrates the SOP for deploying a product in the beta site. The procedure comprises four phases: apply, install, test, and feedback. The first phase is the application to use the beta site test service. After being approved, the SUT can be installed in the second phase and tested in the third phase, and vendors offer feedback on the causes of and solutions for product defects in the fourth phase. We design three forms to record and provide necessary information during the SOP. The three forms are named BS-001, BS-002, and BS-003. BS stands for beta site. We discuss how they are used and their contents in the following sentences.

In the apply phase a developer of a participating vendor is asked to fill out a BS-001 form, which provides information on the testing environment, participant contact, and SUT. In the install phase we evaluate the suitable environment and policies for the SUT. For example, we first compare the throughput of the SUT and the traffic volume from traffic profiling. If the throughput is smaller than the traffic volume, we may turn off a few functions in the SUT to boost the throughput or lower the traffic volume as discussed. In the test phase the developer usually examines the log of the SUT or changes its configuration by remote control. This phase has six states. First is the online state, meaning that testing is in progress. If the SUT fails, it enters the failure detection and notification state, where the administrators know that problems have occurred and gather useful debug information in the BS-003 form as a reference for developers in the information gathering state. BS-003 covers a problem description, frequency, times of occurrence, and so on. After information gathering, it enters the failure recovery state. After the network recovers, the SUT may be put offline in the penalty/offline state. In this offline period the developers debug the problem using the gathered information. After solving the problem, the SUT can be put online again. In the feedback phase the developer is asked to fill out a BS-002 form for the SUT to report the fundamental causes of the problem, and its solutions. This form, as well as BS-003, can be used to obtain statistics for evaluating the effectiveness of the beta site test service. There is a database to store and update the statistics, including the TTF of defects for each SUT.

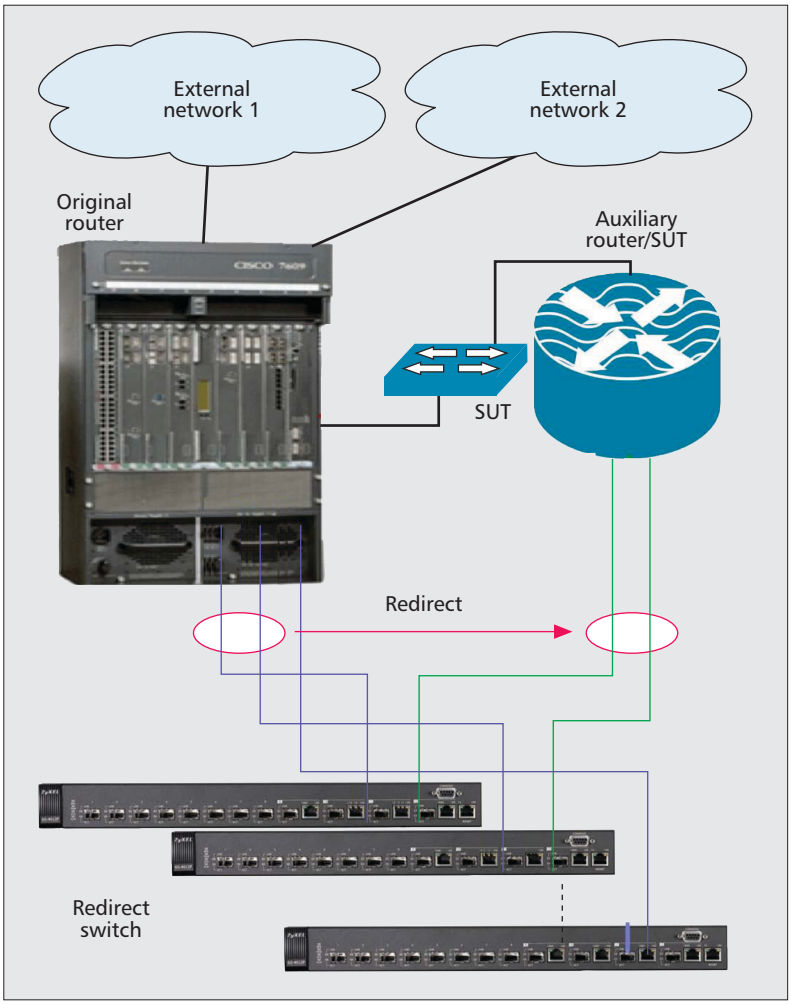


Figure 2. *Regulating traffic volume for the SUT.*

Case Studies

There are two practical case studies to illustrate possible conditions during the entire testing process. The first case is an anti-malware product that detects and removes malware. The SUT consist of two parts. One is a software agent deployed on the computer of each user; the other is a hardware appliance operating in sniff mode. The hardware appliance detects network behavior of malware programs. If it finds suspicious behavioral patterns, it notifies the particular software agent to see whether malware exists on that host. If so, the software agent removes the malware. After finishing the apply phase, we install the hardware appliance in zone 5 and the software agents in zone 1. However, after analyzing the traffic profiling, we found that the traffic volume in zone 5 was more than the SUT could sustain. This would make the SUT lose detection accuracy due to packet loss. Therefore, we used the filtering method illustrated earlier to reduce the traffic volume in zone 5. Since the beta site has 13 subnets of class C, we allowed

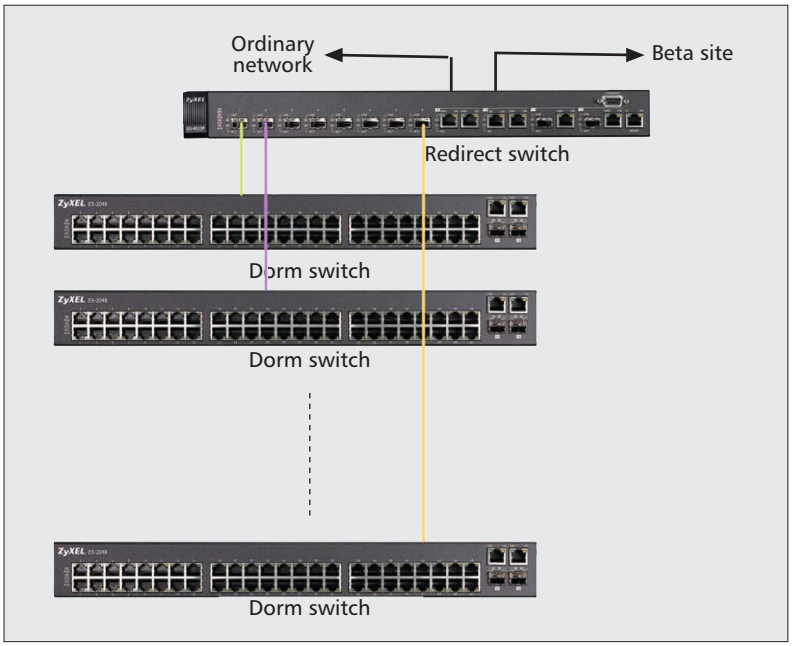


Figure 3. *Using VLANs to separate the beta site from the ordinary network.*

only the traffic from some of these subnets into the SUT.

In the test phase we discovered several product defects, including crash, memory leakage, failure of the hardware to connect with the software agent, and incompatibility with other antivirus software, the Microsoft Media Server (MMS) protocol, and some games on users' PCs. These problems were usually found through automatic notification from beta clients or end user feedback. End user feedback is also helpful to provide some debug information. The debug information was recorded in the BS-003 form as a reference for the developers. If the software agent incorrectly blocks a normal application, the application could be added to the exception list of the software agent to avoid blocking. The bypass switch will not take effect in this case since the SUT is not installed in zone 4. Because failure recovery was fast for incompatibility defects, we did not have to put the SUT offline. After solving the false positives, the applications involved in the problem can be removed from the exception list for further testing.

The second case is a modular switch, which is a layer 3 switch with four modules; each module has 48 ports. In the install phase we like to gradually increase the traffic volume for testing. The developers would then know whether a particular defect is associated with a specific traffic volume level; this also reduced the influence of the SUT on network quality since it was new and its stability unknown. We set up the SUT in zone 2, replacing the auxiliary router shown in Fig. 2, so we could configure and select the traffic from the redirect switch to the SUT. If the SUT was stable, we could gradually increase the traffic volume. In the test phase the problems included failure to communicate with each other among network ports of the same module, failure to connect to the Internet if the medium access control (MAC) address on a PC changed, and failure to be online after rebooting the SUT. The method of failure recovery was to redirect the traffic to the original router.

Lessons

We gained much valuable experience in the process of setting up and operating the beta site. Here are four specific lessons:

- We can use the buffer on the remote console server to store console messages from the SUT. This is helpful for storing more debug information.
- We should redirect the traffic automatically with VLAN. If the redirection is manual, participants are blocked from accessing the network for a while.
- When using the bypass switch, we must configure the proper *timer* and *counter* according to the properties of the SUT and user responses. For slow SUTs, the two thresholds should be set higher or the SUTs might be bypassed even when no problem occurs; however, for users, if the two thresholds are set too high, it might be too late to recover network connectivity.
- Information gathering must be done *before* putting the SUT offline, or important debug information is lost.

Top Product Defects

During the one year of operation, 42 products from 13 vendors were tested in the beta site. Based on the statistics from the BS-002 and BS-003 forms, we found hundreds of critical defects, which would have a significant impact on users and take much time for developers to solve. Most of these are compatibility and stability problems. For example, the anti-malware product testing described earlier saw many compatibility defects. Users reported that many Web TV applications (e.g., ipobar, TVAnt, Vigor) were not usable. After deep analysis, we found the problems resulted from defects of the software agent. The software agent conflicted with the multicast streaming process in the operating systems. Besides Web TV, the software agent also has compatibility problems with many security applications (e.g., Comodo firewall, Avira AntiVir) and game applications (e.g., Gameguard). These applications have implemented a protection mechanism so that the software agent cannot hook them with dynamically linked library (DLL) injection.

Another example is a defect in the modular switch testing also described earlier. A user reports that he has two PCs, with one running Windows XP and the other running Windows Vista. Since only one public IP address is allocated to each user (according to policy), the two PCs must share one public IP address. He uses only one PC at a time to access the Internet, but the PC running Windows Vista can seldom access the Internet normally. After further analysis, we find that the behavior of gratuitous Address Resolution Protocol (ARP) requests differ in the two OSs. In Windows XP the Sender Protocol Address (SPA) is equal to the Target

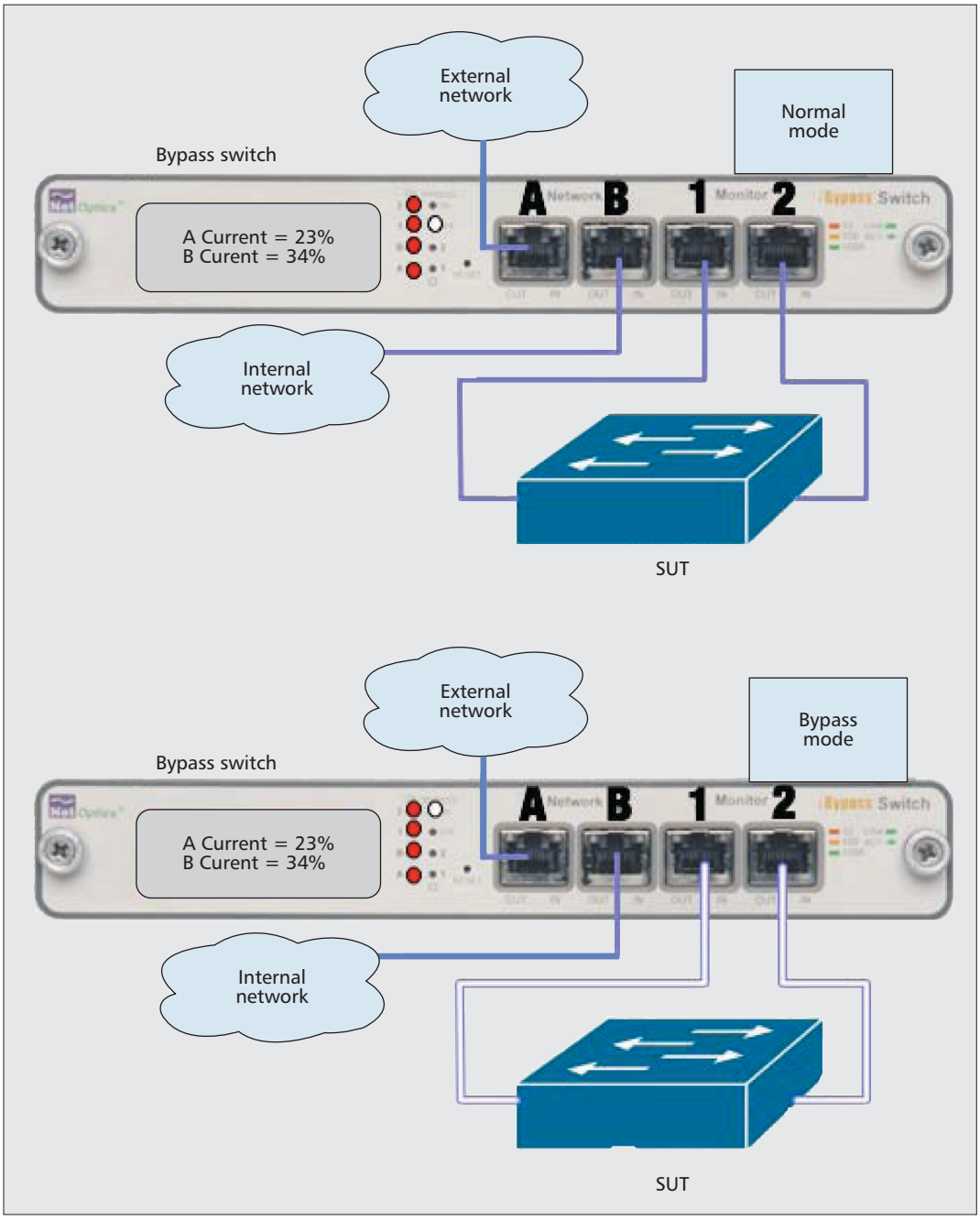


Figure 4. *Bypass switch behavior: normal vs. bypass mode.*

Based on feedback from developers and users in the study, we plan to recruit more users to diversify the network traffic in the beta site, and add more testing environments and features, such as 10 gigabit Ethernet, 802.11n WLAN, IPv6, virtual router redundancy protocol (VRRP), and multicast in the beta site.

Protocol Address (TPA), but in Windows Vista the SPA is always 0.0.0.0. Windows Vista behaves in this fashion to solve the problem with IPv4 address conflict detection. However, the SUT does not know the new design consideration in Windows Vista and will not update its ARP cache when the user switches to run the PC with Windows Vista. The SUT still uses the MAC address of the PC running Windows XP for its mapping; thus, no packets can be forwarded to the PC running Windows Vista.

Based on these two examples, we can see the diversity and complexity of the beta site testing compared with conventional laboratory testing. The possible test cases are well beyond what laboratory testing could cover. Testing products on the beta site can find and fix product defects before it is too late. If the product defects are found by customers, the cost is not only the money but the company's reputation. As for network operators, they can utilize the beta site to try some new technologies before extensive deployment.

As illustrated in Fig. 6a, we classify product defects into four types: stability, performance, functionality, and compatibility. Many products have stability defects (25/42) and few products have functionality defects (6/42), meaning that laboratory testing can help a lot with functionality, and field testing is good at stimulating stability defects. According to the ratio of defects per product, we can say that the beta site, with its great diversity, is efficient in finding compatibility defects (81/12).

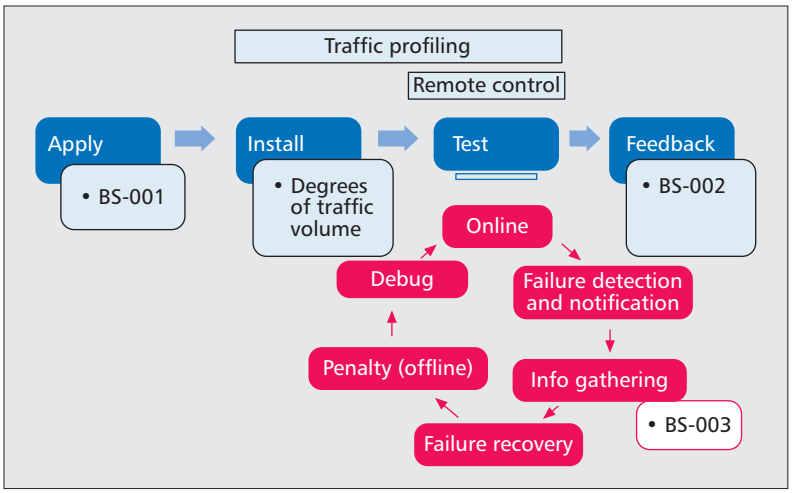


Figure 5. *SOP for testing in the beta site.*

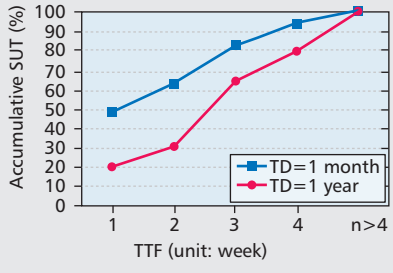
Figure 6b presents the correlation between TTF and its corresponding percentage of products. When the testing duration (TD) is one month, nearly 50 percent of the products have defects that occur in the first week (i.e., TTF ≦ 7 days), and the convergence ratio is only 7 percent. When the TD is extended to one year, the percentage of products with defects found in the first week degrades to 20 percent, and the convergence ratio is improved to 20 percent. The numerical results indicate that the product quality of most SUTs in a real-world environment has a lot of room for improvement. If we further observe the relationship of TTF and TD, we can find an interesting result in Fig. 6c: the curve increases monotonically. This is the average TTF for all test rounds of all SUTs. This relationship means that the beta site service is very effective in improving product quality. We choose four weeks as a unit of TD in Fig. 6c since, according to the information of Fig. 6b, most products show a defect within four weeks. Thus, products also have their software, firmware, or hardware updated within four weeks. Considering both marketing timing and product quality, one month is our suggested minimum TD for the low-end and short life cycle, such as the SOHO router and access point. However, we recommend the TD of one year for high-end and long-life-cycle products, such as core switch/routers and security appliances.

Conclusion and Future Work

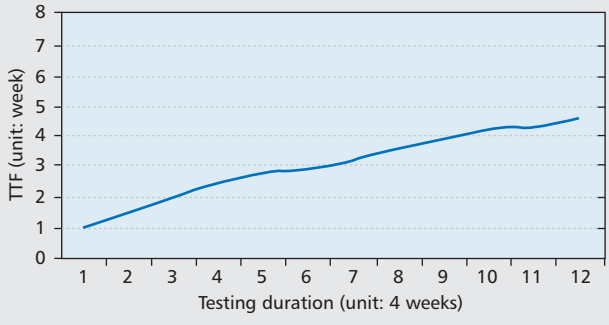
We applied six mechanisms to the beta site to satisfy and balance the concerns from product developers and network users. Beta site testing identified hundreds of critical product defects during one year. From the TTF analysis, we learn that this service is very helpful for improving product quality; however, we also find that the convergence ratio is only 20 percent after a year of testing, meaning that only a few network products can operate normally in a real environment over a long period of time. This can be improved by more active participation in beta site testing on the part of vendors. On the other hand, the number of students participating in the beta site is quite stable (about 1300 persons), meaning that the network quality of the beta site is acceptable. Therefore, the beta site test service presented in this article strikes an effective balance between needs of developers and users. Based on feedback from developers and users in the study, we plan to recruit more users to diversify the network traffic in the beta site, and add more testing environments and features, such as 10 gigabit Ethernet, 802.11n WLAN, IPv6, Virtual Router Redundancy Protocol (VRRP), and multicast, in the beta site.

Types of defects	Stability	Performance	Functionality	Compatibility
Number of defects (152)	38	23	10	81
Number of products (42)	25	9	6	12
Defects per product	1.52	2.875	0.43	6.75

(a)



(b)



(c)

Figure 6. *Statistics on defects and time to fail: a) four types of defects; b) accumulative SUT vs. TTF; c) TTF vs. TD.*

REFERENCES

[1] R. G. Cooper and E. J. Kleinschmidt, "An Investigation into the New Product Process: Steps, Deficiencies, and Impact," *J. Product Innovation Management*, vol. 3, 1986, pp. 71–85.

[2] M. K. Daskalantonakis, "A Practical View of Software Management and Implementation Experiences within Motorola," *IEEE Trans. Software Eng.*, vol. 18, no. 11, 1992, pp. 998–1009.

[3] H. Zhu, P. A. V. Hall, and J. H. R. May, "Software Unit Test Coverage and Adequacy," *ACM Comp. Surveys*, vol. 29, no. 4, 1997, pp. 366–427.

[4] Y. K. Malaiya *et al.*, "Software Reliability Growth with Test Coverage," *IEEE Trans. Reliability*, vol. 51, no. 4, Dec. 2002.

[5] T. Miyachi, K. Chinen, and Y. Shinoda, "Automatic Configuration and Execution of Internet Experiments on an Actual Node-Based Testbed," *Proc. Testbeds Research Infrastructures Development Net. Communities*, Feb. 2005, pp. 274–82.

[6] J. Nakata *et al.*, "StarBED2: Large-Scale, Realistic and Real-Time Testbed for Ubiquitous Networks," *Proc. Testbeds Research Infrastructures Development Net. Communities*, May 2007, pp. 1–7.

[7] A. Bavier *et al.*, "In VINI Veritas: Realistic and Controlled Network Experimentation," *Proc. Conf. Apps., Tech., Architectures, Protocols Comp. Commun.*, Pisa, Italy, Sept. 2006.

[8] K. V. Vishwanath and A. Vahdat, "Realistic and Responsive Network Traffic Generation," *Proc. Conf. Apps., Tech., Architectures, Protocols Comp. Commun.*, Pisa, Italy, Sept. 2006.

[9] S. Antonatos, K. G. Anagnostakis, and E. P. Markatos, "Generating Realistic Workloads for Network Intrusion Detection Systems," *Proc. 4th Int'l. Wksp. Software Performance*, Redwood Shores, CA, Jan. 14–16, 2004.

[10] F. Schneider, J. Wallerich, and A Feldmann, "Packet Capture in 10-Gigabit Ethernet Environments using Contemporary Commodity Hardware," *Passive Active Measurement Conf.*, Apr. 2007.

[11] S. Kornexl *et al.*, "Building a Time Machine for Efficient Recording and Retrieval of High-Volume Network Traffic," *Proc. ACM IMC*, Oct. 2005.

[12] N. Weaver *et al.*, "Protocol-Independent Adaptive Replay of Application Dialog," *Proc. 13th NDSS*, San Diego, CA, Feb. 2006.

[13] J. Newsome *et al.*, "Replayer: Automatic Protocol Replay by Binary Analysis," *Proc. 13th ACM CCS*, Alexandria, VA, Oct. 30–Nov. 3, 2006.

[14] Y.-C. Cheng *et al.*, "Monkey See, Monkey Do: A Tool for TCP Tracing and Replaying," *Proc. USENIX Annual Tech. Conf.*, June 2004.

[15] CAIDA: The Cooperative Association for Internet Data Analysis; http://www.caida.org/.

BIOGARPHIES

YING-DAR LIN (ydlin@cs.nctu.edu.tw) joined the faculty of the Department of Computer Science at National Chiao Tung University (NCTU), Hsinchu, Taiwan, in August 1993 and has been a professor since 1999. He is the founder and director of the Network Benchmarking Lab (NBL), which reviews the functionality, performance, conformance, interoperability, and stability of networking products ranging from switches, routers, and WLAN to security and VoIP; http://www.cs.nctu.edu.tw/~ydlin.

I-WEI CHEN (iwchen@nbl.org.tw/iwchen@cs.nctu.edu.tw) is the executive director of NBL at National Chiao Tung University. He received B.S. and M.S. degrees in computer and information science from National Chiao Tung University. He joined NBL in 2003. At NBL he is engaged in development of testing technologies for network and communication devices. He is especially interested in technologies using real-world network traffic to test products.

PO-CHING LIN (pclin@cs.ccu.edu.tw) received his B.S. degree in computer and information education from National Taiwan Normal University, Taipei, in 1995, and M.S. and Ph.D. degrees in computer science from National Chiao Tung University in 2001 and 2008, respectively. He joined the faculty of the Department of Computer and Information Science, National Chung Cheng University (CCU), Chiayi, Taiwan, in August 2009, where he is currently an assistant professor. His research interests include network security, content networking, and algorithm designing.

CHANG-SHENG CHEN (cschen@cc.nctu.edu.tw) is an associate professor at the Information Technology Service Center, National Chiao Tung University. He received his B.S.E.E. degree from National Taiwan University, and M.S. and Ph.D. degrees from the Department of Computer and Information Science, National Chiao Tung University. His current research interests include Internet-based applications, knowledge engineering, and network intrusion detection.

CHUN-HUNG HSU (chhsu@nbl.org.tw) is a manager at NBL, National Chiao Tung University. He received his M.S. degree from the Department of Computer Science and Information Engineering, National Yunlin University of Science & Technology. His current job focuses on developing test methodologies with real-world network traffic to test Ethernet switches, routers, and WiFi interconnected devices.