PlanetLab und Co. - Vergleich von Test- und Emulationsumgebungen für verteilte Systeme

Florian Weidner

Technische Universität Dresden

Zusammenfassung. Die steigende Zahl von Anwendungen, die auf verteilten Systemen betrieben werden - sei es Business- oder private Software - steigt zunehmend. Ebenso werden Kommunikations- und Übertragungsprotokolle ständig neu- und weiterentwickelt. Diese Entwicklung fordert, deren Funktion vor der Veröffentlichung ausgiebig unter möglichst realen Bedingungen zu testen. Im Folgenden werden verschiedene Emulationsumgebungen auf ihre Features, Möglichkeiten und ihre Kerneinsatzgebiete unter dem Aspekt der Netzwerkemulation untersucht sowie abschließend einem Vergleich unterzogen.

1 Einleitung

Emulationsumgebungen können einerseits wie PlanetLab [10] durch Zusammenschlüsse von Testnetzwerken verschiedener Forschungseinrichtungen entstehen. Ebenso ist es möglich, dass einzelne Institute eine Infrastruktur aufbauen und diese dann Außenstehenden zur Verfügung stellen. Weiterhin können sich, wie bei OneLab[12] geschehen, verschiedene Betreiber von Emulationsumgebungen unter einer Dachorganisation zusammenschließen, um eine möglichst gute Organisation sowie Verfügbarkeit all dieser Systeme zu erreichen. Alle diese verschiedenen Herangehensweisen haben das Ziel, Entwicklern von verteilten Systemen und Protokollen (z.b. Cloud Solutions) die Möglichkeit zu bieten, ihre Anwendung vor dem praktischen Einsatz unter der Nutzung von Netzwerkemulation zu testen.

2 Simulation vs. Emulation

Bei der Netzwerksimulation steht, im Gegensatz zur Netzwerkemulation, keine gesonderte Hardware zur Verfügung. Die Umgebung wird komplett mit Parametern und Algorithmen modelliert. Der Kernaspekt der Emulation hingegen ist es, die Lücke zwischen der Testumgebung und der Realität dadurch zu Verringern, dass ein Teil der Testumgebung auf real existierende Hardware verlagert wird. Die Simulation bietet eine sichere Reproduzierbarkeit, die Emulation hingegen zielt darauf ab, möglichst reale Bedingungen anzubieten [5].

3 Emulationsumgebungen

Im Folgenden Teil werden ausgewählte Emulationsumgebungen hinsichtlich folgender Kriterien vorgestellt: Zugangsvoraussetzungen, Anpassungsmöglichkeiten, kontrollierbare Netzwerkeigenschaften, Existenz vordefinierter Parametersets, Reproduzierbarkeit und Möglichkeit der Emulation von mobilen (drahtlosen) Netzwerken.

3.1 PlanetLab Central

Das populäre PlanetLab mit seinen 1116 Knoten an 544 Standorten bietet im Gegensatz zu den anderen hier vorgestellten Emulationsumgebungen die meisten Ressourcen. PlanetLabs Hauptabsicht ist es, Testumgebung für Overlay-Netzwerke zu sein. Es soll Anwendern ermöglichen, verschiedene Services (zum Beispiel File-Sharing-, Netzwerk-Speicher- und Routingservices) zu testen. Die gebotenen Bedingungen sind sehr realistisch und lassen Rückschlüsse zu, wie gut der Service sich im Internet verhalten würde. PlanetLab kann nicht nur für abschließende Tests genutzt werden, sondern auch um einen reibungslosen Entwicklungsprozess von einer prototypischen Anwendung hin zu einer stabilen Version über mehrere Iterationen zu ermöglichen [10]. Es bleibt zu sagen, dass PlanetLab nicht für die (Drahtlos-)Netzwerkemulation geeignet ist, da kaum Reproduzierbarkeit oder Anpassungsmöglichkeiten geboten werden. Jedoch ist das Testen und Entwickeln von Anwendungen, Protokollen und Services unter überaus realitätsnahen Bedingungen sehr gut möglich.

3.2 Emulab

Der Begriff Emulab bezeichnet einerseits die Netzwerkemulationssoftware als auch die ursprüngliche Einrichtung an der University of Utah. Es existieren weitere Institutionen die Emulabs anbieten und es kann ein privates Emulab (auf eigener Hardware) installiert werden. Folgende Ausführungen beziehen sich exemplarisch auf das Emulab an der University of Utah. Dieses kann von Forschungseinrichtungen, Universitäten sowie Firmen in und außerhalb der Vereinigten Staaten kostenfrei, jedoch nur nach vorheriger Anmeldung genutzt werden. Nach der Registrierung eines Projekts über die Emulab-Homepage [11] ist es möglich, das Forschungsprojekt durchzuführen bzw. zu starten. Die Steuerung erfolgt über ein Webinterface. Spezielle Zugangsvoraussetzungen bestehen nicht. Die Durchführung eines Experiments bzw. eines Testlaufs mit Emulab erfordert die Definition einer statischen sowie einer dynamischen Konfiguration. Der statische Teil beschreibt die Topologie mit ihren Knoten, Verbindungen und Netzwerkeigenschaften. Bezüglich der Knoten werden unter anderem das Betriebssystem sowie zu ladende Pakete genannt. Verbindungen können mit vordefinierten Parametersets (z.B. LAN, Duplex) belegt werden. Die einzelnen Parameter (Bandbreite, Paketsendefrequenz, Verlust, Delay, u.a.) können ebenfalls selbst festgelegt werden. Nachdem die gewünschte Topologie mit ihren Knoten

und deren Verbindungen definiert wurde, muss der dynamische Teil des Experiments deklariert werden. Dieser spezifiert auszuführende Skripte (Aktionen, die während des Testlaufs auftreten sollen, wie z.B. Paketverlust, Nodefehler o.ä.). Traffic-Generatoren müssen hier nicht extra definiert werden - Emulab erstellt diese auf Wunsch anhand der im statischen Teil genannten Parametern (Sendefrequenz, Bandbreite, usw.).
Um mit Emulab Anwendungen und Protokolle in und für Drahtlosnetzwerke zu testen, muss die Definition im statischen Teil angepasst werden. Hier existieren jedoch Einschränkungen bezüglich des Traffics und der Kanäle (Vermeiden von großen Datenmengen im Allgemeinen sowie speziell von Datenmengen die keine Antwort erfordern). Es können die Protokolle 802.11 a/b/g genutzt werden. Zudem können Parameter wie mode (Master, Managed, Adhoc oder Monitor), freq (Sendekanal oder Übertragungsfrequenz), rate (Übertragungsrate), txpower (Sendeleistung), rts (minimale Paketgröße) und frag (maximale Paketgröße, größere Pakete werden fragmentiert) vordefiniert, sowie im laufenden Experiment verändert werden. Nach der Definition der statischen und dynamischen Konfiguration, können diese via Emulab angewandt werden. Das System reserviert nun die entsprechenden physikalischen Ressourcen und führt das Experiment entsprechend durch (vgl. [3], [19].
Nachteilig ist das Fehlen von weiteren Fehlerquellen (abgesehen von der Verlustrate). Diese müssten per Hand im dynamischen Teil definiert werden und sind nicht nativ implementiert. Ebenfalls wird der moderne 802.11n-Standard sowie die Emulation von Mobilfunknetzen nicht unterstützt.

3.3 OneLab (PlanetLab Europe und NITOS)

Mit der Registrierung bei OneLab ist es möglich, vier verschiedene Testumgebungen zu nutzen. Einerseits ist es möglich, PlanetLab Europe zu nutzen, das die gleichen Bedienungen wie PlanetLab Central (vgl. 3.1) sowie dessen Konzept und Architektur nutzt. Das heißt, auch hier wird PlanetLabOS eingesetzt. Die europäische Version bietet neben den Features von PlanetLab Central einige Einstellungsmöglichkeiten an, die für das Testen hilfreich sind. Dazu gehören vor allem das Konfigurieren verschiedener Parameter vor und während des Testlaufs. Es können verschiedene Scheduling-Verfahren ausgewählt werden und die Bandbreite kann geregelt werden. Mittels spezieller Software [8] kann auch hier das Verhalten von mobilen Netzwerken emuliert werden. So können unter nahezu realen Netzwerk-Bedingungen große Anwendungen und Protokolle auf vielen Nodes getestet werden. Zusätzlich zu diesen Parametern ist es sowohl bei drahtlosen als auch drahtgebundenen Experimenten möglich, Delays und Verlustraten zu definieren. Neben der PlanetLab-Software kann noch das OMF-Framework genutzt werden [9]. Es bietet ähnliche Möglichkeiten bezüglich dem Kontrollieren, Durchführen und Evaluieren von Experimenten wie die von PlanetLab entwickelte Software. Weiterhin wird über OneLab die Nutzung von NITOS möglich, einer Einrichtung, die das Testen und Evaluieren von Drahtlosnetzwerken ermöglicht. NITOS bietet dem Nutzer eine Testumgebung für Drahtlosapplikationen die hohe Reproduzierbarkeit verspricht. Die Kommunikation erfolgt hier über

echte Drahtlos-Hardware. Auch hier kommt die bereits erwähnte OMF-Software zur Durchführung und Kontrolle der Experimente zum Einsatz. Mit ihr kann das Experiment hinsichtlich verschiedener Parameter konfiguriert werden. Eine Auswahl dieser sind gewünschte Störgeräusche, die Bitrate (unabhängig vom Protokoll), welcher Übertragungskanal genutzt werden soll, Transmission Power (zum Beispiel um Tests im Energiesparmodus zu ermöglichen) sowie das zu verwendende Protokoll (es werden 802.11 a, b und g unterstützt). Der neueste Standard 802.11n wird von der momentanen Hardware nicht unterstützt. Auch hier ist zu bemerken, das verschiedene Fehlerklassen (Corruption, Duplication) nicht angeboten werden. Als besonderen Punkt bietet NITOS 15 Webcams an, mit denen Tests ausgeführt werden können, die in den Bereich Video fallen. Für Experimente mit Sensoren bietet die NITOS-Infrastruktur zehn Temperatursensoren an, die genutzt werden können. Neben NITOS und PlanetLab Central, ist die Nutzung von ETOMIC und DIMES möglich. Sie konzentrieren sich auf das Analysieren und Messen des Traffics in verschiedenen Topologien und Netzwerken und sollen hier nicht weiter behandelt werden.
Leider ist die Nutzung all dieser Umgebungen für eigene Projekte an Bedingungen geknüpft. Wenn eine Einrichtung Rechner zum Verbund hinzufügt, kann sie ohne Probleme Projekte im System ausführen, hat aber dadurch administrative Verpflichtungen bezüglich des Gesamtprojekts [12].

3.4 ORBIT

Die Umgebung ORBIT (Open Access Research Testbed for Next-Generation Wireless Networks) hat sich ebenfalls auf das Durchführen von reproduzierbaren Drahtlos-Netzwerk-Experimenten spezialisiert. Die Nutzung ist kostenfrei und generell für jeden im Bereich Forschung und Lehre möglich. Für Studenten wird empfohlen, im Voraus die Empfehlung eines Professors einzuholen und erst dann ihr Experiment über die Homepage anzumelden.
Die Durchführung ist nur während vorher reservierten Zeiten möglich. Nach Ablauf, ist der Zugriff auf die Knoten nicht mehr möglich. Um ein Experiment zu starten muss ein sogenanntes Script erstellt werden, welches die Durchführung kontrolliert und steuert. In ihm wird die Topologie, deren Parameter und die Knoten mit ihrer Konfiguration definiert.
Die Nodes sind unabhängig voneinander konfigurierbar und können mit eigenem Betriebssystem oder Image ausgestattet werden. Benötigte Software kann bei Bedarf nachgeladen werden. Die Architektur von ORBIT besteht aus zwei Komponenten. Einerseits kann eine geschützte, definierte Umgebung verwendet werden, die aus 400 untereinander verbundenen 802.11x-Routern besteht, die als 2-dimensionales Array angeordnet sind. Hier kann im Experiment quantitativ getestet werden und es ist eine hoher Grad an Reproduzierbarkeit gegeben. Die Topologie dieser Router ist flexibel und dynamisch veränderbar. Nachdem die Anwendung als stabil gilt, kann man sie auf das sogenannte „Field Trial Network" migrieren. Hier steht dem Nutzer ein konfigurierbares Netzwerk aus 802.11x und 3G-Stationen zur Verfügung, die in einem Gebiet von 10 Quadratkilometern verteilt sind. Diese Umgebung bietet nahezu reale Bedingungen, da sich

alle Knoten in bewohntem Areal befinden. Alle verwendeten WLAN-Karten unterstützen momentan nur die Standards 802.11 a/b/g. Eine Reihe von sogenannte Interferenz-Generatoren, die Störungen im Netzwerk simulieren (zum Beispiel Mikrowellen), sind ebenfalls Bestandteil des ORBIT-Netzwerkes. Mit ihnen ist es zwar nicht möglich,Verbindungen gezielt zu beeinflussen, es kann jedoch eine Umgebung simuliert werden, die über schlechte Bedingungen verfügt. Mit einer Reihe von „off-grid"-Nodes (initial nicht mit dem Netzwerk verbundene Knoten) können auch während dem Experiment zufällige Topologien simuliert werden. Kombiniert mit der Möglichkeit, Pakete von einzelnen Stationen zu ignorieren lässt sich dieser Effekt noch verstärken. Auch Handover und Disconnections können mit dieser Vorgehensweise simuliert werden.

3.5 VINI

Die Emulationsumgebung VINI (VIrtual Network Infrastructure) bietet Forschern ebenfalls die Möglichkeit, ihre Services und Protokolle unter realistischen Bedingungen sowie mit ausgeprägten Kontrollmöglichkeiten zu evaluieren. Es ist ein sogenanntes „Private PlanetLab" und benutzt somit die Softwarepakete und Hardware, die bei PlanetLab Central verwendet werden (auf 26 Rechner an 14 Standorten). Die zur Verfügung gestellte Software wurden durch die Entwickler von VINI um Funktionen erweitert und soll in naher Zukunft auch im gesamten PlanetLab Central eingesetzt werden. VINI ist weniger für das Testen von Anwendungen entwickelt worden, sondern zum Evaluieren von Netzwerkprotokollen und -services (z.B. Routingprotokolle).
Durch die enge Verschränkung mit PlanetLab sind die Zugangsvorraussetzungen zur Nutzung von VINI gleich. Jeder, der berechtigt ist, PlanetLab zu nutzen, hat auch die Möglichkeit, seine Experimente mit VINI durchzuführen.
Der Grundgedanke von VINI ist der Spagat zwischen Realität und Kontrolle während des gesamten Testprozesses. Da jedes Experiment unterschiedliche Anforderungen an diese beiden Aspekte stellt, ist darauf hingearbeitet worden, dem Nutzer möglichst viel Kontrolle über diese beiden Variablen zu geben. Sei es einerseits die Steuerung von Netzwerk Events (Verbindungsfehler, Flash Crowds) oder ein Ändern des Netzwerkverhaltens (Traffic, Topologie). Die Justierung dieser Parameter kann vorher geplant werden, oder auch während des Testlaufs manuell steuerbar, um möglichst viel Kontrolle und auch Reproduzierbarkeit zu erreichen. Ebenso ist es sehr einfach möglich, Experimente in einer exakten Nachbildung des Abilene Netzwerks [7] durchzuführen, um möglichst reale Bedingungen zu erhalten. Die Reproduzierbarkeit wird einerseits durch die von PlanetLab garantierte Isolation der Experimente und andererseits durch eine noch weiter verschärfte Trennung und Verteilung der Ressourcen durch VINI garantiert.
Topologien und Routing können frei definiert werden. Hierbei besteht die Möglichkeit, eigene Routingprotokolle oder aber mitgelieferte verwendet werden. Ebenso ist die parallele Ausführung von Experimente erlaubt. Generell bietet

VINI dem Nutzer die Möglichkeit, realen Traffic während seiner Tests zu verwenden. Hierfür kann zum Beispiel beim PlanetLab-Vini auch Traffic von anderen laufenden Experimenten in ein Experiment geleitet werden. Herkömmliche Traffic-Generatoren können dennoch mit ihren Parametern (z.B. Chunckgröße, Bitrate) definiert und verwendet werden. Um eine hohe Akzeptanz in der Entwicklercommunity zu erreichen, wird die Konfiguration der Topologie sowie der Ressourcen mit der gleichen Syntax wie Emulab durchgeführt, um eine einfache Portierierung von Emulab-Experimenten hin zu VINI zu ermöglichen.
Protokolle, Services und Anwendungen für mobile und drahtlose Netzwerke können mit VINI nicht nativ getestet werden. Zwar kann vom Entwickler das Netzwerkverhalten der drahtgebundenen Verbindungen per Scripts simuliert werden, dies ist jedoch aufwändig und im Allgemeinen nicht zu empfehlen, da dann zwar Reproduzierbarkeit gegeben ist, realitätsnahe Bedingungen aber, sind schwer zu erreichen.

3.6 WhyNET

Am Ende der Reihe der vorgestellten Emulationsumgebungen wird hier noch WhyNET evaluiert. Diese Emulations- bzw. Testumgebung ist ausschließlich für das Testen, Evaluieren und Entwickelung von Drahtlosanwendungen entwickelt worden und integriert verschiedene bereits existierende Softwarelösungen. WhyNET sticht dadurch hervor, dass es ein „Wireless Hybrid Network Testbed" ist, genauer gesagt versucht es, Simulation, Emulation und physikalische Prozesse zu verbinden und die jeweiligen Vorteile dieser Komponenten perfekt zu nutzen. Das Ziel ist also, die realitätsnahen Verhältnisse mit realen phsyikalischen Komponenten zur Verfügung zu stellen, gewisse Effekte (wie Routingverfahren) nutzerspezifisch möglichst realitätsnah zu emulieren und in besonderen Fällen die gute Reproduzierbarkeit durch Simulation zu nutzen.
Um dies alles zu erreichen stellt WhyNET die Infrastruktur zur Verfügung. Ohne hohe Hürden wird die von der UCLA [?] betriebene Umgebung jedem interessierten Nutzer nach Anfrage und Genehmigung zur Verfügung gestellt - allerdings ist hier nur ein eingeschränkter Remotezugang erhältlich. Dieser beinhaltet zusätzlich zu dem Nutzungsrecht Software, Tools und vorhandene Modelle verschiedener bereits durchgeführter Experimente die genutzt werden können, um den Arbeitsaufwand möglichst niedrig zu halten. Als weiteres Feature bietet WhyNET dem Nutzer verschiedene Interfaces an, um bestehende und häufig genutzte Emulations- und Simulationsumgebungen anzubinden. Als Beispiel können hier TWINE [20] oder Qualnet [6] genannt werden. Im Folgenden werden wir kurz Features der beiden vorstellen. TWINE, als spezielle Emulationsumgebung für Drahtlosapplikationen entwickelt, bietet dem Nutzer unter anderem die Möglichkeit, die Transmission Power, Delay und die maximale Bandbreite zu steuern. Qualnet, ebenfalls spezialisiert für Drahtlosanwendungen, ermöglicht es, Verlustrate, Paketgröße oder aber verschiedene Energiemodi für die Sensoren zu regeln. Diese beiden Frameworks sind überaus mächtig und bieten noch weitere zusätzliche Möglichkeiten. Qualnet allerdings ist nur kommerziell verfügbar.
WhyNET zielt darauf ab, den Einfluss neuer Technologien auf die Performance

von bestehenden Anwendungen zu ermitteln und hierfür skalierbare und realistische Szenarien einzusetzen. Hierfür nutzt es, wie eingangs genannt, geographisch verteilte Testumgebungen mit integrierten Emulations- und Multimode-Simulationsumgebungen.
Als Testnetzwerke bietet WhyNET Hardware für 3G-CDMA-Netzwerke sowie Breit- und Schmalbandnetzwerke. Zudem können Software-Defined-Radio Umgebungen genutzt werden. Hardware zum Testen für mobile Adhoc-Netze ist ebenfalls verfügbar. Die Entwickler des WhyNETs erstellten die Software speziell zu dem Zweck, Sicherheitsprotokolle, energieeffiziente Netzwerke und eine optimierte cross-layer Ausführung zu ermöglichen bzw. die Entwicklung dieser zu unterstützen.

4 Fazit

Die hier vorgestellten Emulationsumgebungen eignen sich mehr oder weniger gut zum Testen von Anwendungen im Netzwerk. Viele haben sich auf das Analysieren von Protokollen und nicht auf das Testen von Endnutzeranwendungen spezialisiert. Für solche ist Emulab am besten geeignet, da es dem Nutzer erlaubt, eigene Pakete in das System zu laden. PlanetLab bietet diese Funktion auch, aber stellt wenige Anpassungsmöglichkeiten sowie keine Möglichkeit zur Drahtlosemulation zur Verfügung. Das Testen von mobilen Umgebungen ist leider, außer bei ORBIT, nicht gut ausgeprägt. Fehler sind kaum ohne Schwierigkeiten induzierbar (z.B. Jitter, Reordering). Das Fehlen von vordefinierten Parametersets (außer bei Emulab und WhyNET) verlängert den Testprozess unnötig, da gebräuchliche Konfigurationen (z.B. DSL6000, DSL2000) manuell spezifiziert werden müssen. ORBIT empfiehlt sich als Emulationsumgebung für drahtlos- und mobile Umgebungen, VINI zur Emulation von drahtgebundenen Anwendungen.
Eine zusammenfassende Übersicht ist in Tabelle 1 zu finden.

Tabelle 1. Zusammenfassung der Ergenisse

	PlanetLab	EmuLab	OneLab	ORBIT	VINI	WhyNET
Drahtlose Umgebungen	-	+	+	+	-	+
Mobile Umgebungen	-	+	+	+	-	+
Vordefinierte Parametersets	-	+	-	-	-	+
Einfacher Zugang	+	+	o	+	+	-
Reproduzierbarkeit	-	+	+	+	+	+
Reale Bedingungen	-	+	+	o	+	+
Anpassungsmöglichkeiten	+	-	+	+	+	+

Literatur

1. Overview of the ORBIT radio grid testbed for evaluation of next-generation wireless network protocols. In *IEEE Wireless Communications and Networking Conference, 2005*, volume 3, pages 1664–1669. IEEE, 2005.
2. Andy Bavier, Nick Feamster, Mark Huang, Larry Peterson, and Jennifer Rexford. In vini veritas: realistic and controlled network experimentation. In *in Proc. ACM SIGCOMM*, pages 3–14, 2006.
3. Eric Eide, Leigh Stoller, and Jay Lepreau. An experimentation workbench for replayable networking research. In *In Proceedings of the Symposium on Networked System Design and Implementation*, 2007.
4. Marc E. Fiuczynski. Planetlab: overview, history, and future directions. *SIGOPS Oper. Syst. Rev.*, 40(1):6–10, January 2006.
5. McGregor I. The relationship between simulation and emulation. Number 2, pages 1683–1688, 2002.
6. Scalable Network Technologies Inc. Website QualNet. `http://www.scalable-networks.com/content/products/qualnet`, 6. Juli 2012.
7. Internet2. Website Abilene Network. `abilene.internet2.edu`, 6. Juli 2012.
8. Dipartiment di Ingegneria dell'Informazione Univ. di Pisa Luigi Rizzo. Website Dummynet. `http://info.iet.unipi.it/~luigi/ip_dummynet/original.html`, 6. Juli 2012.
9. Nicta. Website OMF. `http://www.mytestbed.net/`, 6. Juli 2012.
10. The Trustees of Princeton University. Website PlanetLab. `http://www.planet-lab.org`, 6. Juli 2012.
11. The University of Utah. Website Emulab. `http://www.emulab.net`, 6. Juli 2012.
12. UPMC Paris Universitas on behalf of the OneLab partners. Website OneLab. `http://www.onelab.eu`, 6. Juli 2012.
13. UPMC Paris Universitas on behalf of the OneLab partners. Website OneLab. `http://www.isi.edu/nsnam/ns/`, 6. Juli 2012.
14. Larry Peterson, Steve Muir, Timothy Roscoe, and Aaron Klingaman. PlanetLab Architecture: An Overview. Technical Report PDN–06–031, PlanetLab Consortium, May 2006.
15. Larry Peterson, Vivek Pai, Neil Spring, and Andy Bavier. Using PlanetLab for Network Research: Myths, Realities, and Best Practices. Technical Report PDN–05–028, PlanetLab Consortium, June 2005.
16. Timothy Roscoe. 33. the PlanetLab platform. pages 567–581. 2005.
17. Manpreet Singh, Maximilian Ott, Ivan Seskar, and Pandurang Kamat. Orbit measurements framework and library (oml): Motivations, design, implementation, and features. In *Proceedings of the First International Conference on Testbeds and Research Infrastructures for the DEvelopment of NeTworks and COMmunities*, TRIDENTCOM '05, pages 146–152, Washington, DC, USA, 2005. IEEE Computer Society.
18. Mobile Sytems Laboratory (UCLA). Website WhyNet. `http://pcl.cs.ucla.edu/projects/whynet/`, 6. Juli 2012.
19. Los Angeles University of California. Userguide Emulab. `https://users.emulab.net`, 20. Mai 2012.
20. Junlan Zhou, Zhengrong Ji, and Rajive Bagrodia. Twine: A hybrid emulation testbed for wireless networks and applications. In *In IEEE INFOCOM 2006*, pages 23–29. Society Press, 2006.