第 XXVI 部

実ノードを用いた大規模なインターネットシミュレーション環境の構築

第 26 部
実ノードを用いた大規模なインターネットシミュレーション環境の構築

第 1 章　はじめに

Deep Space One ワーキンググループは実環境向けのハードウェアおよびソフトウェアを利用した大規模な実験用環境の構築・運用に関する研究に取り組んでいる。実ノードを用いた大規模な実験設備として StarBED や GARIT、仮想機械を利用した VM Nebula などの開発と運用を通して、実験設備のあり方や、実験設備の制御方法、さらに、実験の体系的分類などの議論を進めている。また、これに加え、実験を柔軟かつ容易に行うため、実験用環境への実トポロジの再現手法や、標準的な実験のテンプレート化などの研究を行っている。

本年度から、Deep Space One ワーキンググループで扱っている大規模実験環境の構築とは別に、大規模実験環境のユーザ視点から利用方法やノウハウの共有、実験例、新たな利用例の考案を行うため、Nerdbox Freaks ワーキンググループが創設された。Nerdbox Freaks ワーキンググループでは、大規模実験環境での実験を、より現実性のある実験評価環境にするべく大規模実験環境における模倣インターネットの構築に取り組んでいるほか、StarBED の利用方法や StarBED と類似した小規模な実験環境の構築方法を演習形式で行うワークショップを開催している。

本報告では Deep Space One ワーキンググループ及び Nerdbox Freaks ワーキンググループの本年度の主な活動について報告する。本報告の構成は以下の通りである。

- Deep Space One ワーキンググループ
  - ネットワーク実験の性質
  - StarBED におけるインタラクティブなノード制御
  - StarBED を利用した体験演習環境
    - SOI Asia 2008 Spring Global E-Workshop
    - インシデント体験演習環境の設計と構築
  - 各種ツール入門マニュアルの整備
- Nerdbox Freaks ワーキンググループ
  - 模倣インターネット環境構築への取り組み
    - 模倣 eBGP 網構築
    - 模倣 eBGP 用部分グラフ抜き出し方式
  - 模倣インターネットのデモンストレーション
    - インターネットコンファレンス 2008
    - ACM SIGCOMM 2008 Demo Session
  - BabyStarBED 構築ワークショップ

（StarBED の詳細については文献 [229] 参照）

第 2 章　ネットワーク実験の性質

重要なサービスがインターネットを利用して提供されるようになってきており、インターネットは今やガスや水道のような社会インフラのひとつとして認識されつつある。インターネットの成長は新たな技術の導入によって支えられ、その成長がさらに新たな技術の開発の呼び水となり、インターネットだけではなくさまざまなネットワーク技術が開発されてきた。

このような成長を支える技術の開発には、当然その検証は非常に重要である。初期のインターネットではインターネット自身が実験環境であったため、新たな技術も詳細な検証なしにインターネットに導入され、そこで検証が行われていた。しかしながら、現在のインフラとしてのインターネットでは、既存の重要なサービスへの影響を排除するためこのような検証方法は許されない。

このような問題を解決するため、さまざまなソフトウェアシミュレータ [15, 34, 35, 138, 149] や、実機を利用したテストベッド [119, 189]、仮想ノードを利用したテストベッド [117, 142] などが提案、運用され、そして実機や仮想ノードを利用したテストベッドでの実験を容易にするためのさまざまなツール群 [47, 119, 147, 176] が利用されている。

インターネットはその成長の速度ゆえに、実験ネットワークから運用ネットワークへの明確な切り替えが行われなかった。このため、実験用の環境を構築する技術および使用方法が未成熟である。

本報告では既存のアプローチをまとめ、各実験の目的などを満たすために利用できるアプローチについて整理する。

2.1 ネットワーク実験

まず、本報告で扱うネットワーク実験の定義を、構成要素、実行順序などについて簡単にまとめる。

2.1.1 ネットワーク実験のアーキテクチャ

実験用ネットワークは PC や仮想ノードといったノードやスイッチがリンクで接続されて構築される。ここでの実ノードは物理的に個々に存在するノードを指し、仮想ノードは、VMware[183] や Xen[144] 等を利用し、一台の実ノードの上にソフトウェアで模倣し仮想的に実現したノードを指す。本報告では、スイッチノードを含め、実ノード、仮想ノードを “ノード（node）” と呼ぶ。これには、ソフトウェアシミュレータ上の抽象化されたノードも含むこととする。また、各ノードを接続するケーブルなどの要素を “リンク（link）” と呼ぶ。“ネットワークトポロジ（network topology）”（もしくは “トポロジ（topology）”）は、ノードとリンクにより形成されるネットワークの形を表し、インストールされている OS やアプリケーション、そしてノードの仕様を含まないこととする。実験用ネットワーク上の各ノードは、その動作の結果としてリンク上に “トラフィック（traffic）” を送出する。

ソフトウェアシミュレータや実ノードによるテストベッドなどの環境を “実験実行環境（environment for conducting experiment）” と呼ぶ。実験実行環境上には、実験用ネットワークトポロジおよび、実験用トポロジを構成するノードやトラフィックを監視するための “管理用ノード（node for management）” および “管理用アプリケーション（management application software）” などの要素により実験を実行するための環境が構築される。ある実験を実行するために必要なすべての要素を含んだ環境を “実験駆動単位（experimental setup）” と呼ぶ。ここで定義した実験のアーキテクチャを図 2.1 に示す。

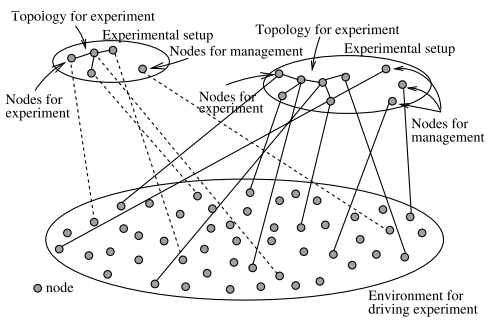

図 2.1.　実験のアーキテクチャ

2.1.2 実験駆動単位に要求される機能

実験実行者は対象環境を実験駆動単位として構築し、実際に実験を実行する。このために必要な機能を列挙する。

- 対象技術の駆動
- 対象実験トポロジの構築
- 全構成要素の観測
- 実環境からの隔離
- 実験結果のトレース
- 実験の再現
- 実環境の再現性

すべての実験実行環境はすべての機能を完全な形でサポートする必要はなく、またサポートすること自体も困難である。実験目的や抽象度の度合いはそれぞれにより異なっており、実験実行環境が対象とする実験群により、それぞれをサポートする度合いも異なる。実験中の実験駆動単位構成要素のさまざまな状態の保存とその提供で、実験結果をトレースすることが可能となる。また、実験には複数回実行した際に同様の結果が得られることが求められる。対象環境を想定した際にその環境を実験駆動単位上に模倣することも重要な機能の一つである。

2.1.3 実験駆動単位の構成要素

実験駆動単位を構成する要素を以下に挙げる。

- 実験の対象技術
- 周辺技術
- 要素の制御用、観測用技術
- 対象技術によるトラフィック
- 周辺技術によるトラフィック

対象技術および対象技術によるトラフィックは基本的に観測する目的となる要素である。基本的にはトラフィックは対象技術自体を動作させることで得られるが、場合によってはトラフィック自体をエミュ

レートもしくはシミュレートすることで、対象技術を必要としない場合もある。周辺技術に関してもその結果トラフィックが発生するが、周辺技術の挙動が非常に詳細に観察され模倣できるのであれば、トラフィックだけを実験駆動単位に導入することも可能である。

上記の技術の実装自体は、ソフトウェアシミュレータなどであればモデル化されたものであり、実ノードを利用したテストベッドでは実環境用のハードウェアやソフトウェア、そして仮想ノードを利用した場合は実環境用のハードウェアは利用できないため、ソフトウェアのみなどといったようにその抽象度は異なる。また、これらの要素に関してもすべての実験においてすべてが必要になるわけではない。実験の段階や目的により取捨選択されるべきである。

2.1.4 論理的検証と実践的検証

検証には論理的なものと実践的なものがある。理論的検証ではアルゴリズムやプロトコルの外乱がない理想的な環境での挙動や性能の検証が行える。これに対して、実践的検証では、外乱を含めて検証が行える。ここで外乱には、アイディアを実装する際に発生するハードウェア、ソフトウェアのバグや、ネットワーク上に存在する他のサービスなどによる影響、そして、刻々と変わり続けるノードやリンク状態が与える影響などが考えられる。

論理的な検証は、モデル化した環境上で対象のアルゴリズムやプロトコルの論理的な挙動、そして性能を把握し、その技術の論理的な意義の確認を行えるとともに、対象技術を実際に実環境用に実装した際の挙動、性能の指標の作成のために利用できる。一方、実践的検証は最終的に対象環境に導入する実装の検証に利用できる。論理的な検証時には想定していなかった、実環境のトポロジやサービスによる影響、実装方式による性能、そして実装時のバグなどを含めた検証が行える。

論理的検証と実践的検証の対象や目的は異なり、それぞれが実験実行者および技術の目的により使い分けられる。論理的検証では、一般的に実環境の実装を利用することはできないが、対象要素、対象環境をモデル化することで理想的な環境での実験が行える。また、少数の計算機もしくは、計算式などによる検証が可能であるため、実行が容易であるといえる。実践的検証では、実環境用の実装をさまざまな点から検証できることが利点である。導入対象の環境を模倣しその上で対象技術を動作させることで、導入した際の問題点の把握と修正を事前に行うことも可能である。また、一般的に実環境用に技術を実装するよりも、論理的検証のための実装の構築および、対象の実験駆動単位の構築が容易であることも利点としてあげられる。

ある技術に対して論理的検証および実践的検証の双方が行われる必要はないが、その目的に沿ったものが選択されるべきである。

2.1.5 実験の開発段階とネットワーク実験

ネットワーク実験にはいくつかの段階があり、その段階により実験内容および利用できる実験実行環境は変化する。本節では実験の段階について整理する。

単体テスト 対象となる技術を論理的もしくは実践的に実装し大まかにその挙動を確認する。

最低限のトポロジでの挙動検証 単体テスト後には、対象技術が動作する最低限の環境での結合試験を行う。これにより対象技術の最低限の挙動の正しさを確認、修正する。

周辺要素を排除した現実的な環境での挙動検証 導入対象の環境を考慮した環境での検証を行う。ただしこの場では、外乱を排し対象技術および関連技術のスケーラビリティについての検証を行う。

周辺要素を含んだ現実的な環境での挙動検証 検証の最終段階として、周辺要素を含め対象環境をできるだけ模倣した環境での検証を行う。この段階での目的は対象技術が周辺技術からどのような影響を受けるのか、また周辺技術にどのような影響を与えるのかを観測し、問題があれば修正することである。

性能試験 以上のすべての段階は仕様などに関する挙動についての検証であるが、当然性能に関する検証も重要である。性能検証は各段階で必要に応じて実施される。

ここであげた各段階は当然すべて行われる必要はなく、実験実行者の目的、対象技術の目的などによってさまざまである。ある段階で問題が見つかり修正された場合には、適切な段階での再検証が必要となる。

図 2.2 に実験順序の例を示す。プロトコルやアルゴリズムの設計は図中 (a) の順序で行われ、その後実装を各段階で行った上で検証をおこなう

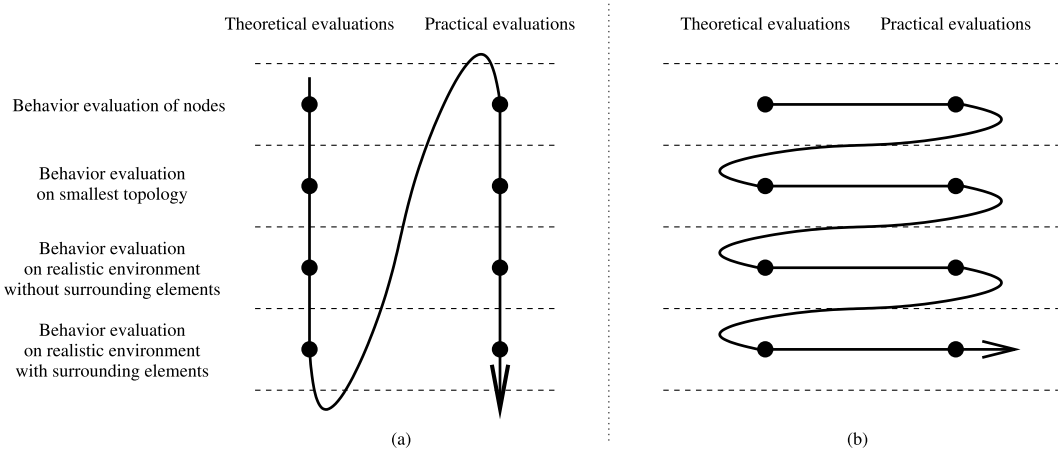

図 2.2.　検証段階の例

場合もある。図中 (b) では、論理的検証と実践的検証を交互に行っており、各段階での論理的、実践的な問題を修正しながら、より大規模かつ現実的な環境での検証を行っている。この順序では対象環境への実装導入を見据え、問題を早い段階で解決することができる。

2.2 実験のフェーズ

ある実験を行うには、実験計画をたてるフェーズ、計画にしたがって実験を実行するフェーズ、そしてその結果を解析するフェーズがある。本報告ではそれぞれを、“計画フェーズ (Planning Phase)”、“実行フェーズ (Execution Phase)”、“解析フェーズ (Analysis Phase)” とする。

本節ではそれぞれのフェーズの概略と、支援可能性について述べる。

2.2.1 計画フェーズ

実験を行うためには、対象技術の目的や仕様などから、以下の内容を決定する必要がある。

- 実験トポロジ
- 実験シナリオ
- トラフィックモデル
- 観測手法および場所
- 利用できる構成要素 (実ノードもしくは仮想ノード、モデル化された要素など)
- 利用する実験実行環境

それぞれの内容を決定するためには、実験支援専用ではなく一般的なツールが利用できる。

実験トポロジの決定にはトポロジジェネレータ [23, 170] の利用やスケールフリー [16] やスモールワールド [114] などのトポロジモデルなどを応用することで可能である。また、各種ベンチマークにはダンベル型トポロジやスター型トポロジが一般的に利用される。

ハードウェア、ソフトウェア的にさまざまなモデルに従ったトラフィックを生成する技術が開発されている [5, 168, 169]。

観測手法については SNMP や ICMP を利用したネットワーク観測ツールが実ネットワーク上でも非常に広く利用されており、これらを実験駆動単位に組み込むことが可能である。

実験の性質などから、実験のカテゴリを定義しそれらに適したテンプレートを用意できれば理想的であるが、実験を分類することは困難であるため、現在は、これまで行われてきた実験の内容をテンプレートとして提供することが現実的である [128]。テンプレートは前述した要素すべてを含むこととなる。

2.2.2 実行フェーズ

実験の実行フェーズでは計画にそって実験を遂行するフェーズである。フェーズの入力は計画フェーズの出力であり、出力は実験データとイベントログや計測結果などの観測データである。

このフェーズでの支援内容としては以下の項目があげられる。

- 実験実行者による実験設定の読み込み
- 実験駆動単位の構築
  - 実験トポロジの構築
  - ノードへのソフトウェア導入

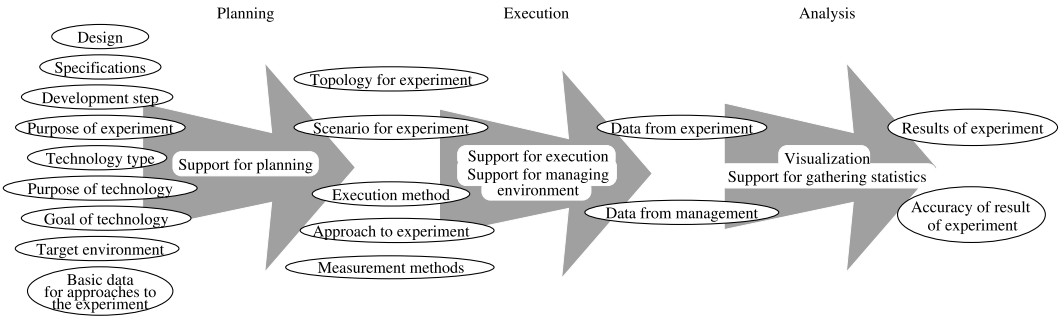

図 2.3. 実験フェーズの入出力

– シナリオの実行
– 観測系の構築
– ログの収集

● 実験リソースの管理

また、これ以外にも、各実験駆動単位の隔離がセキュリティ実験などでは重要となる。

実験の実行に関しては Emulab や StarBED/SpringOS、各種ソフトウェアシミュレータといった実験実行環境によってさまざまな点から支援が実現されている。

2.2.3 解析フェーズ

一つの実験の最終段階として、実験の結果及び観測データを解析し、その結果を得る。このフェーズの入力は、実行フェーズの出力であり、出力は統計データ、可視化されたデータ群となる。

このフェーズでの支援としては、リンク特性などを含めた実験トポロジの可視化や、一般的な統計手法を容易に使うためのインターフェースの提供などが考えられる。ただし、実験により、必要とされる結果は異なるため、完全な自動化を実現することは困難であり、各種利用情報を保存するためのデータフォーマットなどを定義し、一般的な情報に対してのみ可視化および統計手法を提供することが現実的である。対象情報には、CPU や主記憶、ネットワークの利用状況、各ノードで実行されたコマンドリスト、管理データ、リンク特性の変化などが考えられる。

このフェーズでも既存のさまざまなツールが利用可能である。グラフの表示に関しては、Graphviz[56]、tulip[12]、plankton[71] が広く利用されており、ソフトウェアシミュレータでもさまざまな可視化ツールが用意されていることがあり、ns 用の nam[50] は広く利用されている。

図 2.3 に各フェーズの入出力の関係をまとめる。

2.2.4 実験全体

実験の全体の支援としては、各フェーズで利用されるノードやツールの精度や正確性の保証や、各フェーズの出力の正当性の保証などがあげられる。また、Emulab Workbench のように実行された実験の情報をすべて保存し、テンプレートとして提供する試みも重要である。

実験のカテゴリおよびそれに対応する実験内容の決定は一つの理想的な目標としてあげられる。これを実現することにより、同様の目的をもつ実験技術を統一した指標のもとに評価でき、その比較の容易化や、実験実行者は対象技術の改良にのみ注力できるという点が実現できる。

（詳細については文献 [120] 参照）

第 3 章　StarBED におけるインタラクティブなノード制御

これまで我々は、StarBED のようなテストベッドにおけるネットワーク実験支援ソフトウェアの集合体である SpringOS を開発してきた [119]。その中心は Kuroyuri と呼ぶスクリプト言語処理系で、実験で使用する資源やその手順を記述したスクリプトによって実験を駆動する [26]。この形態は資源確保やシナリオ実行を一括で処理する、バッチ処理である。一方、実際の実験現場では、実験中の特定の処理を繰り返し実施したい、スクリプトを記述する前の予備実験のためノードの電源投入を行いたい、という要望が出てくる。我々はユーザの入力を逐次実行する機構でこのような要望を解消する。すなわちインタラクティブなユーザ・インタフェースを設ける。

3.1 OS 切替え

StarBED では全てのノードはネットワーク起動（PXE[74]）としてある。PXE はブートローダをネットワークから取得する機構で、DHCP[43] や TFTP[167] を用いる。転送するブートローダを切替えると、起動する OS を切替えることができる。そこで、我々はブートローダをシンボリック・リンクで管理（作成・削除）するサーバ DMAN を実装した。

3.2 電源投入・切断

StarBED では Wake on LAN（WoL）[4] によって、ネットワークを介してノードの電源を投入する。WoL はデータリンク層の技術のため、指令する PC と指令を受けて起動する PC が同じサブネットワークに接続している必要がある。数百台規模の実験設備では、その全てを一つのサブネットワークに接続することは難しい。同様に、ユーザとノードが直接同じサブネットワークに接続することも難しい。そこで、WoL を中継するプログラム wolagent を作成した。図 3.1 にその配置を示す。

StarBED の大部分のノードは電源を切断するための管理ハードウェアを持たない。そこで、ソフトウェアで電源切断を行う。各ノードで SNMP 受信プログラムを設置し、特定メッセージを受信した際に shutdown プログラムを呼び出す。同様に reboot プログラムの呼び出しによって、再起動も実現した。図 3.2 にその配置を示す。歴史的経緯により、これらの SNMP メッセージは NEC MIB を利用している。NEC 製の snmpd プラグインや、我々が別途開発した SNMPmine を用いる。

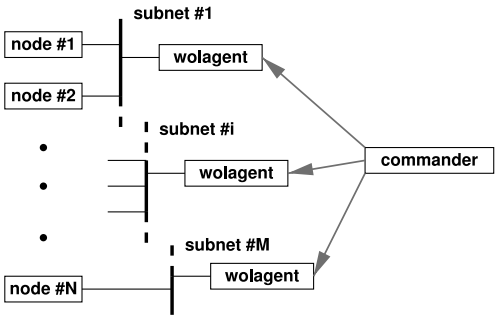

図 3.1.　WoL 関連配置

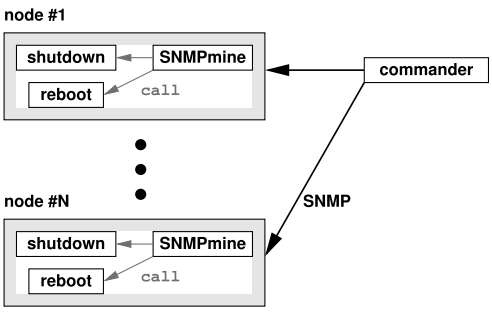

図 3.2.　SNMP 関連配置

3.3 ユーザ・インターフェイス

GUI で数百台制御する際には数百回、数千回マウス操作が必要となり現実的ではない。そこで、今回は CUI を採用した。作成した CUI プログラム、sbpsh を用いることで、ユーザはノードの電源投入・切断、リブート、OS 切替を指示できる。たとえば、ノード 100 台を第二パーティションで起動する際には次の 2 行を入力する。

```
> setdiskboot sintclb1-100 2
> poweron sintclb1-100
```

ハイフンは連続したノード群を表現する。このコマンド列は、DMAN へ 100 台分のブートローダ切替え、wolagent へ 100 台分の起動を発行する。

また、コンマによって隔たりがあるノード群を指定できる。以下の例は、sintclb8 を除いた 99 台の電源オフを指示する。

```
> poweroff sintclb1-7,sintclb9-100
```

3.4 おさらい

ノードをインタラクティブに制御するため、ユーザインターフェイス sbpsh を開発した。sbpsh はノードの電源投入・切断、リブート、OS 切替を指示できる。図 3.3 に全体のシステム構成を、図 3.4 に技術

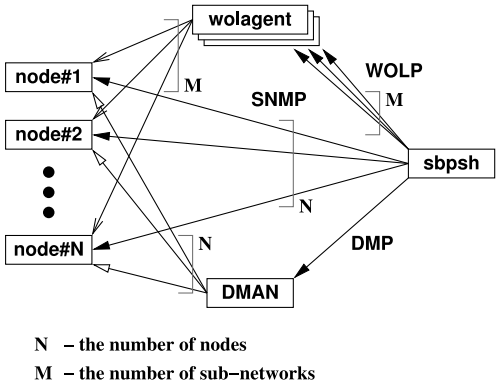

図 3.3.　システム構成

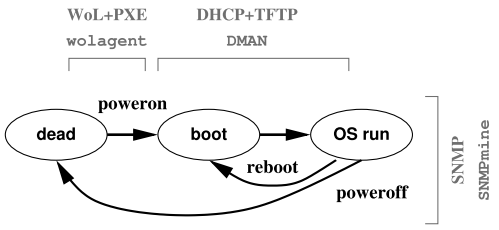

図 3.4. ノード活動サイクル

名や実装名を加えたノード活動サイクルを示す。（詳細は文献 [217] を参照）

第 4 章　StarBED を利用した体験演習環境の構築

ネットワーク技術が普及するにつれ、ネットワーク管理者や技術者の不足が問題となってきている。このため、多くの教育機関では講義を通した学習を行っているが、ネットワーク管理技術の習得には、実際のネットワーク機器やコンピュータを利用した実践的な体験演習が不可欠である。従来のネットワーク技術者は、実験ネットワークであったインターネットそのものの運用を通し、さまざまな失敗をすることにより技術を習得してきた。しかし、現在の社会基盤としてのインターネット上では、失敗がインターネット上の重要なサービスに重大な影響を与える可能性が大きく、失敗が発生するような活動を行うことはもはや許されない。したがって、実践的な体験演習を行うためには、インターネットとは別の体験演習専用の環境が必要である。このような環境を構築するための研究は広く行われている。直感的にもっとも容易に環境を構築する手法は、PC やネットワーク装置を一カ所に集め、それらの機材を利用して体験演習環境を構築する方法である [30]。この方式では、受講者と講師が同じ場所にいる必要がある。ここで問題となるのは、以下の 2 点である。

1. 環境構築のためのリソースの収集のコスト
2. 講師と参加者が一カ所に集まるためのコスト

これらの問題は体験演習の規模が巨大化するほど顕在化する。体験演習が常時行われている状況であれば、必要となる機材を定常的に運用することもできるが、そうでない状況では、多数の機材を用意する経済的コスト、設置のための空間的コスト、それらを管理するための人材的コストなど多くの問題が発生する。また、地理的に離れた場所にいる講師および参加者が体験演習に参加したいという要求が発生することが多い。このような場合、一カ所に講師および参加者が集まるためには、時間的、経済的なコストが発生する。前者の問題解決には VMware[183] や Xen[144]、Linux VServer[104] などによる仮想機械を利用し、一台の物理ノードを多重化する手法などがとられている [7, 21, 112, 130]。後者の問題には、VPN や ssh を利用し、外部からの接続を受け付ける手法が一般的である [21, 112, 148]。これらを複合した環境も当然存在する。体験演習環境を常時管理することは前述のコストから容易ではないため、さまざまな研究が行われている。一方、近年、ネットワーク実験に関する研究が盛んになるにつれ、Emulab[189]、StarBED[119]、PlanetLab[142]、VM Nebula[117]、GARIT といったネットワーク実験専用のテストベッドが世界各地に設置されはじめている。このようなテストベッドは、時分割および空間分割により、複数の利用者で共用することで、その運用コストを下げるという運用形態をとっている。ネットワークテストベッドと体験演習環境の要求事項はきわめて類似しており、ネットワークテストベッドの実験の一つとして体験演習を行うという試みは直感的である。

本節では、体験演習環境に必要な機能について議論し、実際にネットワークテストベッドの一つである StarBED を利用した体験演習の実例として SOI Asia 2008 Spring Global E-Workshop およびインシデント体験演習について述べる。

4.1 体験演習環境

本節では、体験演習環境で行われる内容および環境の性質についてまとめる。

4.1.1 体験演習の内容

体験演習の形態にはさまざまあり、物理的にコンピュータをケーブルで接続し、OS をインストールする事から作業を始める場合や、初期状態としてある環境が用意され、その環境に変更を加えることで目標を達成するといったケースがある。前者は、物理的な機材の配線作業による物理トポロジの構築、OS およびアプリケーションのインストールと設定などのすべての過程を体験演習の一部として参加者が行

う形式である。この形式の体験演習は、物理的な配線作業が必要となるため、遠隔での実施は困難である。一方、初期状態の環境を講師側で用意し、それに変更を加えていく形態の実験は、最低限の設定を前もって行うことで、遠隔利用が実現できる。また、仮想機械を利用する場合でも、初期状態の用意の仕方により、利用している演習用ノードが仮想機械であるのか、物理機械であるのかを参加者から隠蔽できる。両者ともその目的は実際の環境中で利用されている環境を模倣した環境上で、実運用されている機器を利用しながら、現実に即した問題への対処や、環境構築などを行うことである。このような環境上で、失敗を含めさまざまな体験をすることで、どのような操作が環境全体にどのように影響を及ぼすのか、ある問題に対してはどのような対処法が存在するのかなどを、身をもって学習することが重要である。

4.1.2 体験演習環境の構築

前節で述べた、ネットワークトポロジおよび各ノードの設定の全行程を体験演習の一部として参加者が行う場合には、PC やネットワーク装置といった物理的な機材を用意するだけで良い。一方、初期状態を提供する場合に必要な作業は以下の通りである。

1. 物理的機材の用意
2. 物理的配線
3. 演習用ノードへの OS と必要なアプリケーションのインストール
4. 演習用ノードの初期状態に必要な設定
5. ネットワーク機器の初期設定

これに加え、体験演習には以下の機能が必要となる。

演習ノードのコンソール監視機構　講師は、体験演習中に各参加者の進行状況や理解度の確認のために、すべての参加者の演習状態を確認したいという要求がある。体験演習の場ではこのような要求のためには、実際に参加者が操作しているコンソール画面を共有し、どのような操作を実行しているかの確認が有効である。これを実現するために、参加者が操作する演習ノードのコンソールを、講師が視認することが求められる。

演習ノードのコンソール共有機構　また、体験演習中には、さまざまな問題が発生すると考えられる。ここで言う問題とは、参加者の操作ミスによる予期されないものである。このような操作ミスは、参加者の不注意により発生するものであるが、体験演習ではこのような操作ミスは避けられない。また、このような操作ミスにより得られる体験こそが体験演習の本質的な目的であるといえる。遠隔参加している参加者が操作ミスにより、OS などや主要なアプリケーションが正常に動作しなくなった場合、参加者がそれを修正することは困難であることが多いため、講師が操作ミスによる問題を解決する必要がある。このため、演習ノードのコンソールを参加者と講師が共有し、問題を修正するための機構が必要となる。

演習ノードの死活管理機構　演習ノードでトラブルが発生した場合などに、ノードをハードウェア的に再起動したい場合や、何らかの原因でノードの電源が切断されてしまった場合には、外部からの電源の投入機構が必要となる。これをここでは死活管理機構と呼ぶ。

外部接続機構　遠隔から講師もしくは参加者が体験演習に参加する場合には、インターネットなどを通して体験演習環境に接続するための機構が必要になる。

外部環境からの隔離機能　外部接続が必要である一方、体験演習の参加者の操作は前もって予測できないため、外部と体験演習環境の接続状態によっては、外部環境に影響を及ぼす可能性がある。これを防ぐためには体験演習環境を十分に隔離する必要がある。遠隔からの参加者および講師がいない場合は、外部接続機能自体を無効にすることが必要である。

参加者の挙動保存機構　すでに述べたとおり、講師は参加者の操作を確認したいという欲求を持つが、すべての参加者のコンソールを常時確認することは不可能である。したがって、問題が発生した場合、もしくはある参加者の進行状況を確認したい場合に、それまでにどのような操作を実行しているかを把握できることが好ましい。このために、各参加者が実行した操作履歴を保存し、講師の要求により迅速に確認できる機構が用意されていることが望ましい。

4.2 体験演習の実例

ここでは StarBED 上に構築された体験演習環境を利用して行われた、体験演習例として SOI Asia 2008 Spring Global E-Workshop およびインシデント体験演習について報告する。

4.2.1 SOI Asia 2008 Spring Global E-Workshop

我々はStarBED上に構築した環境を用いた、体験演習型ワークショップを開催した。SOI Asia Projectはインターネットを利用して、アジア地域を対象とした教育活動を行っているプロジェクトであり、その活動の一部としてリアルタイムでの授業を提供している。このような授業を行うためには、授業の動画をリアルタイムで受信する環境をすべての参加拠点に構築する必要がある。SOI Asia 2008 Spring Global E-Workshopはこのための拠点を構築するオペレータを育成するために2008年3月末から4月初旬にかけて行われた。本ワークショップでは、各エッジノードでのIPアドレスの設定など基本的な設定から、Multicastの経路制御といったネットワーク管理者としての活動までの広い範囲を網羅する演習を行った。

本ワークショップでは合計3つのトポロジを用意し、参加者はそれぞれ1台のPCノードを操作し、実際に遠隔地から送信された動画を受信できる環境を構築した。利用したトポロジを図4.1、4.2および4.3に示す。また、表4.1にOSやその台数などの情報を示す。実験の参加者は、インターネットから、踏み台サーバを経由して、体験演習環境にログインした。このときログインにはsshを用いている。また、体験演習環境は前述の通り実際の環境を体験するために AI3のネットワークに接続された。

体験演習環境の構築はネットワーク実験環境の構築とほぼ同じ機能で実現できる。これには、OSやアプリケーションのインストール、各ノードでのIPアドレスや経路情報の設定および、スイッチでのVLANの設定などによるトポロジ構築がある。StarBED上の実験支援OSであるSpringOSを利用すれば、これらの操作を容易に行える。ネットワークインターフェースの数および性能を重視し、StarBEDのグループFノードを採用した。グループFノードは3.2 GHzのPentium 4のCPU、8 GB memory、2台の80 GBのハードディスク、そして4つのギガビットイーサネットのネットワークコントローラーを搭載している。本節では環境構築に利用した機能および、必要となった時間などをまとめる。

OSとアプリケーションのインストール OSとアプリケーションのインストールにはSpringOSの機能である、テンプレートとして構築・設定した一台のノードのハードディスクの内容を別のノードのハードディスクにコピーすることで、ノード設定を複製する機能を利用した。本ワークショップ用のテンプレートノードのディスクイメージの作成、そしてその配布に必要な時間を計測した。それぞれの結果を表4.2および4.3に示す。

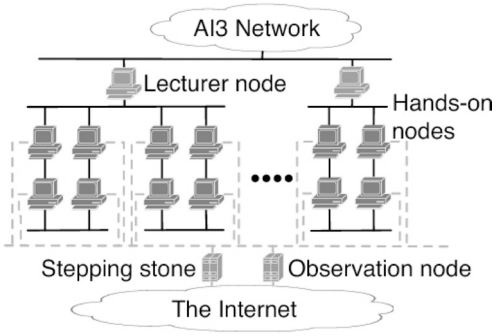

図 4.1. Topology 1

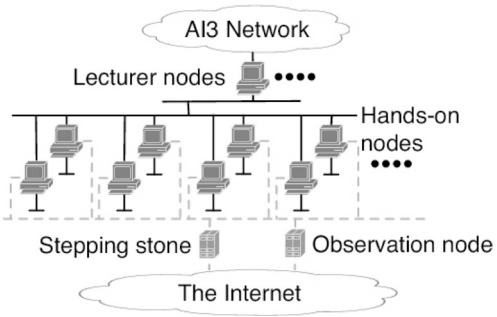

図 4.2. Topology 2

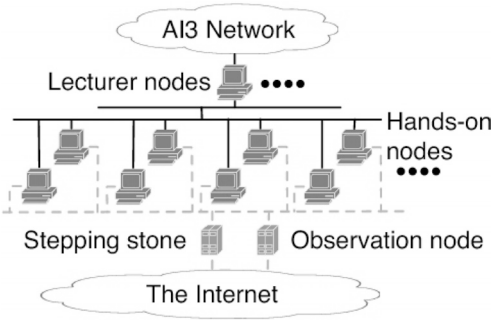

図 4.3. Topology 3

表 4.1. Node and Topology Information

Day	OS	Topology	# of nodes	
			hands-on	lecturer
1, 2	FreeBSD 6.2	1	56	4
3	FreeBSD 6.2	2	56	1
4	Fedora Core 6	3	56	1

表 4.2. Time for Picking Up OS Images

OS	partition size	time (s)
FreeBSD	18 GB	1275
Fedora Core	16 GB	873

表 4.3. Time for Distributing OS Images

OS	# of nodes	time (s)
FreeBSD	67	2552
Fedora Core	60	1832

表 4.4. Time Required to Build Target Topologies

Topology No.	time (s)
1	1528
2	1622
3	1280

ネットワーク設定 ネットワーク設定はSpringOSのSwitch Managerの機能を用いて行った。Switch Managerはスイッチの各ポートのVLANを所有者情報に基づいた調停処理を行いながら設定する。SpringOSは自動的にトポロジを構築する機能を有するが、今回は物理トポロジへの論理トポロジの投影は手動でおこなった。これは、体験実習環境という性質上、論理トポロジと物理トポロジの投影が明確にわかる必要があるからである。各物理ノードのインターフェース番号と所属するべきVLAN番号の記述を入力とし、Switch Managerを介してスイッチの設定を行いスクリプトを利用して、ネットワークの設定を行った。このときに必要であった時間を表4.4に示す。また、ワークショップ後に対象トポロジ構築に必要な時間を再計測した結果を表4.5に示す。

演習ノードのコンソール監視・共有機能 演習ノードのコンソール監視共有機構には、StarBEDに設置されているKVM装置を利用した。これにより、シングルコンソールからStarBEDに設置されているすべての実験用ノードのコンソールの監視、操作が行える。この機能はブラウザを通して利用することもでき、講師はVPNでStarBEDの内部ネットワークに接続、その上でKVMを利用して参加者の進度管理や、トラブル時のサポートを行った。

表 4.5. Time for Building Topologies (Re-measured)

topology	trial	time (s)
1	1	1622
	2	1699
	3	1699
2	1	1622
	2	1629
	3	1629
3	1	1280
	2	1278
	3	1279
Delete VLAN	1	2201
	2	2199
	3	2196

本ワークショップでは、講師がKVM装置に接続できなくなるというトラブルが一件あったが、これはKVMのサーバ機器を再起動することで解消した。これ以外には、環境としての障害が発生することはなく、スムーズに演習が行えたと言える。ワークショップ後のアンケートの結果からも、参加者、講師とも環境に満足したことが確認でき、高性能な演習環境を構築できたと言える。(詳細は文献 [118] 参照)

4.2.2 インシデント体験演習環境の設計と構築

ネットワークセキュリティについて学習する上では、ウィルス・ワーム・ボットなどのマルウェアや各種の攻撃が実際にどのように行われ、どのように観測できるのかを知ることが非常に重要である。そのため、実践的な演習を行うことが求められるが、多人数が同時に利用でき、かつ、現実に近く安全な演習環境を構築・運用することは容易ではない。そこで我々は、StarBEDを用いてマルウェアなどの体験演習を行うことができる演習環境の設計と構築を行った。

インシデントの体験演習には、いろいろな対象者や形式が考えられるが、本稿のインシデント体験演習環境は、文部科学省の「平成19年度　先導的ITスペシャリスト育成推進プログラム」によるプロジェクト「社会的ITリスク軽減のための情報セキュリティ技術者・管理者育成」[87] の一環として行われる教育コースの実践科目の一つ「インシデント体験演習」を対象とする。

「インシデント体験演習」では、現実的な規模と複

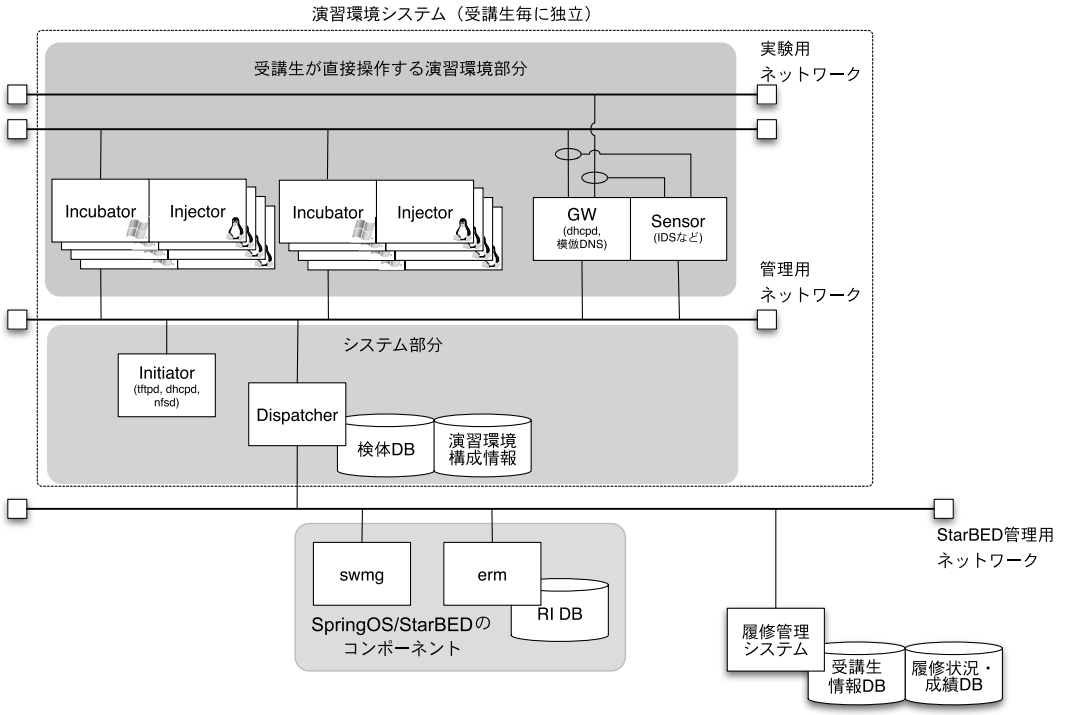

図 4.4. 演習環境システムの概要

雑さを持つサイトへのさまざまな攻撃と、それらに対する監視・分析・防御・回避・復旧等の技術を実践的に体験習得することを目的としており、受講生は 20 名程度を想定する。この科目は、実習を中心に行い、実際にスキャン、DoS 攻撃、ワーム感染、自サイト内のボット、フィッシング被害、P2P クライアント検出などのインシデントを体験し、監視・発見・分析・防御・回避・復旧を行い、レポートを提出する形式で行われる。

サイト構成やネットワーク構成のひな形、利用する攻撃ツールやマルウェア検体は講師が用意し、それらを元に体験演習を行う。体験演習では、損害を与えうる実際の攻撃ツールやマルウェア検体を利用する可能性もあるため、外部に影響を与えないように配慮する必要がある。講師は、自習期間を含めて、体験したインシデントとそれに対するレポートの内容を照合、評価し、成績をつける。演習には、StarBED[229] を用い、実習は StarBED が設置されている北陸リサーチセンターで、自習は遠隔からリモート操作で行う。

体験演習では、講師が用意した現実的な規模と複雑さを持つサイトを構築し、それらに対するさまざまな攻撃を模擬し、受講生が監視・分析・防御・回避・復旧などを実践する必要がある。そのため、受講生向けの模擬サイトの構築と模擬攻撃の実施を支援する機能が求められる。

20 名程度の受講生が同時に演習できるようにするために、同種の演習環境が最大 20 並列程度で動作する必要がある。また、ある受講生の環境から他の受講生の環境へマルウェアの活動や攻撃の影響が波及することがないように、それぞれの環境が独立し、他への影響が排除されている必要がある。同様に、繰り返しの演習が行われる場合、個々の演習が後の演習に影響を及ぼさないように演習環境は演習終了後に浄化される必要がある。自習期間には、講師が介在することなく演習を実施できる必要があり、自動で演習環境の構築、演習の開始、演習結果の記録、演習環境の浄化を行えねばならない。

実践的にマルウェアの活動や攻撃を模擬するために、実際に使われるものと同じハードウェア（PC など）と OS、アプリケーションソフトウェアの上で模擬する必要がある。特に、仮想化技術を利用した場合、再構築の容易さや隔離の側面では有益だが、仮想化技術検知機能を持ったマルウェア検体が動作しないなどの制約がある。このようなことを避けるため、実機を用いる必要がある。また、多くのマルウェアが行うインターネットへの接続性確認などのリアリティチェックを回避する必要がある。

インシデント体験演習環境は、上記の観点から実機による PC クラスタなどの多数の PC からなる環境上に構築する必要がある。演習環境専用に PC クラスタを構築するのは、費用の観点から非効率と考え、既存の PC クラスタを利用するものとする。既存の PC クラスタを利用するため、演習環境には実習期間や自習期間の間に構築と撤収を行えるような構築支援機能が求められる。また、既存の PC クラスタがインターネットに接続されている場合には、演習に際してマルウェアの活動や攻撃の影響が波及することがないように、演習環境は隔離されねばならない。

以上を踏まえ、インシデント体験演習環境の構築支援機能とインシデント模擬機能の設計を行い、実際に、20 名程度で StarBED で 3 日間の演習を行った。図 4.4 に演習環境システムの概要を示す。（文献 [232] 参照）

演習用の検体を提供していただいた奈良先端大インシデントレスポンスチームに感謝します。

第 5 章　各種ツール入門マニュアルの整備

これまで Deep Space One ワーキンググループおよび Nerdbox-freaks では、さまざまなツールを開発してきた。StarBED における実験支援ツールである SpringOS、CAIDA や route view などが計測したトポロジ情報から、Quagga の設定ファイルを生成する AnyBed、そして Xen を用いて多数の仮想ノードからなるトポロジを構成する XENebula などがその例である。これまで、これらのツールのドキュメント整備は十分でなかった。そこで、初心者にも簡単にこれらのツールを利用してもらえるように、チュートリアル的なドキュメントをまとめた。それぞれ「できる StarBED」、「できる AnyBed」、そして、「できる XENebula」として執筆されており、StarBED のホームページ [172]、AnyBed のホームページ [10] からダウンロードが可能である。現時点で「できる XENebula」については最終修正中であり、まだ一般公開はされていない。

第 6 章　模倣インターネット環境構築への取り組み

新しいプロトコルスタックやアプリケーションソフトウェアをインターネット上に導入する場合、事前にインターネットに近い環境で機能やスケーラビリティを検証することが望ましい。そこで我々は、機能や規模が実際のインターネットに近い検証・実験環境を提供するための構築手法（模倣インターネット）の研究を行っている。Nerdbox Freaks では、StarBED での実験利用方法の一環として模倣インターネット環境および模倣インターネット環境を用いた実験環境として NERDBOX（Network Emulation for Realistic Deployment and Behavior Observation eXperiments）を開発している。

6.1 模倣 eBGP 網構築

図 6.1 に示すように、模倣インターネット環境とは、実インターネットの観測データなどを元にして、実インターネットのトポロジーやトラフィック、ノードの振る舞いなどを実験環境上に模倣して作成し、実インターネット上での実証実験と同じくらいリアリティおよびスケーラビリティのある実験を再現可能かつ制御、観測が可能な状態にしてアプリケーションなどの動作検証が行えるようにしようという取り組みである。

現在 Nerdbox Freaks では Anybed[176] と XENebula[231] を用いて StarBED[119] 上に模倣 AS 間ネットワークを構築する NERDBOX のプロトタイプシステムの開発を行っている。物理ノードを用いる場合も仮想ノードを用いる場合も基本的には図 6.2 に示すような形で模倣 eBGP 網は構築される。まず、AnyBed によって CAIDA AS Relationship Data（以降 CAIDA ASRD と略記）[23] から Quagga bgpd 用の設定ファイルを作成し、次に物理ノードもしくは XENebula で作成した ttylinux 上に配布し、Quagga bgpd を起動させて模倣 eBGP トポロジーを作成する。

模倣 eBGP トポロジーを構築する際、トポロジーの大きさや AS 間の結合密度によっては AS 間リンクが 5000 リンク以上存在する可能性がある。そのた

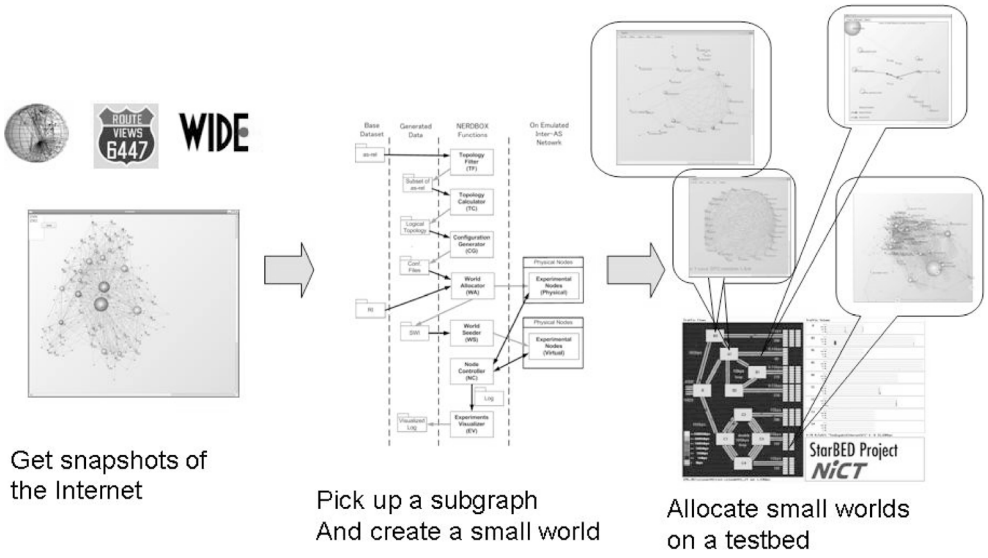

図 6.1. 模倣インターネット構築イメージ

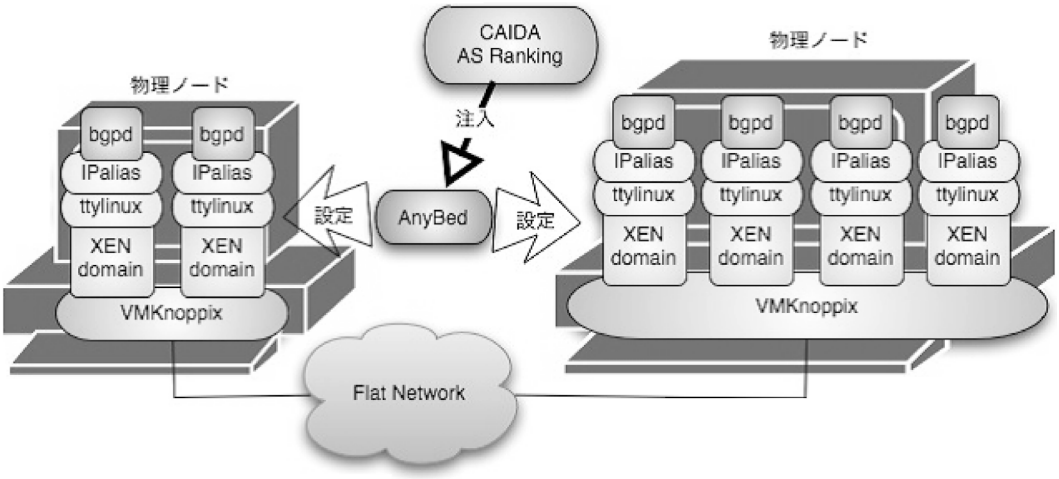

図 6.2. XENebula による仮想ノード構築と模倣 eBGP 作成

め、VLAN による仮想インターフェースを用いた設定ファイル作成では StarBED[119] のような実験環境では VLAN 番号をすぐに枯渇してしまい、模倣 eBGP 網の構築が失敗する。そこで、NERDBOX では IP alias を仮想インターフェースとして利用し、Internet Exchange ポイントのように同一 L2 上で eBGP ピアリングを行う形で模倣 eBGP 網を構築している。

このプロトタイプシステムを用いて、JPNIC（Japan Network Information Center）に登録されているすべての日本の IPv4 の AS（458 AS）からなる AS 間トポロジを物理ノードを用いて模倣することに成功している。また、CAIDA の ASRD の上位 10,000 AS を XEN による仮想化を用いて、120 台の物理ノード上で模倣することにも成功している。この他、単に eBGP 網を構築するだけではなく、構築した eBGP 網が実験者の意図どおりにできているのかを確認するための検証ツールや、eBGP 網の状態を視覚化して把握するためのツールの開発も行っている。

6.2 模倣 eBGP 用部分グラフの切り出し方（Usenix Security CSET 08 Workshop）

CAIDA ASRD には 2008 年 8 月 11 日の時点で 28,744 の AS が記録されているが、これらすべてを StarBED 等の実験環境にマッピングすることは、XEN などの仮想ノードを用いても非常に難しい。そこで、実験目的に合わせた CAIDA ASRD から eBGP トポロジーの部分グラフを抜き出すことが重要になる。部分グラフの抜き出し方法に

ついて Nerdbox Freaks にて検討した結果を整理し、Usenix Security Workshop on Cyber Security Experimentation and Test（CSET '08）にて発表した。（発表概要に関しては文献 [64] を参照）

投稿論文では、部分グラフの抜き出しには、従来方法である AS の Tier レベルに基づき部分グラフを抜き出す Tier-Level Filtering（TLF）の他に 4 種類の方法（Top Ranking Filtering（TRF）、Root-AS Neighbors Filtering（RANF）、Region-Based Filtering（RBF）、List-Based Filtering（LBF）が現実的に必要とされている部分グラフ抜き出し法であると考察し、各 4 種類の疑似コードを記述し、比較検討やフィルタリングルールを組み合わせた部分グラフの抜き出し方などをまとめた。（各フィルタリングルールについては文献 [64] に掲載している論文を参照）

発表後の質疑応答では、以下のような質問を受けた

Q.1 IP アドレスの割り当てはどうしているのか？

実際のプレフィクスを割り当てているわけではない。現在はプライベートアドレス空間（10.0.0.0/8）から各リンクに対し、/24 を割り当てている。

Q.2 BGP トポロジの詳細は？

単純に neighbor として登録する場合と、CAIDA ASRD の Peering 推定に基づき Routemap を作成して構成する場合の 2 種類を行えるようにしてある。現在 Confederation や MED、local pref などのコスト設定は考慮していない。

Q.3 Intra-AS と Inter-AS トポロジのマージについては？

現在仕様を検討している段階である。CAIDA ASRD や RouteViews に掲載されている BGP router ID と RockeFuel に掲載されている OSPF の router ID が一致しない可能性があるので難しい。

Q.4 実際に 600 node で実験しているのか？

かなり簡単に構築し、実験できるようになっている。

Q.5 Emulab のツールは StarBED や構築した eBGP 環境で使えるのか？

我々は SpringOS や Anybed などの独自開発ツールを使っているが StarBED 自体は普通のクラスタで IPMI、ネットブートなどのごく普通のテクノロジを使って構成しているため、Emulab のツールも使えると思う。

Q.6 StarBED について教えて欲しい

StarBED Project のコンタクト先を紹介した。

第 7 章　模倣インターネットのデモンストレーション

Nerdbox freaks では、模倣インターネット環境の活用方法を広めるべく国内外の学会にてデモンストレーションを行った。

7.1 インターネットコンファレンス 2008

国内学会でのデモンストレーションとして、2008 年 10 月 23 日、24 日に独立行政法人情報通信研究機構沖縄亜熱帯計測技術センターで開催されたインターネットカンファレンス 2008（IC2008）にて NERDBOX のプロトタイプシステムによる模倣 eBGP 網のデモンストレーションを行った。（詳細については `wide-papaer-deepspace1-sota-IC2008-00.txt` 参照）

IC2008 でのデモンストレーションでは、2008 年 9 月 WIDE 研究会の実験にて行った模倣 eBGP 網構築実験のうち、PC10 台を用いた仮想マシンによる日本の IPv4（449 AS）の eBGP トポロジの構築とトポロジの可視化と操作が行える Viewer のデモンストレーションを行った。（デモンストレーションの物理構成詳細に関しては 9 月 WIDE 合宿実験の報告を参照）

7.2 ACM SIGCOMM 2008 Demo Session

国外の学会でのデモンストレーションとして、ACM SIGCOMM 2008 Demo Session において "NERDBOX: An Emulated Internet for Scalability Tests of Running Codes" というタイトルでデモンストレーションを行った。（ACM SIGCOMM 2008 Demo Session に投稿した Extended Abstract については `wide-paper-nerdbox-freaks-sigcomm2008-00.txt` を参照）

Nerdbox Freaks では、模倣インターネット環境を NERDBOX（Network Emulation for Realistic Deployment and Behavior Observation eXperiments）と銘打ち、新しいプロトコルスタックやアプリケーション、新サービスの大規模スケーラビリティテストと実インターネットへの導入シナリオの検証環境として用いる環境として、デモンスト

図 7.1. デモンストレーション会場の様子

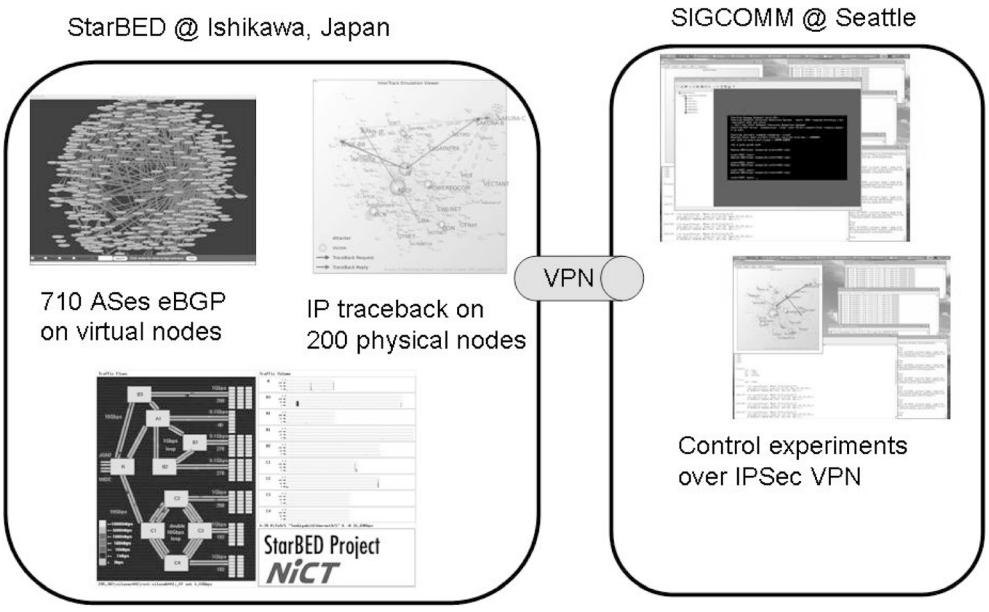

図 7.2. デモンストレーション用トポロジー

レーションを行った。デモンストレーションは 8 月 20 日（15:15–16:15）と 8 月 21 日（12:00–13:45）の 2 日間行われた（図 7.1 はデモンストレーション会場の様子）。

図 7.2 が ACM SIGCOMM 2008 で行ったデモンストレーションのトポロジーである。デモンストレーションでは、NICT 北陸リサーチセンター内にある StarBED と ACM SIGCOMM 2008 の間を VPN で接続し、StarBED 上のグループ A とグループ F で 2 つの実験を実際に行い、図 7.3 に示す形で各実験の操作と状態監視をデモ会場から行うという形でデモンストレーションを行った。グループ A の実験ではトレースバックワーキンググループの協力のもと、物理ノードを用いて日本国内 BGP のトップ 200 AS 分のトポロジーを摸した環境での IP トレースバックの動作実験を行い、デモ会場側で QT4 で作成し

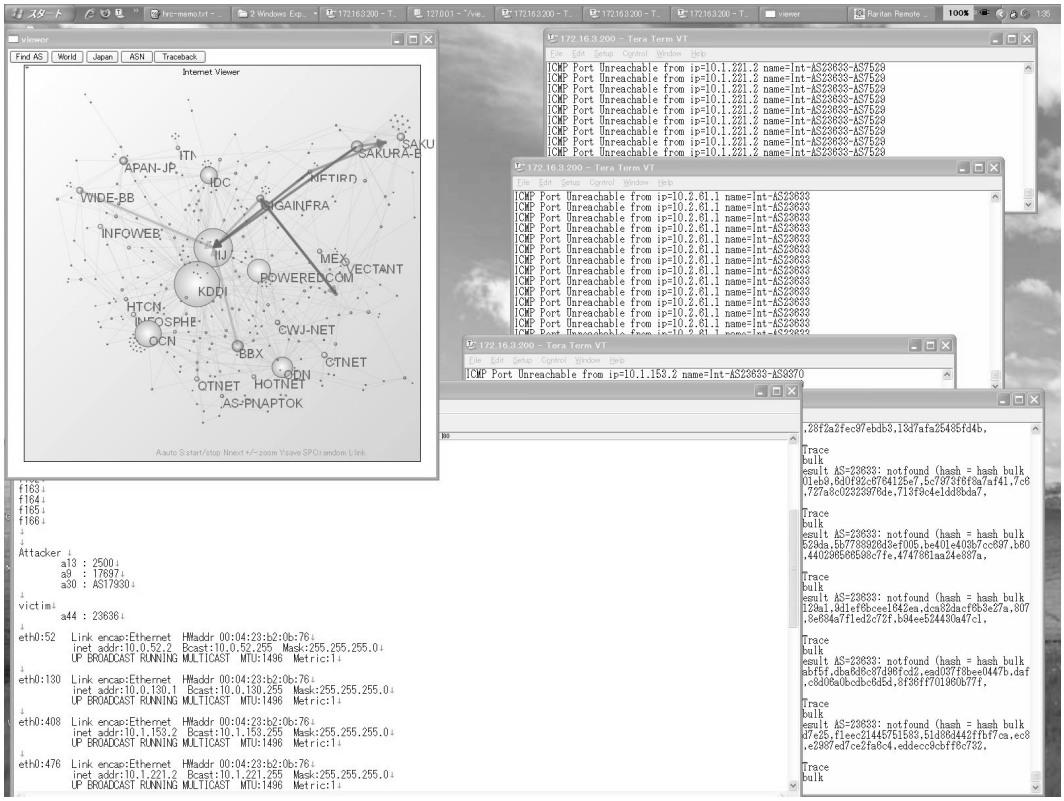

図 7.3.　デモンストレーションのスナップショット

た viewer 上に StarBED 上での実験状態をリアルタイムで描画し実験の状態監視を行うデモンストレーションを行った。一方、グループ F では XENebula を用いて物理ノード 20 台の上に仮想ノード 710 台を構築し、2008 年 7 月の時点の日本国内と周囲 1 AS ホップを交えた 710 AS を模倣した eBGP 環境を構築し、TouchGraph を用いて作成した viewer により BGP ステータス監視を行った。

デモンストレーションでは来場者、とくに研究者から興味を持っていただき、いくつかの質問を受けた。質問をまとめると以下の通りである。

Q.1 StarBED は米国から使えるのか？

StarBED Project に尋ねてみてください。

Q.2 AnyBed は Emulab でも動くのか？

AnyBed は Quagga と Debian をベースにしたレイヤ 3 のコンフィグを作成するだけなので、問題なく動くと思います。

Q.3 自分で改良した BGP（Quagga）を StarBED で試すことはできるか？

問題なくできます。独自カーネルでの実験なども行ったことがあります。

Q.4 この Viewer（トレースバック用の viewer と BGP ステータスの viewer 両方）はどこで入手できるのか？

どちらも自作で、トレースバック用の viewer は QT4 で作成し、BGP の viewer は Java の TouchGraph を使って作成している。残念ながら今は公開していないが、そのうちオープンソース化するつもりでいる。

第 8 章　BabyStarBED 構築ワークショップ

Nerdbox Freaks では StarBED と同質の機能を持つ小規模テストベッド環境の構築方法や、SpringOS、AnyBed、XENebula など、Deep Space One ワーキンググループで研究開発が行われている大規模検証環境における実験環境構築ツールのインストール、設定方法や、実際の実験における使い方などの伝授と情報共有、そして、大規模検証環境における新たな研究課題の追求を行うべく、2008 年度に 3 回にわたって「BabyStarBED 構築ワークショップ（通称

できる StarBED ワークショップ)」と題して、演習形式で各種テストベッド環境構築ツールの習得とブレインストームを行うワークショップを開催した。

8.1 第 1 回ワークショップ

第 1 回ワークショップは、5 月 7 日（水）から 5 月 9 日（金）の 3 日間に渡って、奈良先端科学技術大学院大学インターネット工学講座にて開催された。第 1 回ワークショップの参加者は 10 名であった。

8.1.1 演習内容

第 1 回ワークショップでは SpringOS の設定方法と SpringOS を用いた実験の演習、および熟練者による SpringOS のカスタマイズ方法を学んだ。

初日の演習では、まず奈良先端大宮本大輔氏が演習環境の物理構成（図 8.1）を説明した。演習環境は図 8.1 に示す Dell 社製のブレードサーバと Proside 製の 1U サーバを用いた小規模クラスタ環境を奈良先端大で用意し、講師の情報通信研究機構宮地利幸氏から要望のあった OS である Debian 4.2Release をあらかじめインストールし、openssh-server、openssh-client などのパッケージをインストールして、遠隔ログイン可能な状態にしておいた。そのあと、宮地氏により StarBED の概要と SpringOS に関する概要を講習していただいた後、2 グループに別れて用意した小規模クラスタ環境に SpringOS のインストールと設定を行った（図 8.2 参照）。

2 日目は、同じく宮地氏により SpringOS に含まれている実験シナリオ制御ツールである kuroyuri を用いた実験シナリオの記述方法と実験の制御方法の講習を行い、iperf を用いたトラフィック計測実験をテスト実験として演習を行った。

3 日目は、初日、2 日目の復習として、再度インストールから実験までを演習した後、東芝土井氏により kuroyuri の実験シナリオのカスタマイズ方法を講習していただいた。

8.1.2 参加者からの感想

参加者から以下のような感想をいただいた。

- 個別に分かれているツールごとに build をするのは手間なので、SpringOS のパッケージ化をきちんとやってほしい。
- kuroyuri の master のオプション数が多いのはいまいち
- kuroyuri のシナリオで perl の BEGIN{} 相当の何かがあればよいのでは？
- 初心者向けに各種ツールのログの見方を教えてほしい

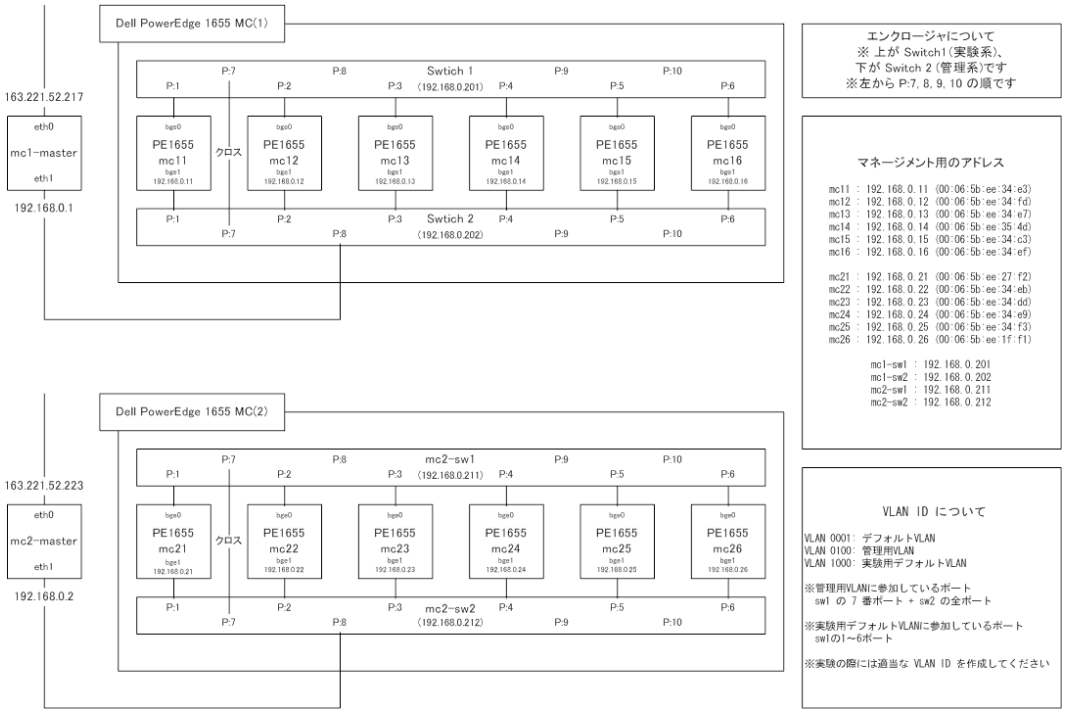

図 8.1. 第 1 回ワークショップ物理トポロジー

図 8.2.　第 1 回ワークショップ演習風景

8.2 第 2 回ワークショップ

第 2 回ワークショップは、6 月 30 日（月）から 7 月 1 日（火）の 2 日間に渡って、北陸リサーチセンターにて開催された。第 2 回ワークショップの参加者は 13 名であった。

8.2.1 演習内容

第 2 回では、講師に情報通信研究機構宮地利幸氏、および AnyBed の開発者である情報通信研究機構鈴木未央氏を迎え、StarBED 全体の説明と利用方法、AnyBed による StarBED での実験方法、および StarBED の物理ノードを用いた模倣 eBGP 環境の構築方法に関して講習と演習を行った（図 8.3 はワークショップの演習風景）。

初日は、まず StarBED の概要に関して宮地氏に講習していただいた。そのつぎに、奈良先端大宮本大輔氏に「こうやれば初心者でもはまらない！ できる StarBED」という題名で、StarBED の特有の利用方法、ノウハウ、TIPS に関して講習していただいた。

そのあと、北陸リサーチセンターで用意していただいた Dell のタワー型 PC 2 台を用いて AnyBed マスターサーバ環境の構築方法の演習を行った。初日の演習では、AnyBed マスターサーバで設定した実験ノードで起動させる Debian の OS イメージを StarBED の TFTP サーバ内で SpringOS に含まれているツールの一つである erm と sbpsh で TFTP の設定を行い、PXEboot にて実験ノードを起動させるところまでを行った。

2 日目は、初日に作成した環境を用いて AnyBed を用いて eBGP 実験トポロジーを作成する方法を学習した。演習は「できる AnyBed」に沿って行われた。演習を一通り終えたのち、奈良先端大櫨山氏により、CAIDA Project による推定 AS トポロジーデータセットを用いた模倣 eBGP トポロジーの作成方法に関する講習を行った。

8.2.2 参加者からの感想

参加者から以下のような感想をいただいた。

- eBGP だけでなく、iBGP 網などイントラドメインのトポロジーまで混ぜて作れるようになると、できる実験の幅が広がるのでぜひ行ってほしい

8.3 第 3 回ワークショップ

第 3 回ワークショップは慶応義塾大学湘南藤沢キャンパスにて 11 月 30 日（日）から 12 月 3 日（水）の 4 日間行われた。第 3 回ワークショップでは慶應義塾大学鈴木茂哉氏の御厚意により、Auto-ID ラボの実験機材をお借りして、ワークショップの演習環境

図 8.3. 第 2 回ワークショップ演習風景

を構築し、演習を行った。第 3 回ワークショップの参加者は 15 名であった。

8.3.1 演習内容

初日（11/30）は、物理環境構築編と題し、StarBED と同質の機能を持つ小規模実験クラスタ（BabyStarBED）の物理環境の構築方法に関して、奈良先端大櫨山氏が講師として講習を行った。まず、奈良先端大で過去 8 年間にわたって構築してきた小規模実験クラスタ（GARIT）を例として講習を行い、その後参加者によってワークショップで用いる物理環境の整備（図 8.4 参照）を行い、サーバへの OS のインストール、コンソールサーバの設置、配線、DHCP 設定、NFS 設定などの基本設定を行った。

2 日目（12/1）は第 1 回、第 2 回ワークショップの復習として、まず宮地氏により、StarBED の概要と SpringOS の概要に関しての講習が行われた。そ

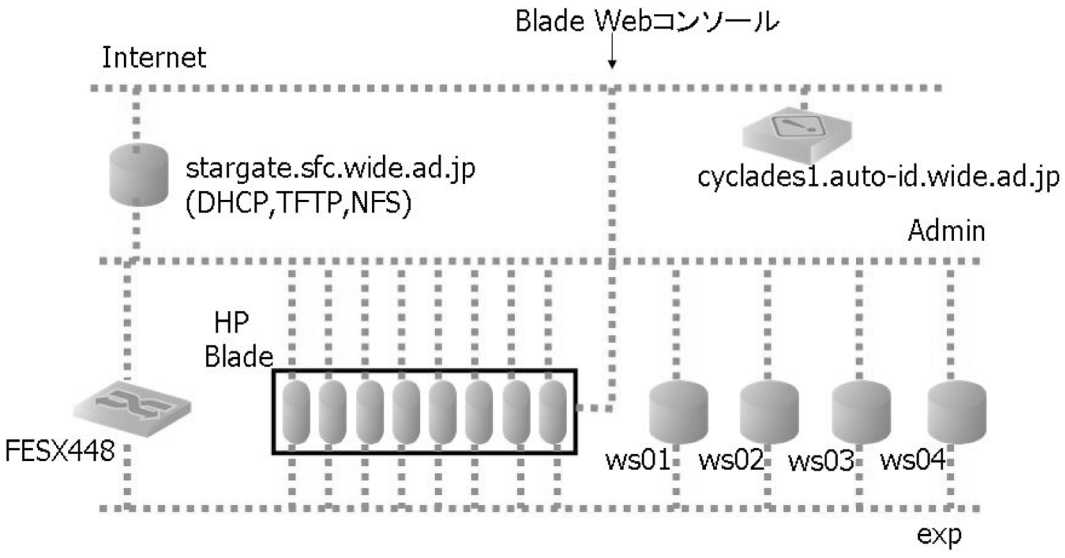

図 8.4. 第 3 回ワークショップ物理トポロジー

図 8.5.　第 3 回ワークショップ演習風景

の後 SpringOS を演習環境にインストールし、erm と sbpsh による実験ノードの起動確認までを演習として行った。

3 日目（12/2）は、まず情報通信研究機構三輪信介氏により XENebula の概要について講習していただいた（図 8.5 は 3 日目の演習中の風景）。XENebula の概要について講習を受けた後、演習として、XENebula 環境の構築を行い、実験ノード上での XEN の起動とその上での仮想ノードの起動の確認までを行った。

4 日目最終日（12/3）には、XENebula 応用編として、さまざまな eBGP トポロジーを作成する演習を行った。演習では、複数物理ノードをまたいで XENebula 環境で構築する数珠つなぎ eBGP トポロジーや、AnyBed で成型した JP ドメイン Top 50 AS 間の eBGP トポロジーの構築、負荷試験として物理ノード 2 台での JP ドメイン 467 AS トポロジーの起動テストなどを行った。午後には、XENebula の内部構造とカスタマイズ方法に関して三輪氏から講習を受けた。

8.4 BabyStarBED 構築ワークショップのまとめ

3 回にわたるワークショップを開催し、どのワークショップもおおむね好評であった。参加者からは、演習を行ったツールに関し、機能拡張に関するコメントや参加者各自の実験への利用に関していくつかコメントをいただいた。頂いたコメントを反映し、各種ツール、ドキュメントのアップデートや、次回以降のワークショップ開催に活かしていきたい。

第 9 章　おわりに

本報告では、今年度の Deep Space One ワーキンググループおよび Nerdbox Freaks ワーキンググループの活動報告を行った。本年度は、各種ツール群の開発以外にそれらのツールを利用しやすくするためのマニュアル群の整備、そして Nerdbox Freaks ワーキンググループを設立し、実験環境をより現実的な環境に近づけるための研究開発を行った。

来年度も Deep Space One ワーキンググループと Nerdbox Freaks ワーキンググループとの協調により、柔軟な実験環境を構築するためのツール群の研究開発とともに、それらを利用しより現実的な実験を行うための研究開発も平行して行っていく。また、

ドキュメントの整備およびチュートリアルの開催による利用者のサポートも引き続き行っていく予定である。