第 XIV 部

Explicit Multicast

第 14 部
Explicit Multicast

第 1 章　はじめに

明示的マルチキャスト（XCAST: eXplicit Multicast）は、従来のISM（Internet Standard Multicast）でのグループアドレスに変えて、パケットヘッダに到達すべき複数のマルチキャストアドレスを明記することで宛先を指定するマルチキャスト方式である。XCAST はISM にくらべてグループメンバー数に制約がある一方で、グループ数に関するスケーラビリティに優れており、多地点ビデオ会議やネットワーク対戦型ゲームなど、多数のエンドユーザーがプライベートなマルチキャストグループに対して発信が必要な用途に有効である。

WIDE project では 1999 年度に v6 WG の活動として XCAST の研究開発を開始して以来、IBM、Alcatel、KAIST、Soongsil University 等との協調により、統一 XCAST プロトコルの策定と Internet Draft としての提案、∗BSD での XCAST6 実装、Soongsil University による Linux 上の XCAST6 実装との相互接続実験、WIDE 6Bone 上での試用などを行って来た。

2003 年度は、これまでの実績を元に以下の目標を掲げて研究開発活動を行った。

- XCAST 実装の堅牢化とプロトコルの洗練化
- X6-Bone（XCAST 実験網）の運用拡大によるノウハウの蓄積
- XCAST の啓蒙・普及活動
- XCAST アプリケーションの開発

本報告書では 2003 年の XCAST WG の活動を以下の順に報告する。

1. X6-Bone: XCAST NLA1 実証網
2. ミーティング実験
3. VNC for XCAST6 評価実験
4. グループ管理サーバ改良
5. 通信状況測定ツール整備
6. XCAST6 over UDP

第 2 章　X6-Bone: XCAST NLA1 実証網

X6-Bone は XCAST WG が 2002 年度から運用する Internet 上の仮想 IPv6 網である。

各種の実証実験を通じて判明した知見として、半透過トンネルによる漸近的展開手法は、ネットワーク管理者の手を煩わせることなしに、エンドユーザーが気軽にマルチキャストに参加できることがわかった。一方で、参加する端末のネットワーク接続状況と配送リストの順番によっては、マルチキャストグループ参加者全体でパケット配送遅延の増加やパケットロスの増加が著しくなる場合が多々あることがわかった。これらの現象の原因にはさまざまな要因が考えられ、解消する手法もいろいろなものが考えられる。

しかし、根本的には XCAST を解釈し分岐を行うルータの展開が、インターネット全域的に行われることであるとする立場から、これに近い論理的トポロジーが構成できる XCAST ルータによる IPv6 ネットワークを構築することとした。XCAST WG ではこのために WIDE 6Bone の pNLA1 より 3ffe:51b::/32 の割付を受け、これを各種協力団体に/40 で配布する実験を開始した。

ネットワークトポロジーは、慶応大学湘南藤沢キャンパス内に設置した XCAST6 ルータ（xgate.sfc.wide.ad.jp）をハブとして協力団体を v6/v4 トンネルで接続するスター型としている。協力団体には/40 が割り付けられ、これをそれぞれの XCAST6 ルータで受けた後、構成員に/48 で配布する。このようにすることでトラフィックを局所化するとともに、XCAST の分岐がルータで必ず起こるネットワークを仮想的に構築する。

現在、表 2.1 の 5 団体を収容している。のちに述べる拡大 XCAST Meeting で xgate を経由した通信を行うとともに、各団体が独自で企画運営している Meeting ではそれぞれの団体のルータが XCAST 分

表 2.1. X6-Bone 参加団体とプレフィックスの割り当て

参加団体	割り当てプリフィックス
CBUG（調布／西東京*BSD ユーザーグループ）	3ffe:051b:4400::/40
S*BUG（Shikoku *BSD Users Group）	3ffe:051b:4500::/40
NoBUG（北海道*BSD ユーザーズグループ）	3ffe:051b:4600::/40
BAY-BSD（FreeBSD と飲もう会・BAY-BSD ユーザーズグループ）	3ffe:051b:4700::/40
NBUG（名古屋*BSD ユーザーグループ）	3ffe:051b:4800::/40

岐を行い、トラフィックの局所化が行われているものと考える。

第 3 章　ミーティング実験

3.1 WG 定例ミーティング

2002 年度から引続き、WIDE 6Bone を用いて週に 1 回のミーティングを行っている。2003 年に行った定例ミーティングの回数は 14 回であった。ミーティング内で話し合われた内容は、主に各研究グループ内の研究紹介や各種運用実験の連絡などであった。

3.2 Midnight XCAST meeting

定例ミーティングとは別に、XCAST の普及を促進するためのイベントをかねて、WIDE プロジェクトメンバーの外からの参加者を募った Midnight XCAST meeting を開催した。主に国内各所の地域の BSD ユーザー会を中心に募集を行い、4 回のミーティングを開催した。また、Networld+Interop 2003 のイベント「BSD なひととき」を XCAST6 で中継した。

これらのイベントでは、定例のミーティングと比べて、参加ノード数が増え（最大 10 拠点）、参加者のネットワーク環境が多様になる。このため実装のバグが発覚しやすく、運用上のノウハウも得られやすかった。

3.2.1 6th MIDNIGHT XCAST MEETING

2003 年 2 月 15 日に開催された。 XCAST/FreeBSD が 8 サイトを越えると vic、rat での転送ができなくなるバグが報告された。

3.2.2 7th MIDNIGHT XCAST MEETING

2003 年 5 月 10 日に開催された。前回のバグのトラブルシュートのために、参加ノードを増やすため、ユーザ会ごとでなく個人ごとに参加してもらい、挙動の検証を行った。検証の結果、バグを発見することができ、後日修正をパッチとして作成し配付した。

3.2.3 BSD なひととき

2003 年 7 月 2 日に開催された Networld+Interop 2003 のイベント「BSD なひととき」を、会場の幕張メッセより X6-Bone 経由で全国の BSD ユーザーグループに中継し、中継先からのフィードバック映像を会場スクリーンに表示した。

3.2.4 8th MIDNIGHT XCAST MEETING

2003 年 8 月 2 日に開催された。NBUG には可搬型 XCAST 環境を構築していただき、ADSL が敷設された飲食店から、複数映像ソースの同時 XCAST 中継を行っていた。このように中継環境を持ち運んで運用する場合は、別途 IRC などの連絡用の手段を用意するべきではないかという問題提起がなされた。

3.2.5 9th MIDNIGHT XCAST MEETING

2003 年 11 月 1 日に開催された。NBUG は Air H" を用いての中継を行ったが、帯域不足で xcgroup がタイムアウトしてしまう問題が起こった。また、最高で 10 ノードが通信した際には、ノード間の通信が途絶し、途絶状況を把握するため ping などを行っても、unicast での通信すらできないという現象が報告された。このような場合、全体の通信状態を俯瞰的に把握し、問題解決する手段が必要であるという問題提起がなされた。

第 4 章　VNC for XCAST6 評価実験

4.1 グループ通信と XCAST6

XCAST6 は、受信者端末が未配送の受信者に対してパケットを送信し直す動作を行うため、網上のルータが XCAST6 に未対応でも擬似的なマルチキャスト転送を実現する。このことは、キャリアや ISP 等網の運営主体によるマルチキャストサービスの提供状態に依存せずに、インターネット到達性が保証されている全ての受信者を通信の相手に指定できることを意味する。この特徴は、人と人のコミュニケーションのための通信、ことさら複数の参加者が同時に即時的な情報を共有するグループ通信にとって有効である。逆にグループ通信は、XCAST6 のキラーアプリケーションであり、グループ通信がインターネットで広く実施されることが XCAST6 普及の重要な要素である。

4.2 VNC for XCAST6 の問題点

遠隔のコンピュータのデスクトップ画面を複数の利用者で共有するグループ通信アプリケーションとして、VNC（virtual network computing）[1] を XCAST6 に対応させたものがある。

このグループ通信アプリケーション（VNC for XCAST6）をインターネットで実用・展開するには、次に挙げる課題がある。

課題 A VNC のセッション広告およびグループ管理方式の確立

課題 B ヘテロな受信者の能力や利用可能な帯域に応じた流量制御の確立

課題 C 他の TCP フローと公平な帯域共用を実現する輻輳制御の確立

まず課題 A について説明する。VNC は、コンピュータのデスクトップ環境を提供するサーバ（Xvnc）とサーバが提供する画面を表示するクライアント（vncviewer）で構成される。Xvnc が提供する画面を XCAST6 で共有するには、Xvnc に受信者（vncviewer）の IP アドレスのリストを教える必要がある。つまり、新たな受信者が VNC の画面の共有に加わりたい場合には、自らの IP アドレスを Xvnc の受信者リストに追加する必要がある。このセッション確立に関する手続きを何らかの方式で実現する必要がある。

課題 B、C は、トランスポートに対する要求である。VNC では、画面更新のたびに画面変更部分の情報（以後 RFB と略記）をクライアントに伝送する。画面の更新頻度に応じて RFB の伝送に必要な帯域が大きくなる。インターネットでは、受信者毎に利用可能な帯域は様々であり、その帯域を越えて RFB のデータグラムを送信してもデータグラムを構成する全てのパケットは正常に伝送できない。結果として、受信者に RFB が正常に伝送できないので、画面を共有して協働作業をするというグループ通信の要求を満たすことができない。この問題を解決するため、受信者の利用可能な帯域に応じた流量制御が必要となる。また、インターネットは公共のものであるため、送信者と受信者の間の配送路上のすべてのリンク上で、他の TCP フローと公平な帯域共用を実現する必要がある[2]。

課題 B、C の要求を満たすトランスポートを設計するためには、次の分析が必要となる。

分析 画面更新部分の情報の送信速度、画面の更新頻度が参加者間の協働作業にどのような影響を与えるか

この分析の結果、利用者間での協働作業のためには、最低限必要な画面の更新頻度が存在することが確認できれば、この画面の更新頻度以上に RFB を転送する場合、RFB の転送を間引くことが可能であることがわかる。すなわち、課題 B、C を解決するトランスポートとして RFB の転送を間引く方式が採用できる。

4.3 2003 年春合宿における評価実験

我々は、上記の分析の実施、課題 A の解決を主な目的として、2003 年春合宿において「VNC for XCAST6 評価実験」を実施した。この結果、640 × 480 のコンピュータ画面で、MagicPoint によるプレゼンテーション表示や ImageMagic 地図画像編集操作を、VNC を用いて共有する場合、RRE 方式[3] で転送した場合に、1 Mbps が必要であるという結果を得た。

1　http://www.realvnc.com/
2　S. Floyd, “Congestion Control Principles”, RFC 2914, September, 2000
3　RRE ＝ランレングス圧縮方式。同一のビット列が圧縮走査方向に連続する場合、高い圧縮効果が得られる。

また、合宿期間中の BOF で、1 台の Xvnc サーバ上で複数の VNC セッションを提供する場合、既存のグループ管理ツールである xcgroup を用いることはできないことを確認した。xcgroup は、グループ管理サーバである xcgroupsrv.cgi と http で通信し、受信者の IP アドレスを登録・取得し、同一端末上のアプリケーションに MBUS とよばれるアプリケーション通信路を用いて通知する。xcgroup が用いる MBUS のメッセージでは、異なるセッションを識別することができないので、1 台のローカル端末で複数の異なるセッションの受信者リストを通知できない。この対策としては、個々の Xvnc プロセスがグループ管理サーバと直接通信する方式で対応した。

これらの結果を踏まえ、今後、課題 B、C を解決するトランスポートの設計・実装、課題 A を克服するセッション管理・グループ管理を実現するソフトウエアを実装する計画である。

なお、実験システムの構成は、Camp-net の報告書に記載されているので参照していただきたい。

第 5 章　グループ管理サーバの改良

xcgroup サーバは、XCAST6 を使ったグループマルチキャスト通信を行う際に、参加メンバーの IPv6 ユニキャストアドレスを管理するサーバである（図 5.1）。グループに参加するノードは、あらかじめ xcgroup を起動し、明示的にグループに参加（join）する。このとき、xcgroup サーバ内に join したノードのアドレスと参加時のタイムスタンプを有するエントリがメンバーリストに追記される。XCAST6 通信を行う時には、xcgroup サーバのメンバーリストに基づき、join 中のノードアドレスが XCAST6 ヘッダの宛先リストに格納され、パケットを送信する。

また、XCAST6 ノードにて起動している xcgroup を終了すると、xcgroup サーバへ離脱（leave）メッセージが送信され、メンバーリストから削除される。

5.1 xcgroup サーバへの syslog 機能の追加

従来の xcgroup サーバは、内部的に保持しているメンバーリストへのアドレス追加／削除を遂行するのみで、現在のメンバー情報しか取得することができなかった。そこで、xcgroup サーバに join/leave 受信時のログを syslog 経由でダンプするよう改変し、join/leave 受信時の挙動をトレースできるようにした。

この機能は、xcgroup サーバを起動する計算機にて syslog デーモンが稼働していることが前提になるが、syslog デーモンが動いている計算機にて、facility を “LOCAL0”、log level を “info” としたエントリを syslog.conf に設定すると、xcgroup サーバがログをダンプするようになる。具体的な設定例は以下のようになる。

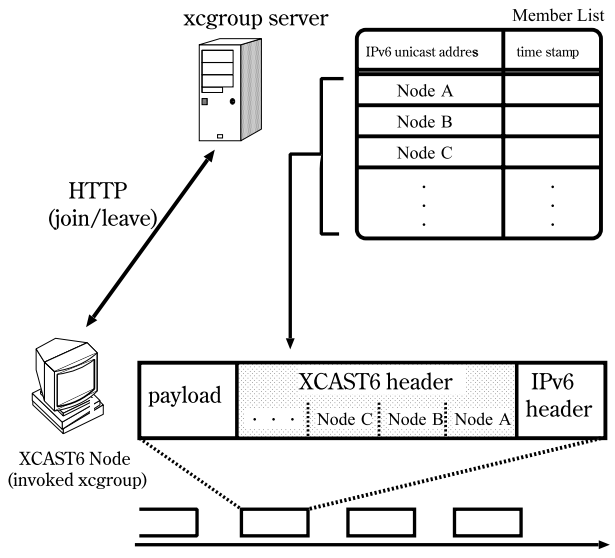

図 5.1.　xcgroup の基本動作

表 5.1. syslog で記録されるノードの状態

ステータス	ログに記録されるタイミング
join	初回 join コマンド受信時
leave	leave コマンド受信時
timeout	time stamp + timeout 値 < 現在の time stamp 時

/etc/syslog.conf

```
...
local0.info    /var/log/xcgroupsrv.log
```

あるノードに関するメンバーのリストと個々の状態を表 5.1 の 3 つと定義し、ログ記録を行う。

join はグループへ参加中のノードの状態を表し、xcgroup サーバがそのノードからの join コマンドを最初に受信した時にロギングされる。これは、ノードにて起動している xcgroup が約 60 秒毎に join コマンドを送信しているからである。

leave はグループを離脱したノードの状態を表し、xcgroup サーバがそのノードから leave コマンドを受信した時にロギングされる。leave コマンドはノードにて起動している xcgroup が終了（SIGINT 受信時）した時に 1 度だけ送信される。

timeout はメンバーリストにエントリされているが、タイムスタンプが指定されたタイムアウト値を超えている状態を表す。タイムアウト値は変数 “XCGroupTimeout” にて設定され、メンバーリストにあるタイムスタンプとタイムアウト値を足した値が、現在のタイムスタンプを越えている場合に timeout と判定され、ロギングされる。これは、

- ネットワーク障害
- ノード障害
- xcgroup の異常終了

などの理由により、leave コマンドが正常に送信されず、メンバーリストにエントリされ続けることを回避するためのステータスである。

なお、“XCGroupTimeout” は xcgroupsrv.cgi に記述されており、デフォルトで 180 秒（3 分）となっている。

5.2 join メンバーの推移グラフ

上記の syslog 機能を元にログファイルをベースにした join 数を視覚的に表示するスクリプト “xcgstats.cgi” を新たに作成した。このスクリプトを利用する際には、syslog.conf に設定したログファイルを指定する必要がある。

xcgstats.cgi

```
#!/usr/bin/perl
...
 $FILE_LOG ='/var/log/xcgroupsrv.log'
```

このスクリプトはログファイルを解析し、ステータスログがダンプされた時刻を X 軸に、あるグループに join しているノードの累計を Y 軸とする時系列グラフである。図 5.2 はそのサンプル画像である。

グラフ部は Perl の GD::Graph モジュールを使用し、png 形式で出力する。その下はステータス部であり、ログファイルの各エントリに一致する。各フィールドは以下の通りである。

- DATE: ロギングされた時刻
- ADDRESS: 対象となるノードの IPv6 アドレス
- STATUS: ノードのステータス

5.3 今後の課題

現在のバージョンではログファイル全体を走査し、マッチしたグループ名のログエントリをすべて表示する仕様になっている。そのため、ログファイルが肥大化すると、ログ走査のオーバーヘッドが高くなり、ステータス部の表示が冗長になってしまう。今後は日付指定や時間指定機能を付加し、特定の間隔の推移を表示できるよう改良すべきである。また、xcgroup サーバ自体も単一のログファイルにダンプするのではなく、グループ毎にログファイルを作成するよう改良することも必要となるだろう。

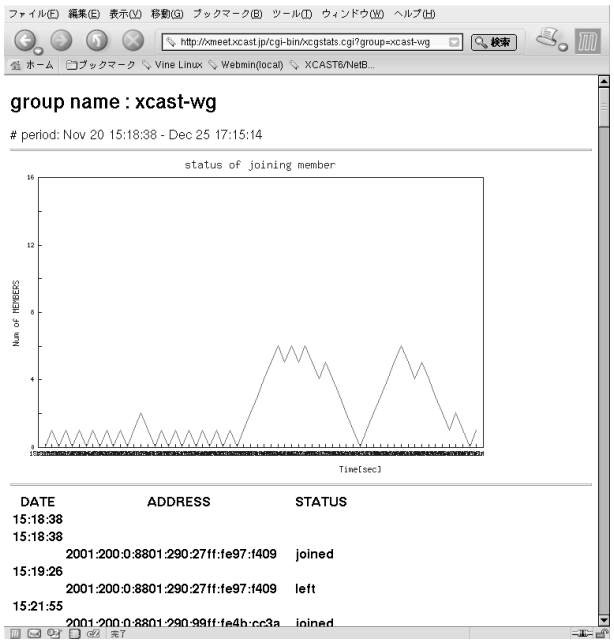

図 5.2.　join メンバー数推移グラフ例

第 6 章　通信状況測定ツール整備

6.1 XCAST トラフィックのモニタリング

6.1.1 目的

XCAST WG では、毎週 XCAST を用いてミーティングを行ってきた。また、WIDE X6-Bone を構築し、組織／個人に対して IPv6 ならびに XCAST の接続性を提供してきた。しかし、現在までトラフィックのモニタリングを行っておらず、XCAST を用いたトラフィックがどれくらい流れているのかは記録されていない。また、XCAST を用いて、多地点でミーティングを行っている際に障害が発生した場合、トラフィックのモニタリングを行っておらず、原因を突き止めることが困難であった。

そこで、XCAST WG では、今後の X6-Bone の運用、XCAST の利用調査を目的として、DTCP サーバとして公開している XCAST ルータ（pc7.fujisawa.wide.ad.jp）で XCAST のみのトラフィックの流量のモニタリングを行うことにした。

6.1.2 実装

通常のトラフィックの流量をモニタリングする場合、snmp を用いて MIB にアクセスし、mrtg によりグラフ化する。しかし、XCAST トラフィックのみの流量をモニタリングする場合、MIB を用いることはできない。そこで、libpcap ライブラリを用いてパケットをキャプチャーし、XCAST トラフィックの流量のモニタリングを行うことにした。IP ヘッダのオプションヘッダを参照し、hop-by-hop オプションの値をチェックし、XCAST トラフィックかどうか判別し、XCAST パケットの流量のみを記憶する。そして、rrdtool を用いて、定期的に送信、受信した XCAST パケットの流量をグラフ化する。また、トラフィックのモニタリングは、4 時間毎、1 日毎、1 月、半年毎のスケールでグラフを作成する。

6.1.3 XCAST トラフィック

以下に、12 月 11 日、18 日に XCAST WG に XCAST を用いてミーティングを行った際の pc7.fujisawa.wide.ad.jp でのトラフィックの流量を示す。

6.1.4 今後の課題

今後の課題として、以下を挙げる。

- xcgroup の情報と統合し、グループに参加、離脱するメンバーの影響がわかるようにする。
- VIC、RAT などのアプリケーション毎にモニタリングを行う。

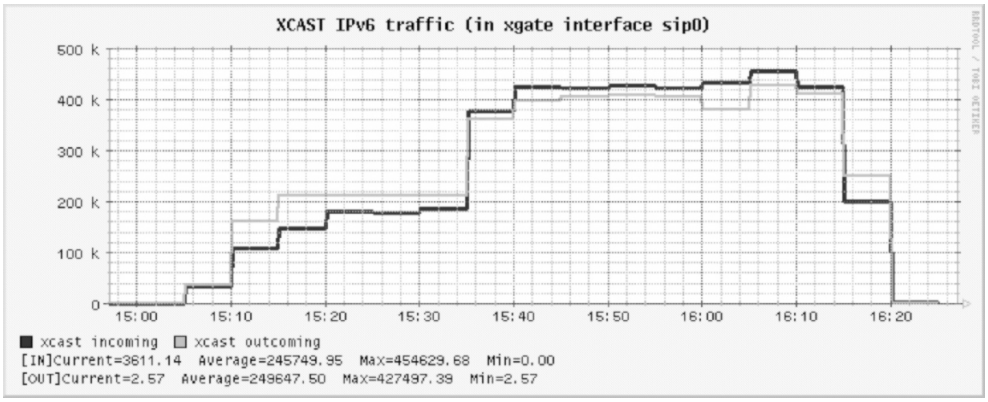

図 6.1. 12 月 11 日のミーティング時の XCAST トラフィック

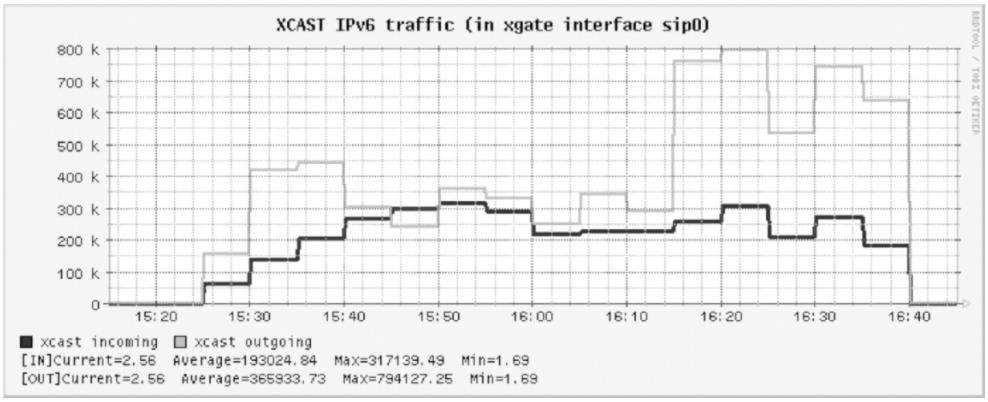

図 6.2. 12 月 18 日のミーティング時の XCAST トラフィック

- 長期的に XCAST トラフィックのデータを蓄積する。

第 7 章 XCAST6 over UDP

Abstract

XCAST (Explicit Multicast) is a datagram delivery system suitable for multi-party applications like tele-conferences and network games. From late 1990's many volunteers make technical works on the Internet as well as discussion and BoF in IETF. However, we cannot get "rough consensus" in IETF multicast community that prevents us from getting necessarily IANA resources to start broader deployment of XCAST.

XOU (XCAST6 over UDP) is a substitute system of basic-XCAST that delivers XCAST6 datagram as UDP datagram without any official IANA resource. In XOU, we use a special SSM tuple instead of ALL_XCAST_NODE multicast address and embed other information including list of destinations at the position from option header to the lead of UDP payload.

Using XOU, we can avoid IANA resource problem that is obstacle against acception of XCAST for several open source OSes.

7.1 Introduction

XCAST (Explicit Multicast) is yet another multicast protocol that is suitable for multi-party applications for small group such as tele-conference and network games. While conventional multicast and SSM have scalability in terms of numbers of receivers in each group, XCAST is scalable in terms of number of the group on the network. In other word, XCAST is complement of conventional ASM and SSM.

Since 1999, volunteers in IETF collaborate in developing, and testing interoperability of implementations of XCAST protocol stack. Because the implementations have been released as open source, community was organized mainly in NetBSD, FreeBSD and Linux users in Asian countries. Especially, WIDE project started operation of X6-Bone (XCAST6 Bone) in 2002 and that connected 9 groups and individuals broadly distributed in Japan.

While hacking on the Internet, we have discussed to promote XCAST technology in Multicast Directorate, Routing Area Meeting and a BoF of XCAST itself. We could get understanding of area directors and IESG, but on the way of achieving rough consensus of whole of Multicast community in IETF, it prevented us from taking official IANA protocol resources necessarily to deploy the XCAST implementation.

Some guys in XCAST community are committer of *BSD and they want to easily distribute and deploy XCAST6 code in their current version and maintain XCAST6 system. However to do that, it is the largest obstacle that we don't have any type values of option headers for XCAST6.

XOU (XCAST6 over UDP) is an alternative way of XCAST to do without official IANA resource. XOU simulates UDP datagram. That means for ordinal nodes it looks UDP completely, while for nodes that know what XOU is, it can be found an explicit list of destination in the UDP payload and branch so that they forward the datagram as original XCAST nodes behave. Reorganized points from original XCAST to XOU are as follows.

- Instead of ASM address, ALL_XCAST_NODE, use the SSM channel (X6Bone_DUMMY_SOURCE, ALL_XCAST_SSM_ID).
- Shift XCAST information from option headers to the beginning of UDP payload.

7.2 Header format

XOU datagram organized as follows (Figure 7.1).

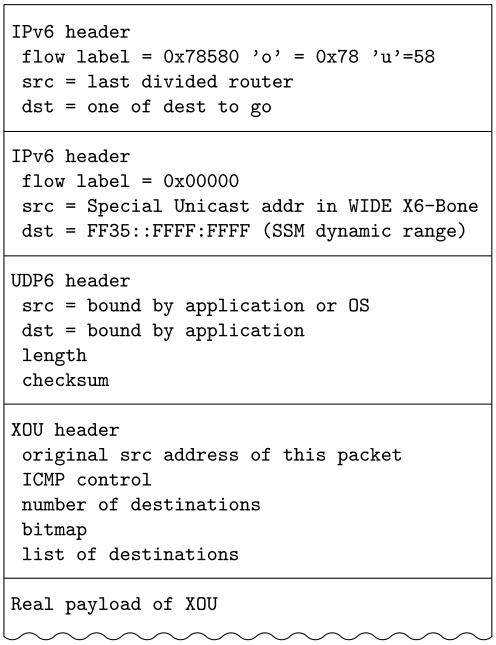

図 7.1. XOU datagram structure

The first IPv6 header is similar for semi-permeable tunneling of original XCAST6 protocol. In flow label field, we embed special value '0x78580' that means 0x and ascii code of 'o', 'u' and 0x0. For XOU router this is predefined as special value while it is no meaning for non-XOU router. However, other application can use this value independently. It's just a hint for XOU router. In the source address field of it, a unicast address of outgoing interface of last XOU branch router is indicated. And destination address is one of unicast address from explicit list of destinations that the datagram must reach.

The second header is also IPv6 header. That is a mark the datagram is XOU one. The source address of this header is X6Bone_DUMMY_SOURCE (not assigned yet) that is a special unicast address from the range assigned for WIDE X6-Bone network. WIDE XCAST WG will preserve it and never assign for any node. The destination address of the header is ALL_XCAST_SSM_ID (FF35::FFFF:FFFF) that is from the SSM range. As WIDE XCAST WG will keep uniqueness and persistence of X6Bone_DUMMY_SOURCE, we can treat

SSM channel (X6Bone_DUMMY_SOURCE, ALL_XCAST_SSM_ID) as a mark of XOU datagram as well as ALL_XCAST_NODE is in the original XCAST basic specification.

The third header is a UDP header. It includes source and destination port just same as ordinal UDP header. In length field of this UDP header, remained length of UDP payload including XOU header, which is described followings, is indicated. And in the checksum field, UDP checksum is calculated with zero-filled XOU header followings.

For non-XOU node, succeeding part of XOU datagram looks ordinal UDP payload. And whole the XOU datagram is ordinal UDP datagram with wrong UDP checksum. However, as XOU system there is the fourth header that includes

- the unicast address of original source node
- the number of destinations
- the bitmap of delivery status
- the explicit list of destination

After the XOU header, real UDP payload of the XOU datagram succeeds.

7.3 Datagram processing

7.3.1 Transmission

Sender must allocate and prepare the packet buffer (mbuf, skbuf, etc.) that is long enough to store two IPv6 headers, one UDP header and XOU header that fit for the number of destinations and real payload to carry. In the first header, substitute flow-label 0x78580, store outgoing interface address as source address and store first one of the destination list in XOU header in destination address. After storing all data, zero-fill the bitmap field temporarily, calculate UDP checksum and restore them.

7.3.2 Forwarding

Forwarding by non-XOU nodes By the node that does not know what XOU is, or that does not have any flow-label information for 0x78580, the datagram is forwarded as the unicast datagram. This behavior is similar with that of semi-permeable tunneling of original XCAST6. Even if flow-label information for 0x78580 registered on the router, XOU information is ignored and special unicast procedure for the label goes on.

Forwarding by XOU nodes XOU nodes register 0x78750 as a special label for XOU in protocol stack initialization. Receiving the datagram with this label, the node checks the datagram which is XOU packet or not. This is done by checking the second header that is IPv6 header with SSM channel (X6Bone_DUMMY_SOURCE, ALL_XCAST_SSM_ID).

If it is not proceeded by ordinal unicast procedures, or receiving XOU packet, the node skips UDP header and proceeds to XOU header. Checking bitmap for each destination, it looks up unicast routing table to find next hop entry, bundles destinations that have the same next hop and duplicates the datagram if two or more next hop entries are found. For each datagram, it updates bitmap and rewrites source and destination addresses of the first IPv6 header and forward it.

7.3.3 Receiving

If loop-back entry is found as a result of forwarding process based on XOU header, UDP protocol stack must be up-called with following exceptional handling.

1. Before calculating checksum, bitmap must be zero-filled.
2. Slide UDP payload pointer to next of XOU header in order for recv(2) to return real payload of XOU datagram.
3. Store original source address into sockinfo in order for recvfrom(2) to return correct source address of XOU datagram.

7.4 Security consideration

To prevent DDoS, branching XOU router MUST re-write source address of first IPv6 header to the one of outgoing interface. Without it, cracker is easily able to attack arbitrary node using ICMP error reply.

7.5 Detailed format of XOU datagram

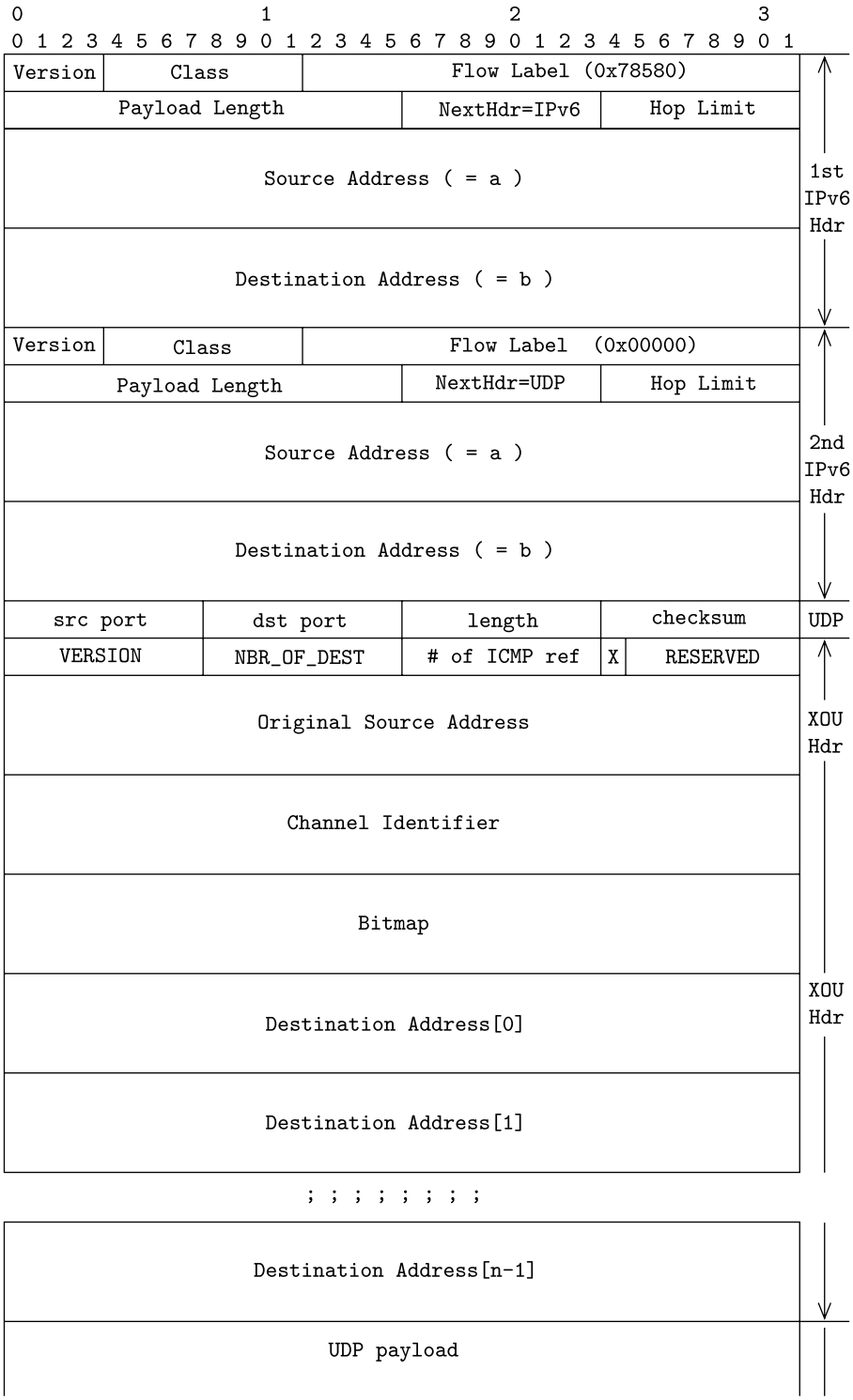

図 7.2. XOU detailed datagram format