マルチメディア通信と分散処理 86－6
グループウェア 26－6
（1998. 1. 29）

Host Information Protocol に関する研究

丹野 一† 木村 成伴, 海老原 義彦‡

†筑波大学大学院 工学研究科 ‡筑波大学 電子・情報工学系

〒305 茨城県つくば市天王台 1-1-1

{one,kimura,ebihara}@netlab.is.tsukuba.ac.jp

あらまし TCP プロトコルでは，送信したメッセージに対応する返答がタイムアウトになることにより，アクセス先のホストダウンを検出している．通常この動作には数分を要し，メッセージの再送処理のためにネットワーク資源が浪費される．そこで，本研究では，TCP アクセス監視によるダウンホストの検出，ホストアップ情報の伝達とダウンホストへの動的な状態監視によりホストアップを検出するなどの手法を用い，ドメイン内のホスト状態を各ホストが保持することで，ダウンホストへのアクセスを抑制する方式を提案する．また，この方式を用いることでダウンホストにアクセスしたユーザーの待ち時間を短縮でき，ネットワークへの負荷を減少できることを示す．

キーワード HIP, アクセス制御, ホスト状態監視, TCP

A Study of Host Information Protocol

Hajime Tanno † Shigetomo Kimura, Yoshihiko Ebihara‡

†Graduate School of Engineering ,University of Tsukuba

‡Institute of Information Sciences and Electronics, University of Tsukuba

1-1-1 Ten-nodai, Tsukuba, Ibaraki 305, Japan

{one,kimura,ebihara}@netlab.is.tsukuba.ac.jp

Abstract In TCP, a system detects that the destination host is down, only when the acknowledgment for a transmission message is timeout. Until the timeout occurs, in general, it takes several minutes, and the network resources are consumed because the source host tries to retransmit the message many time. This paper presents the access control method for down hosts, where each host retains the down or up status of all hosts in its domain. The down status is detected by monitoring the timeout of TCP accesses. The up one is by receiving host-up information and/or dynamic monitoring to the down hosts. For the proposed control method, we shows that the user waiting time to access to a down host is reduced and then the network load is decreased.

key words HIP, access control, host status monitoring, TCP

1 はじめに

近年のネットワーク環境の向上により，数多くのホストが相互に接続され運用される形態が一般的になっている．その構成が複雑になるにつれ，ネットワークの内部で障害が生じても，それを特定する事が困難なものになる．

現在のTCPプロトコルでは，アクセス先のホストがダウンしているとき，送信したメッセージに対応する返答がタイムアウトになることによりこれを検出している[1],[2]．通常この動作には数分を要し[3]，メッセージの再送によりネットワーク資源が浪費される．これを解消するため，当研究室ではドメイン内ホストのアップ/ダウン情報を各ホストが保持し，ダウンホストへのアクセスを防止するHost Information Protocol(HIP)及びその改良型である，Dynamic Host Information Protocol(DHIP)を提案している．しかし，これらの方式では，ホスト情報が実状とは異なる場合が生じたり，ホスト情報取得のために多くのパケットが流れたりするなどの問題があった．

本稿では，これら2方式について簡単に述べた後，TCPアクセスを監視してダウン状態を検出し，ホストアップ情報の伝達とダウンホストへの動的な状態監視によりアップ状態を検出するなどの改良を施した新しい方式を提案する．最後に，従来の方式との比較検討を行う．

2 HIPおよびDHIPの概要

2.1 Host Information Protocol

HIPが実装されたシステムでは，各ホストは，自分以外のホストのアップ/ダウン状態を記録しているHost Information Table(HIT)を保持している．メッセージを送出する際にはHITが参照され，ダウンホストに対するアクセスであった場合には直ちにホストダウンエラーを返して，メッセージの送出を中止する．これにより，ダウンホストに対する無駄なパケットの送出を防ぐことができ，パケットの再送が低減するとともに，ダウンホストにアクセスしたユーザの待ち時間を短縮することが可能となる．図1で，ホストH3のHITにはG1とH1がアップ，H2はダウンしていることが登録されている．例えば，H3からH2へアクセスを行うとき，HITを参照することでH2に対するアクセスはホストダウンエラーとなり，その時点でアクセスが中断される．

各ホストはブート/シャットダウンする時に，ブロードキャストによってそのアップ/ダウン情報(ホスト情報)を同一サブネットワーク内の全ホストに対して送出する．この情報を受け取ったホストは直ちにHITを更新する．図2は，ホストH2がシャットダウンする場合の動作例である．H2はダウンする時に，ダウン情報のブロードキャストを行う．これを受け取ったH1,H3,G1はそれぞれのHITにあるH2の状態を更新している．

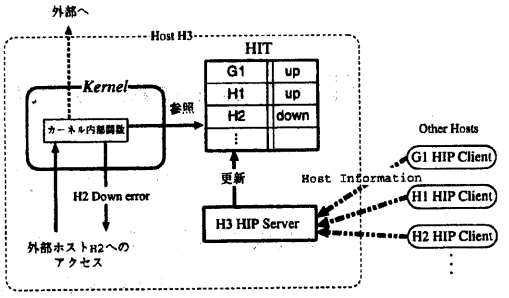

図1: HIPの動作

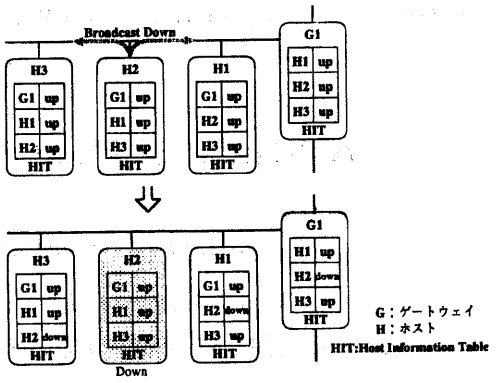

図2: HIPが実装されたサブドメイン

しかし，ブロードキャストはコネクションレス型のトランスポートプロトコルUDP(User Datagram Protocol)を利用するため，ホスト情報が全てのホストに対して確実に伝わる保証はない[4]．したがって，ネットワークもしくはホストの状態によっては，ブロードキャストされたホスト情報を受け取ることのできないホストがでる可能性がある．また，ホストがシステムの障害などによりダウン情報を送出せずに突然ハングアップした場合や，あるホストがダウンしている間に別のホストからダウン情報が送出された場合など，HITが実状を反映しない事態が生じる恐れがある．

HITに誤った情報が保持されている場合，HIPによるダウンホストへのアクセス制限が働かなくなるばかりか，アップ状態のホストに対しても通信が行えなくなるなどの問題が生じる．

ところで，ネットワークの経路情報はルーティングテーブルに保持されている．ルーティングテーブルは，ネットワークへの経路のためのテーブルとホストへの経路のためのテーブルとの二つに分けられる [5]．ホスト外部へのアクセスの際には，まずホストへの経路のためのテーブルを参照し，見つからないときは，ネットワークへの経路のためのテーブルを参照して経路を決定する．経路が見つからなかった場合はあらかじめ設定されている既定の経路を用いる．この二つのテーブルのうち，後者はARP(Address Resolution Protocol) テーブルとよばれている．このテーブルはARPによって動的に更新され，一定時間使用されていないエントリは削除される．このARPテーブル中のフラグにホストのアップ / ダウン状態を示すフラグ，HIPDOWNフラグを追加し，このフラグの部分をHITとしている．

HIPでは，HITの更新をホスト情報の伝達のみによっているため，HITに保持される情報は，ARPの動作によって削除されないような恒久的なエントリで無ければならない．しかし，この恒久的なエントリを作成することで，もともとARPテーブル中にあった他の情報が失われてしまう．また，ARPテーブルを永続的なエントリで占めることは，ARPテーブルの肥大化を招き，その弊害はHTPの実装上の問題にとどまらず，HIP以外のシステムにも影響を与える．また，ホストの追加や削除は手動で行わなければならないといった管理面の問題もある．

これらの問題を解消するために、定期的にサブネットワーク内のホストの状態監視を行うことにより、HITを更新するようにしたDynamic Host Information Protocolについて次節で述べる．

2.2 Dynamic Host Information Protocol

DHIPではホストが各自のホスト情報を配布するのではなく，各ホストが能動的にホスト情報を集めるように改良された．具体的には，各ホストがpingコマンドを用いて応答要求のパケットを送出し，定期的にサブネットワーク内の全ホストの状態の確認を行う．これによりアップ / ダウン状態の変化が検出された場合に，HITにその情報を直ちに反映させることができる．図3にはホスト状態の監視を行うDHIPの動作例を示す．

DHIPでは，動的なホスト状態の監視を行うことで，HIPのようにHITを永久的なエントリを用いて作成する必要がない．したがって，ARP本来の

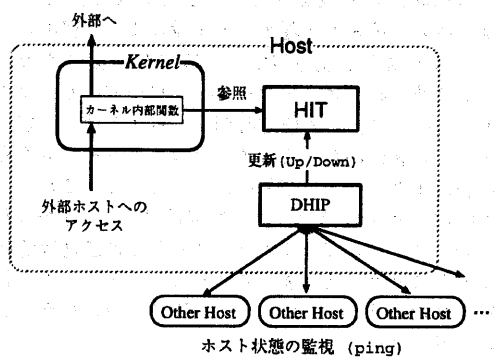

図 3: DHIPの動作

機能を活かすことで，ネットワーク内部のホストの追加や削除に対して自動的に対応が可能である．

定期的なホストの監視を行うためのpingコマンドの実行は，その回数や間隔をそれぞれのネットワークに応じて設定する必要がある．例えば，ネットワーク負荷が高くpingに対する応答が失われやすい環境では，ホスト監視の間隔を短く設定することでHITの信頼性を高めることができる．

この方式は，HITを常に最新の状態に保つことができるという点で有効であるが，サブネットワーク内の全てのホストから全てのホストに対して応答要求のパケットが定期的に送出されてしまうという問題が生じる．このため，サブネットワーク内のホストの数が多い場合，多量の応答要求のパケットが送出されることになり，ネットワークのスループットの大幅な低下を招いてしまう．一方，これを防ぐために監視の間隔を長く設定すると，今度はHITの更新が遅くなってしまう．最悪の場合，ダウンしていたホストが復旧しても監視間隔の時間だけそのホストにはアクセスできない．

次節以降ではこれまでに指摘した問題点を解消するための改良型HIPを提案し，その実装システムについて述べる．

3 改良型HIP

3.1 代理ホスト

ホストがアップ / ダウンする際に，DHIPではホスト情報のブロードキャストは行われていなかった．しかし，状態の変更があったことをより早く他のホストに通知するためには，HIPと同様，ホスト情報の伝達を行うことが有効である．ただし，ホスト情報の伝達にUDPのブロードキャストが用いられているため，情報の損失が起こる可能性はある．

この情報伝達の確実性を向上させる改良型HIPでは，ホスト情報の代理ホストを設ける．代理ホストはホスト情報を受け取ると，既定時間後にそのホスト情報を再度ネットワークにブロードキャストする．実際には，この既定時間内にホスト情報が変化すると情報が混乱するため，ブロードキャストする情報はホストアップ情報のみとしている．これは，2.1節で述べたように，ダウン情報が誤って伝わった場合の方が問題であるため，UDPでの伝達は危険であることと，後で述べるように，ホストダウン状態の検出はTCPコネクション確立の際に行うことができるためである．

この仕組みにより，ホストアップ情報が時間をずらして複数回流されることになり，情報の伝達を保証しないUDPであっても確実性を増すことができる．

なお，ホストアップ情報が正しく伝わらなかった場合に於いてもHITを正しい状態に保つために，さらにホストの状態監視を行うことが必要である．これについては3.3節で述べる．

3.2 ホストダウン状態の検出

改良型HIPでは，TCPコネクションタイムアウトの監視を行うことでホストダウンの検出を行う．すなわち，ダウンホストにアクセスする場合，このホストがHITにアップ状態であると記録されているならば，従来のシステムと同じようにそのホストに対してアクセスを行い，タイムアウトを待つ．TCPによるメッセージの送出は，UDPとは異なり，確実に伝達されることが保証されている．したがって，タイムアウトにより接続先ホストがダウンしていると判断でき，HITにホストダウン情報を登録する．これにより，二回目以降のアクセスでは，すぐにホストダウンエラーが返ることになる．なお，ホストダウン状態の検出にUDPコネクションが対象になっていないことに注意されたい．UDPは，情報が確実に伝達されることを保証していないため，接続先ホストがダウンしていても検出する手段はなく，また，そのような機構も必要はない．

この方法により，一度はダウンホストに対するアクセスがなされてしまうが，HITのホストダウン情報は正しいことが保証される．また，ホストダウン情報がブロードキャストされなくても，DHIPのようにアップ状態にあるホストに対して監視を行う必要はなく，また，不要なパケットを送信することなくホストダウン検出を行うことができる．図4は，カーネル内部において，TCPコネクションの監視を行い，コネクションタイムアウトが起きたときにはHITの更新を行う動作の例を示している．図中のカーネル内部の関数については，4.1節において説明する．

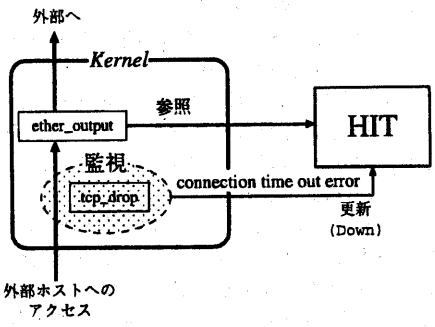

図4: ダウンホストの検出

3.3 ホストアップ状態の検出

3.1節で示した手順で，ダウン状態だったホストからのアップ情報を受け取ることができれば，HITが更新時点からそのホストへのアクセスが可能となる．しかしながら，2.1節で述べたようにこの情報が伝達する保証がないため，ホストアップ状態に変わったことを確実に検出する仕組みが必要である．このため，改良型HIPではダウンホストを対象とし，DHIPのように定期的にホスト状態を監視することでもアップ状態の検出をおこなう．前節で述べたように，アップ状態にあるホストに対する監視は行う必要がないことに注意されたい．図5は，ダウンホストに対する状態監視の動作例を示している．Host1は自身のHITを参照し，ダウン状態であると記録されているホストH3に対してICMPエコー要求を送出している．これに対する応答があった場合には，HITのホストH3に関するエントリを更新する．

通常のネットワークの状態において，ダウンホストの数はアップホストの数に比べてかなり少ない．このため，監視のためにネットワークに送出されるパケットの量はDHIPのそれよりも，かなり抑えることができる．ただし，アクセスが集中するホストがダウンした場合，そのホストに対する監視パケットが集中する恐れがある．これを避けるために監視パケットは一定間隔ではなく，ある範囲内でのランダムな間隔で送出することとする．

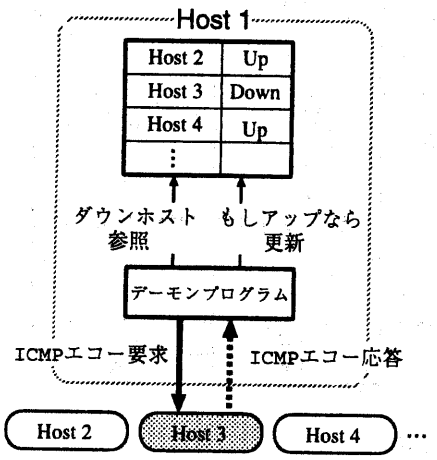

図 5: ダウンホストに対する状態監視

4 実装システム

本節では3節で述べたシステムをオペレーティングシステム FreeBSD 2.2.5-RELEASE に実装するプログラムの概要について述べる．プログラムはカーネルの変更とホスト監視のためのデーモンの二つに分けられる．

4.1 カーネルの変更

1. HIT の生成

   HIT は ARP テーブル内のホストの状態にホスト情報を格納する新たなフラグを追加することで実現した．今回の実装では，ARP テーブル内のフラグ (rt_flag) に HIPDOWN フラグを追加している．なお，この実装では，カーネル内部とデーモンプログラムによって HIT を更新する際に，直接 ARP テーブルを参照し，HIPDOWN フラグのみを書き換えるため，ARP テーブル中の他の要素には影響を与えない．

2. ダウンホストへのアクセスの抑制

   FreeBSD 2.2.5-RELEASE では，イーサネットを利用した外部ホストに対する通信は，最終的にカーネル内部の関数 ether_output を通じてネットワークに送出される．この機構を利用し，ダウンホストへのアクセスを抑制するために，ether_output 関数で HIPDOWN フラグを検査し，HIPDOWN フラグの立っているホストに対する通信はこの段階でホストダウンエラーを返すようにした．これにより，ユーザーの待ち時間は数分から数ミリ秒に短縮される．

3. TCP アクセスタイムアウトの監視

   FreeBSD 2.2.5-RELEASE では TCP のタイムアウトが起こった場合，カーネル内部の tcp_drop 関数が実行される．この関数においてタイムアウトの原因がコネクション確立時のタイムアウトであるか否かを判別し，その場合 HIT に HIPDOWN フラグを立てるようにした．

4.2 監視デーモン

1. HIP ダウンホストの監視

   本システムでは，HIT を参照しダウンホストに対してのみ監視を行う (図 5)．ホストの監視には，ping コマンドでも利用している ICMP のエコー要求メッセージをダウンホストに対して送ることでその応答を待つ方法を用いた．また，監視の間隔は一定ではなく 3 分から 7 分までの 1 秒刻みのランダムな間隔で行うようにした．ダウンホストに対して ICMP エコーパケットを送出する際には，そのままパケットを送出しようとしても HIPDOWN フラグがそのホストに対して立っているため，無条件にホストダウンエラーとなってしまう．それを避けるために，事前に HIPDOWN フラグを落とし，エコーパケットを送出した後，その応答を待つ．もし，応答が一つも返って来ない場合には再度HIPDOWN フラグを立てるようにした．

2. ホストアップ情報の監視

   HIP におけるホスト情報のブロードキャストとほぼ同じであるが，このシステムではホスト情報はアップ情報のみである．各ホストは，アップ時にホストアップ情報をブロードキャストするプログラムを実行する．ホストアップ情報を受け取ったホストは，HIT の更新を行う．また，信頼できるホストのデーモンプログラムには，ホスト情報の代理送信機能を設ける．これは，受け取ったホストアップ情報を一定時間後に再度ネットワークに送出するものである．

   このホストアップ情報がネットワークの障害などで受け取れない場合でも，前項で述べたダウンホストに対する監視を行っているため，その障害がなくなった後に，応答要求の返事が戻って来れば，ホストアップの状態であることを知ることができる．

比較項目		HIP	DHIP	改良型HIP
ホスト状態が変化した場合にネットワークに流される情報	ホストアップ情報	有り	無し	代理ホストからの情報も有り
	ホストダウン情報	有り	無し	無し
HITの更新タイミング(アップからダウン)		ホストダウン情報を受信した場合	アップホスト監視に対する応答がない場合	TCPコネクションタイムアウトを起こした場合
HITの更新タイミング(ダウンからアップ)	ホストアップ情報の受信	有り	無し	有り
	ダウンホスト監視に対する応答の受信	無し	有り	有り
ホストの状態監視のためにネットワークに流されるパケット数 N:全ホスト数 n:ダウンホスト数 c:応答要求パケット数		無し	(N-n)(N-1)c	(N-n)nc
HITに矛盾が生じた場合の復旧		手動	自動	自動
ホストの追加,削除に対する対応		手動	自動	自動

表 1: 各方式の比較

5 比較検討

従来の方式であるHIPおよびDHIPと改良型HIPの比較を表1に示した．表からわかるように改良型HIPでは従来の方式に比べ，ホスト状態監視のために必要な通信量を減らしつつも，HITの信頼性の向上が得られている．加えて，ネットワーク内部のホストの追加や削除があった場合にも，自動的に対応できるようになっている．

6 おわりに

本研究では，サブネットワーク内の各ホストが各々のホストのアップ/ダウン情報をHITに保持し，通信を行う際にそのテーブルを参照することでダウン状態にあるホストへのアクセスを抑制し，無駄なパケットの再送を抑えるとともにユーザの待ち時間を短縮するHIPおよびDHIPシステムの問題点を指摘し，これを克服した改良型HIPシステムを提案した．この方式では，ダウンホストに対する監視，TCPタイムアウトを監視することによるホストダウンの検出，ブロードキャストされるホスト情報の代理ホスト機能などが導入されており，この結果，HIPやDHIPよりも，ホスト情報の伝達の確実性が向上し，かつHITの信頼性を高めている．

本システムでは，サブネットワークの外部からダウンホストにアクセスを行った場合にTCPタイムアウトが起こってもサブネットワークのゲートウェイホストのHITは更新されない．これは，そのホストが主体となっているTCPコネクションのタイムアウトは監視しているが，ゲートウェイのように単にパケットを中継している場合は関与していないことに起因する．したがって，このような場合にもHITの更新を行うようにするには，ゲートウェイホスト用に特化した監視を行う必要がある．これについては改良の必要がある部分である．

参考文献

[1] Douglas E. Comer and David L. Stevens: *"Internetworking with TCP/IP Vol. 2 Design,Implementation, and Internals"*, Prentice-Hall, Inc., 1991. 邦訳 TCP/IPによるネットワーク構築 Vol. 2 設計，実装，内部構造.

[2] Dimitri Bertsekas and Robert Gallager: *"Data Networks Second Edition"*, Prentice-hall International, Inc., 1992.

[3] Douglas E. Comer: *"Internetworking with TCP/IP Vol. 1 Principles,Protocols,and Architecture Third Edition"*, Internals Prentice-Hall, Inc., 1995. 邦訳 第3版 TCP/IPによるネットワーク構築 Vol. 1 原理，プロトコル，アーキテクチャ.

[4] Kevin Washbrun and Jim Evans: *"TCP/IP: Running a Successful Network"*, Addison-Wesley Publishing Company, Inc., 1993. 邦訳 TCP/IPバイブル.

[5] Samuel. J. Leffler, Marshall K. Mckusick, Michael J. Karels and John S. Quarterman: *"The Design and Implementation of the 4.3 BSD UNIX Operating System"*, Addison-Wesley Publishing Company, Inc., 1989. 邦訳 UNIX 4.3BSDの設計と実装.