原文：Anonymous Web Transactions with Crowds, Communications of the ACM, Vol.42, No.2, pp.32-38 (Feb. 1999)より許可を得て翻訳

# 名前を隠して仲間とWebトランザクション

Michael K. Reiter reiter@research.bell-labs.com
Aviel D. Rubin rubin@research.att.com
翻訳：安藤　進 sando@twics.com

Webサーバの LOG ファイルは，訪れたユーザに関する情報で一杯だ．まず，各訪問者のアクセス内容が記録される．そのほかに，ユーザの IP アドレスもある．IP アドレスから，ユーザのインターネットドメイン名，職場，地理的な場所もだいたい分かる．さらに，ユーザが使用しているプラットフォームの種類，このサイトにくる前にいたWebページも分かる．少し努力すれば，次に訪れようとするサーバも推測できる[7]．ブラウジングセッション間でユーザの IP アドレスが変更されても（たとえば，DHCP を使用して動的に IP アドレスを割り当てる[4]），Webサーバは同じユーザによる複数のセッションをリンクできる．最初のブラウジングセッションでユーザのブラウザに独自のクッキーを埋め込み，それ以降のセッションでそのクッキーを取り出すのだ．さらに第三者も，これとほとんど同じ監視機能が利用できる．たとえば，ユーザの ISP やローカルゲートウェイ管理者は，ユーザの通信内容をすべて覗くことができる．

> 姿をくらますには群集に紛れ込むことだ．
> だって，匿名性は仲間が好きだから

このような監視機能はユーザプロファイリングを可能にする．多くの企業や消費者はこれを役立つツールと見なす．ユーザプロファイリングを利用すれば，Webサーバは各ユーザごとの管理が可能になるし，企業は従業員の活動を個別に監視できる．しかし，これはユーザプライバシーにマイナスの効果を与える．プライバシーのないことがインターネット通信の特徴であるとよくいわれるが，インターネット通信ほどユーザの個人的な情報がこれほど幅広く収集されかつ暴露されたことは，歴史上かつてなかった．こうした状況を踏まえて我々は，Webユーザが自分自身に関するどのような情報をだれに知らせるのかを制限できるようにすべきだと主張したい．そのための手段として，P3P（Reagle と Cranor の「プライバシー選択プラットフォーム」☆1を参照）と呼ばれる手法がある．これを利用すると，ユーザが自分自身のプライベートな情報をWebサーバに与える条件について交渉できる．しかし，適用対象はサーバだけ（プライベート情報を入手可能なそれ以外の者は対象外）であり，制限できる情報が限られる（たとえば，ユーザの IP アドレスは対象外）．しかも法的な拘束力がないし，現在あまり普及していないので，ほかの手段も考慮すべきだと考える．

この論文の目的は，Crowds と呼ばれるシステムを紹介することである．これは第三者にプライベートな情報を漏らさずにWebから情報検索を可能にするシステムである．Crowds の目的は，ブラウジングを匿名で可能にすることで，ユーザに関する情報とユーザが検索する情報をWebサーバや第三者から隠すことである．Crowds は，ユーザの識別が潜在的に可能となる情報をWebサーバに知られないようにする．ユーザの IP アドレスもドメイン名も知らせない．そのほか，ユーザがここを訪れる前のページやユーザが使用しているプラットフォームなどの情報も知らせ

☆1 Reagle, J. and Cranor, L.F., 安藤　進 (訳): プライバシー選択プラットフォーム, 情報処理, Vol.40, No.7, pp.723-729 (July 1999).

ない．

Crowdsは，関心を持つ者がユーザのマシンから送られるメッセージを監視できる場合に対応する保護も提供する．たとえば，ユーザが訪問中のサイトをユーザのゲートウェイ管理者に見えないようにする．Crowdsはこれを実現するためにほかのユーザのマシンからWebリクエストを発行するので，ユーザに固有の情報をほかのユーザに見えないようにする．

## Crowdsの仕組み

Crowdsは，Webユーザを地理的に多様なグループにまとめる．このグループを「crowd」と呼ぶ．グループのメンバに代わり，crowdがWebトランザクションを実行する．各crowd内では，ユーザのローカルマシンの「jondo」と呼ぶ（「ジョン・ドウ」と読む．氏名不詳者という意味）プロセスでユーザを表現する．ユーザがcrowdに参加する場合は，最初に自分のローカルマシンでjondoを起動する．次にjondoは，プロトコルを起動し，crowdに参加する．こうしてjondoは，現在のcrowd参加メンバを知ることができる．ほかのメンバも新規参加のjondoを知る．

これでユーザは，jondoとしてcrowdへの参加が承認されたことになる．それからjondoは，crowdを使用してWebサーバへリクエストを発行する．そのとき，だれがリクエストを発行したのかをWebサーバとほかのcrowdメンバに知られないようにしなければならない．そのためにユーザは，すべてのネットワークサービスに対するプロキシとしてローカルjondoを使用するようにブラウザを設定する．こうすると，ブラウザによるすべての通信がjondoに直接送られるようになる．ユーザがブラウザ経由でURLを要求すると，そのURLに対するHTTPリクエストは，要求されたURLに対するエンドサーバではなくjondoに転送される．jondoは，このリクエストを受け取ると，crowdのメンバを無作為に選び出し，そのメンバにリクエストを転送する．crowd内では，あるメンバがほかのjondoからリクエストを受け取ると，そのメンバはほかのメンバを無作為に（どのcrowdメンバも同等の確率になるような無作為抽出法で）選んでそのメンバにリクエストを転送するか，最終的な宛先であるエンドサーバにリクエストを送信する．ここでは，エンドサーバへの送信よりjondo転送が優先される．システム全体のパラメータ$pf > 1/2$は，エンドサーバへの送信よりjondo転送を選ぶ確率が高いことを示す．$pf$の値がシステム提供の匿名プロパティに与える影響については後述する．

要するに，ブラウザ経由で発行されたリクエストは，いくつかのjondoによって中継されながら最後にエンドサーバに届けられることになる．この一連の経路を「パス」と呼ぶ．Crowdsプロトコルの重要な特徴は，リクエストがパスの各「ホップ」で渡される形式が同じであることだ．そのため，各jondoは，発信元のjondoがリクエストの原発信者であるのか，それとも単なる中継者であるのかを区別できない．これは，Crowdsの匿名性を実現する重要な特徴である．さて，Webサーバがリクエストに対する応答（HTMLページや画像など）を返すと，パスの各jondoが先ほどの発信元を受信先に変えて中継する．こうしてパスを逆にたどりながらリクエストの原発信者に送られる．リクエストを発信したjondoが応答を受け取ると，ブラウザに応答を返すことでユーザに通知する．crowdのパスを図-1に示す．

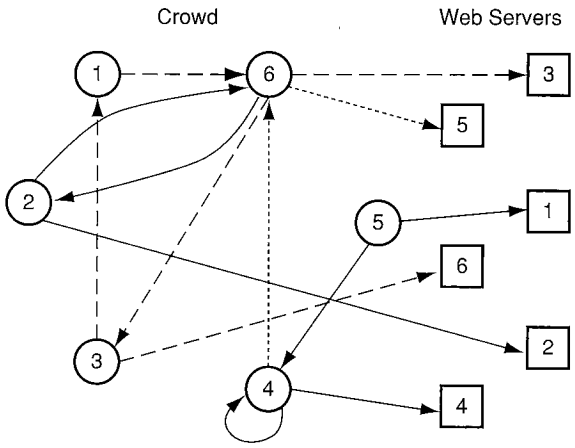


図-1　crowdのパス．各パスの発信者とWebサーバのラベルは同じ．crowdメンバから始まりWebサーバで終わる一連の同じ種類の連続矢印がcrowdのパスを示す．

初回以降同じjondoから発信されたリクエストは，初回と同じパスを経由してWebサーバへ送られる．宛先のサーバが前回と異なっていてもパスは同じである．いったん確立されたパスをそのまま使用するのは，匿名性に対する攻撃から守るためである（文献8）を参照）．静的なパスを維持するために，各jondoはパス上の受信元と送信先を記録しておく．あるjondoが同じ受信元（同じパスIDを持つjondo）から別のリクエストを受け取ると，前回と同じ送信先へ転送する．パスが変更されるのは，パス上のjondoで障害が発生した場合と，新しいjondoがcrowdに参加した場合である．新規参加者があった場合，従来のパスはすべて破棄し，最初から作り直す．従来のパスをそのまま維持すると，新たに参加したjondoにより構築されたパスだけが目立ってしまう（この場合，リクエストの発信者がほかのjondoに知られてしまう）．

各jondo間の通信は，それらのjondoで共用される暗号キーで暗号化される．これにより一定水準の匿名プロパティを実現できる．Crowdsプロトコルの詳細については文献8）を参照．

## 匿名プロパティ

匿名通信プロパティは，少なくとも3つの座標軸で特徴

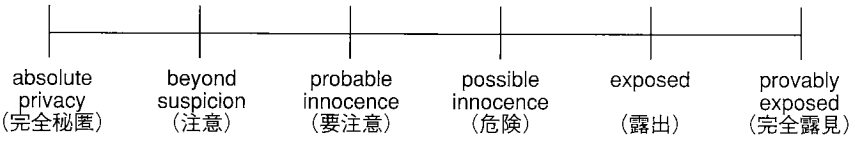


図-2　プライバシーの6段階

| Attacker | Sender anonymity | Receiver anonymity |
|---|---|---|
| local eavesdropper | exposed | $P$ (beyond suspicion) $\underset{n\to\infty}{\to} 1$ |
| $c$ collaborating members, $n > \frac{pf}{pf-1/2}(c+1)$ | probable innocence<br>$P$ (absolute privacy) $\underset{n\to\infty}{\to} 1$ | $P$ (absolute privacy) $\underset{n\to\infty}{\to} 1$ |
| end serves | beyond suspicion | N/A |

表-1　Crowdsの匿名プロパティ

付けることができる．第1は匿名性のタイプである．ここで，送信者の匿名性と受信者の匿名性について考えてみたい．送信者はWebユーザで，受信者はWebサーバである．送信者の匿名性は，メッセージ（Webリクエスト）を送信する側の身元を隠すことであり，受信者（とメッセージそのもの）もできれば隠すことを意味する．受信者の匿名性は，受信者（Webサーバ）の身元を隠すことである．

第2は，匿名性のタイプによってどのような敵が存在するかという観点である．ここでは，Webサーバ，ほかのcrowdメンバ，ローカル盗聴者（ユーザのローカルゲートウェイ管理者など）であり，いずれもユーザのマシンが参加した通信の全部（およびそれだけ）を監視できる者である．

第3は匿名性の程度である．匿名性の各タイプの強度をスペクトルの形で**図-2**に示す．スペクトルの左端は「完全秘匿（absolute privacy）」☆2で，これは通信の存在すらも敵に気づかせない最強の匿名性である．右端は「完全露見（provably exposed）」で，送信者や受信者の身元，両者の関係を敵が他人に立証できるほど知られてしまうレベルである．スペクトル中間に「注意（beyond suspicion）」と「要注意（probable innocence）」がある．「注意（beyond suspicion）」は，メッセージが送信された事実が敵に知られてしまう可能性はあるが，敵の目には，個々の送信者がそれぞれ等しい確率で発信者であるように見えるレベルである．これより強度が低い「要注意（probable innocence）」は，敵の目には，特定の送信者が発信者であるように見えるが，少なくともその送信者は原発信者ではなく，敵の推測は外れる確率の方が高いレベルである．

Crowdsが実現する匿名プロパティを**表-1**に示す．第1に，リクエストの原発信者であるjondoは，crowdの中から無作為に抽出したメンバにリクエストを転送するので，エンドサーバがどのメンバからリクエストを受け取るかはほぼ等しい確率になる．したがって，エンドサーバはリクエストの発信者を知ることはできない．ただし，リクエストがcrowdのメンバから発信された事実に気づく可能性はある．敵がエンドサーバの場合，これは図-2の「注意（beyond suspicion）」に相当する．

第2に，どのjondoにとっても，受信元のjondoがリクエストの原発信者であるのか，単に中継しただけなのかは

☆2 プライバシーの6段階について原語の表記と訳語が対応しないと不審を抱かれた読者もいるだろう．原著者のコメントに基づいて少し補足しておく．

**Absolute privacy**：「絶対プライバシー」が直訳．「プライバシーは絶対に露見しない」という意味．通信している事実そのものも敵に知られない．プライバシーの観点でいうと個人の身元は完全に守られる．訳語として「完全秘匿」とした．

**Beyond suspicion**：「疑いなく」が直訳．このままではピンとこないだろう．あるグループが通信している事実は敵（盗聴者）に知られるが，特定の個人は分からない．敵を探偵に見立て，犯人（原発信者）を探すといった考え方のようだ．「プライバシーはほぼ安全だが，（盗聴に）注意」という気持ちで「注意」と訳した．

**Probable innocence**：「確実に無罪」が直訳．敵を探偵に見たてると，犯人（原発信者）を検挙して裁判にかけても，確実に無罪（innocence）になる．敵の推測が外れる確率が多いが3割ぐらいは当たる確率．事実上推測は困難．個人のプライバシーの観点で言えば「そろそろ注意したほうがいいですよ」といった気持ちで「要注意」と訳した．

**Possible innocence**：「多分無罪」が直訳．敵の推測は五分五分．個人のプライバシーの観点で言えば「真剣に盗聴対策を考えてください」といった気持ちで「危険」と訳した．

**Exposed**：「露出」が直訳．犯人が分かる．個人のプライバシーの観点で言えば「見られていますよ」といった気持ちで「露見」と訳した．

**Provably exposed**：「確実に露出」が直訳．第3者が他人に犯人の身元を証明できる．個人のプライバシーの観点で言えば「丸見えですよ」といった気持ちで「完全露見」と訳した．

分からない．したがって，パス上にあるどのjondoもリクエストの原発信者を知ることはできない．もう少し厳密に述べてみよう．まず，crowdの中でリクエストの原発信者がだれであるかを共同で推測しているメンバ数が$c$だとする．当然，この$c$の中には原発信者はいない．パスで最初にリクエストを受け取った共同者にはその相手が原発信者であるように見える．この可能性が高いのは，この共同者には少なくとも原発信者がそのパスにいることが分かるからだ．しかし，文献8）で明らかなように，パスを構築した時点でcrowdのメンバ数が$n$個だとした場合，原発信者からリクエストを直接受け取る共同者がだれであるかの確率はせいぜい1/2である．ここで，$c$，$n$，$pf$の相互関係を表-1に示す．たとえば，リクエストを転送する確率が$pf$=3/4で，crowdメンバ数が少なくとも3($c$+1)であるとした場合，共同者の目には，少なくともその半分が原発信者でないように見える．図-2に関連させていうと，どのcrowdメンバも原発信者でないように見える（送信者の匿名性は「要注意」）．さらに，$c$個の共同者のだれも特定のパスに存在しない（したがって，そのパス上の通信をだれも観測できない）確率は，crowdのサイズの拡大に伴って増大する．したがって，$n$が増大すればするほど，$c$個の共同crowdメンバに対する送信者と受信者の匿名性の絶対プライバシーの確率は1に近づく．

第3に，crowdを使用すると，盗聴者から受信者の匿名性を保護できる．盗聴者の例としては，ユーザのマシンに関連したすべて（および特定の）通信を観測できるゲートウェイ管理者が挙げられる．Crowdsがjondo間のすべての通信を暗号化するので，発信者が自分自身でリクエストを宛先に直接送信しなかった場合，盗聴者がリクエストの最終送信先を知ることはできない．crowdのサイズが拡大すればするほど，原発信者が自分で直接送信する確率が低下するので，受信者（ここではWebサーバ）が盗聴者に見つかる確率も低下する．言い換えると，$n$の値が増加すればするほど，盗聴者の目には受信者の匿名性(容疑）は1に近づく．

## リスクと制限

Crowdsの利用者にとって知っておくべきリスクと制限事項がいくつかある．第1は，crowd内から発信されたリクエストを送信するのはcrowdメンバのだれかであるので，Webサーバのログには，送信したjondoのIPアドレスが発信者のアドレスとして記録される可能性があることだ．もちろん，このリクエストが盗聴された場合，送信したjondoの所有者にとって，そのjondoがCrowdsを実行したことは事実だが，リクエストの発信元ではないことを説明することは簡単だ．実際，Crowdsはこの説明どおりに設計されている．

第2に，Crowdsはリクエストの発信者の匿名性は保護するが，リクエストの転送に使用されるパス上のjondoがリクエストの内容を見ることは阻止できない．したがって，Crowdsを使用しないより使用した方が，リクエストの内容が第三者（crowd内のほかのjondo）に漏れる確率が高くなる．これは，特定のWebサーバに対するユーザのアカウント名とパスワード，ユーザのクレジットカード番号など，発信者に固有の情報が漏れる可能性があるので重大だ．しかし，このような通信はユーザの身元を何らかの方法でエンドサーバに知らせるのが目的なので，Crowdsのような匿名化ツールをこの種の通信に適用するのは逆効果になる．この問題の対策として我々は，ブラウザでプロキシの設定を無効にすることを勧める．こうすると，ブラウザとWebサーバとの間で直接通信が可能になるからだ．そのほかの対策として，SSLを利用してブラウザとWebサーバ間の直接暗号化通信を実現するようにCrowdsを改良することも考えられる（文献6）を参照）．ただし，現在のインプリメンテーションはこの機能をサポートしていない．

第3に，プロキシベースの匿名化サービスと同様，Crowdsにも抜け道がある．ユーザのブラウザがモバイルコード（JavaアプレットやActiveXコントロールなど）をダウンロードし実行すると，そのモバイルコードがネットワークコネクションをオープンしてサーバに接続し直す可能性がある．このコネクションは，通常，ユーザのjondoを介さずにWebサーバに直接なされるので，ユーザの身元がWebサーバに知られてしまう．この対策として，Crowdsを利用する場合，ブラウザの設定でJavaアプレットやActiveXコントロール（少なくともそのネットワーク機能）を無効にすることを勧める．

第4に，Crowdsを利用すると，ユーザの目にも明らかなほどWebページの検索回数が増加する．一般的に，jondoを実行するマシンのネットワークトラフィックと負荷も増加する（応答時間の実験結果については文献8）を参照）．この実験結果から，Crowdsに固有のオーバヘッドはきわめて小さく，むしろ，検索ページに埋め込まれている画像の数の影響が非常に大きいことが分かった．さらに分析の結果，crowdのサイズを拡大しても各jondoの負荷はほぼ一定であるので，Crowdsは大規模ネットワークのサポートが可能である．最近これらの実験結果についてCrowdsユーザに対して非公式の調査を実施したところ，ほとんどのユーザにとってCrowdsの性能はほぼ満足できることが分かった．現在は一部インタプリタ型で遅いPerl 5で実装しているが，その代わりにCなどのコンパイル型言語でCrowdsを実装すれば，応答時間をもっと短縮できるであろう．また，crowdメンバ同士のネットワーク近接性が向上するようにcrowdを編成すれば，jondo間の通信遅延を減少させることが可能になるだろう．

## 他のシステムとの比較

Web通信を匿名化するツールとしてアノニマイザ

(Anonymizer) ☆3がある．これはWebリクエストに対する簡易プロキシとして機能するWebサイトである．ユーザがURLを求めるリクエストを発信すると，そのリクエストはいったんアノニマイザに転送され，アノニマイザがそのURLをエンドサーバに送信する．アノニマイザは，サーバからレスポンスを受け取り，ユーザのブラウザへ転送する．アノニマイザはエンドサーバからユーザの匿名性を保護する役割を果たす．そのため，エンドサーバはリクエストの発信者を知ることができない．このような仕組みを実現した例としてLPWA（Lucent Personalized Web Assistant）がある．このプロキシは，エンドサーバにユーザの身元を知らせずに特定個人向けのWebブラウジング（アカウント名とパスワードを要求するWebサイトにアクセスすること）を明示的にサポートする（文献5）を参照）．

アノニマイザやLPWAよりCrowdsが優れているのは，受動的な盗聴者によってユーザの匿名性が破られることがまったくないことである．つまり，アノニマイザやLPWAのシステム管理者は，ユーザのブラウジング行為を知ることが可能なので，システム管理者が知り得た情報を他者に漏らしたり悪用したりしないことを信じるしかない．ところがCrowdsを使用すれば，ユーザのブラウジング行為はだれも知ることができない．1人のユーザを監視するだけでも，盗聴者はすべてのjondoのすべての通信を監視するか，自分のローカルマシンでユーザの行為を直接監視しなければならない（1つのcrowdが複数の管理ドメインにまたがる場合，すべてのjondoのすべての通信を監視するのは困難である）．

> Crowdsは，ユーザの識別が潜在的な可能な情報をWebサーバに知られないようにする．ユーザのIPアドレスもドメイン名も知らせない

匿名Web通信をサポートするミックス（mixes）[1]ベースのシステムには，一般的なインターネット通信（文献9）を参照）のほかにもいくつかある．ミックスネットワーク（mix network）は専用ルータ（コールミックス）の集まりである．各送信者が通信を中継しながらいくつかのミックスを経由して宛先に届ける．ルーティングプロトコルは階層化暗号技術を使用する．そのほか，各ミックスでメッセージをバッファに格納したり順序を入れ替えたりすることで，ミックスに入力されたメッセージと外部の盗聴者の目に触れる出力メッセージとの相関関係を隠す．Crowdsとミックスとの詳細な比較については，紙幅の関係で簡単に要点だけを述べておきたい．ミックスネットワークが提供する匿名プロパティはCrowdsと異なっており，プロトコルもより重くなっている．結論だけを述べると，Crowdsが提供する匿名プロパティとその匿名プロパティを実現するプロトコルの方がミックスネットワークより同期通信（Webトランザクションなど）に適合しているといえる．

## 配置問題

ユーザの請求に基づいて配布したCrowdsコードは，現時点で約1,400コピーになる．現在我々は，インターネット上で実行するアクティブなCrowdsの保守に取り組んでいる．Crowdsを開発し配布した経験は，高度な分散ソフトウェアシステムをインターネットにどのように配置するのかという実際的な諸課題を解決するためのヒントを与えてくれるという意味で，非常に役立っている．あらかじめ予期していた問題もあったし，まったく予想外の問題もあった．

インターネットのネットワーク接続形態に起因する問題がいくつかある．第1はファイアウォールである．ファイアウォールは，1つのcrowdが複数の管理ドメインにまたがることを制限する．すべてのネットワークサーバと同様，jondoもIPアドレスとポート番号で識別され，特定の標準ポート以外のポートに着信するコネクションを許可しない企業ファイアウォールが多い．ファイアウォールは，通常，ファイアウォールの外側にあるjondoからファイアウォールの内側にあるjondoへのコネクションを許可しない．外部に向かうコネクションに対してはどのポートからでも許可するファイアウォールが多いので，あるjondoがファイアウォールの外側に向かうパスを開始してその延長線でWebサーバに接続する可能性は否定できない．事実，ファイアウォールの内側にあるCrowdsユーザが自分のjondoをこのように設定することはあり得る．しかし，ファイアウォールは，ドメイン外のパス上にある最初のjondoに開始元のコンピュータがそのドメイン内にあるかどうかを検証する手段を提供する．このjondoは，パスを戻る方向に位置するjondoへコネクションをオープンする．この試みが失敗した場合，そのパスがそのドメインで開始されたことを示すことになる．したがって，ファイアウォールの内側にあるcrowdメンバには，ファイアウォールの外側にあるメンバと同じ匿名性は得られない．ファイアウォールが課すこのような制限を回避するには，たとえば，SSLと標準SSLポートを使用してjondoを通信させればよい．このような通信を許可するファイアウォールもある．しかし，SSLを使用すると，プロトコルのコストが高くなるし，SSLポートを使用すると同一マシン内におけるほかの用途の間で競合が起きる可能性がある．さらに，ファイアウォールの外部にあるjondoからファイアウォールの内部にあるjondoへのコネクションを許可すると，せっかくファイアウォールで保護したリソース（特にファイア

☆3 http://www.anonymizer.com

本文で紹介したCrowdsのコードは，米国とカナダ内には無料で配布している．http://www.research.att.com/projects/crowds

ウォールの内側に設置したWebサーバ）を外部に曝すリスクを負うことになる．

第2の問題はモデム接続である．モデム経由でインターネットへ接続する場合，Crowdsの潜在的なユーザの大部分は比較的狭い帯域幅と大きな遅延を我慢しなければならない．crowdのメンバは，ユーザが接続するたびに，自分のWebリクエストだけではなく，自分のコンピュータがメンバとして参加しているパスに送り出されるリクエストにも遅延が生じる．そのため現在のcrowdは，インターネットへ直接接続するcrowdメンバだけを対象にしている．モデム経由でしか接続できないユーザは別個の遅いcrowdを使用するしかない．もちろん，他人のマシンでjondoを使用してインターネットへ高速接続することも可能だが，そのような場合は匿名プロパティによる保護が得られないので，ユーザのブラウザとjondoとの間の通信は盗聴される可能性がある．

そのほかに，jondoの一時性という問題がある．Crowdsユーザが使用するjondoがcrowd内に留まる時間（ユーザのブラウジング時間など）は非常に短く一時的であることがよく知られている．Crowdsのα版では，jondoがよくクラッシュしたものだが，その原因の1つは一時性にある．その後の新バージョンでjondoの信頼性が向上しても，クラッシュ問題は残った．この一時性という問題は，jondoの一時的な所有者だけではなく，crowdの匿名化機能全体に対しても悪影響を与える．jondoがcrowd内に留まる時間を長くするには，その利点をユーザに分かってもらうようにするしかなさそうだ．そこで我々は，crowd内jondoの滞在時間をもっと長くすることを勧める啓蒙活動を開始した．crowdに参加するjondoの頻度を制限することで，この目的を実現することも可能だ．この対策はほかの意味でも役立つ（文献8）を参照）．

最後に，米国の輸出規制の問題を指摘しておきたい．強力な暗号技術を採用したソフトウェアは輸出規制の対象になるが，Crowdsも対象になる（文献3）を参照）．そのため，Crowdsの利用は，現在，米国とカナダだけに制限されている．

## 政治

より広く社会的政治的な観点で眺めると，インターネット上の匿名性は暗号カギの預託と並んで，電子プライバシーをめぐる議論の中心的なテーマになっている（文献2）と3）を参照）．これらの議論を踏まえれば，Crowdsが必ずしも歓迎されていないことは驚くに当たらない．

たとえば，Crowdsの普及に懸念を表明したインターネットベンダもある．インターネットで利用できる匿名化サービスとしてはアノニマイザプロキシしかなかった頃，アノニマイザ経由で提出されたクレジットカード番号の多数が盗まれたものであることに気がついたインターネットベンダがいる．そのため，アノニマイザ経由の購入注文を拒否するようにWebサーバを設定した．我々がCrowdsをリリースする意向を表明したところ，これらのベンダから反対意見が出された．その理由として，どの購入注文がcrowd経由であるのかを検証するのが困難であること，さらに，クライアントのIPアドレスでリクエストをフィルタにかけたり不正なクレジットカードの使用者を突き止めたりすることは不可能だというのだ．このような理由で，crowdから来たリクエストを識別するための特別なヘッダフィールドを用意すべきだと要請された．これに対して我々は，仮にそのようなヘッダを用意しても，ユーザがコードを修正すれば簡単にヘッダを削除できるので，効果はないと述べた（Crowdsのソースコードは無料で配布している）．

そのほか我々が気がついた範囲でも，仕事でCrowdsの使用を禁止するポリシーを策定した大手企業が少なくとも1社はある．勤務中に「不適切な内容」を見ていた従業員がいたとしてもそれがだれであるかを特定できないことに，管理者が懸念を抱くのも当然だ．民間企業がそのような決まりを作ってはならないことはない．民間企業は，従業員の電子メールを閲覧したり電話の内容を盗聴する権利を有するからだ．

## まとめ

Crowdsは，ユーザが匿名でWebコンテンツを検索できるようにするためのツールである．Crowdsが提供する強力な匿名性は，Webトランザクションに適しており，プロキシベースのアノニマイザよりはるかに広い範囲の不正行為に対抗できる．本稿では，まずCrowdsの仕組みを簡単に紹介し，この技術には良い側面があると同時に制約事項もあることを我々の体験に基づいて説明した．この技術を採用する方々が増加すればするほどその有用性も高まる．crowdの規模が拡大すれば，匿名性が高まり，しかも性能にあまり影響が出ないからである．


参考文献

1) Chaun, D.: Untraceable Electronic Mail, Return Addresses and Digital Psedonyms, Commun. ACM 24, 2, pp.84-88 (Feb. 1981).
2) Denning, D. et al.: To Tap or Not to Tap, Commun. ACM 36, 3, pp.24-44 (Mar. 1993).
3) Diffie, W. and Landau, S.: Privacy on the Line: The Politics of Wiretapping and Encryption, MIT Press, Cambridge, Mass. (1988).
4) Droms, R.: Dynamic Host Configuration Protocol, RFC-1541 (Oct. 27 1993).
5) Gabber, E., Gibbons, P., Matias, Y. and Mayer, A.: How to Make Personalized Web Browsing Simple, Secure and Anonymous, In Proceeding of Financial Cryptography '97 (1997).
6) Garfinkel, S. and Spafford, G.: Web Security and Commerce, O'Reilly (1997).
7) Reiter, M.K., Anupam, V. and Mayer, A.: Detecting Hit-Shaving in Click-Through Payment Schemes, In Proceedings of the 3rd USENIX Workshop on Electronic Commerce, pp.155-166 (Aug. 1998).
8) Reiter, M.K. and Rubin, A.D.: Crowds: Anonymity for Web Transactions, ACM Trans. Info. Syst. Security 1, 1 (Apr. 1998).
9) Syverson, D., Goldschlag, M. and Reed, M.G.: Anonymous Connections and Onion Routing, In Proceedings of the 1997 IEEE Symposium on Security and Privacy (May 1997).


（平成11年5月6日受付）