

The seal of the University of Bologna is a large, circular emblem in the background. It features a central figure, likely a saint or scholar, surrounded by various scenes and text. The outer ring contains the Latin motto "UNIVERSITAS BOLOGNENSIS" and "S. PETRI APOSTOLI". The inner ring contains "COLLEGIUM BOLOGNENSIS" and "S. PETRI APOSTOLI". The central figure is seated and holding a book. The seal is rendered in a light, golden-brown color.

# **User untraceability in the next-generation Internet: a proposal**

**Mauro Tortonesi**

**Renzo Davoli**

**Technical Report UBLCS-2002-8**

August 2002

Department of Computer Science  
University of Bologna  
Mura Anteo Zamboni 7  
40127 Bologna (Italy)

The University of Bologna Department of Computer Science Research Technical Reports are available in gzipped PostScript format via anonymous FTP from the area `ftp.cs.unibo.it:/pub/TR/UBLCS` or via WWW at URL `http://www.cs.unibo.it/`. Plain-text abstracts organized by year are available in the directory ABSTRACTS. All local authors can be reached via e-mail at the address `last-name@cs.unibo.it`. Questions and comments should be addressed to `tr-admin@cs.unibo.it`.

## Recent Titles from the UBLCS Technical Report Series

- 2000-15 *An Adaptive Mechanism for Securing Real-time Speech Transmission over the Internet*, Aldini, A., Gorrieri, R., Rocchetti, M., November 2000.
- 2000-16 *Enhancing Jini with Group Communication*, Montresor, A., Babaoglu, O., Davoli, R., December 2000 (Revised January 2001).
- 2000-17 *Online Reconfiguration in Replicated Databases Based on Group Communication*, Kemme, B., Bartoli, A., Babaoglu, O., December 2000 (Revised March 2001).
- 2001-1 *Design and Analysis of Protocols and Resources Allocation Mechanisms for Real-Time Applications* (Ph.D. Thesis), Furini, M., January 2001.
- 2001-2 *Formalization, Analysis and Prototyping of Mobile Code Systems* (Ph.D. Thesis), Mascolo, C., January 2001.
- 2001-3 *Nature-Inspired Search Techniques for Combinatorial Optimization Problems* (Ph.D. Thesis), Rossi, C., January 2001.
- 2001-4 *Desktop 3d Interfaces for Internet Users: Efficiency and Usability Issues* (Ph.D. Thesis), Pittarello, F., January 2001.
- 2001-5 *An Expert System for the Evaluation of EDSS in Multiple Sclerosis*, Gaspari, M., Roveda, G., Scandellari, C., Stecchi, S., February 2001.
- 2001-6 *Probabilistic Information Flow in a Process Algebra*, Aldini, A., April 2001 (Revised September 2001).
- 2001-7 *Architecting Software Systems with Process Algebras*, Bernardo, M., Ciancarini, P., Donatiello, L., July 2001.
- 2001-8 *Non-determinism in Probabilistic Timed Systems with General Distributions*, Aldini, A., Bravetti, M., July 2001.
- 2001-9 *Anthill: A Framework for the Development of Agent-Based Peer-to-Peer Systems*, Babaoglu, O., Meling, H., Montresor, A., November 2001.
- 2002-1 *A Timed Join Calculus*, Bünzli, D. C., Laneve, C., February 2002.
- 2002-2 *A Process Algebraic Approach for the Analysis of Probabilistic Non-interference*, Aldini, A., Bravetti, M., Gorrieri, R., March 2002.
- 2002-3 *Quality of Service and Resources' Optimization in Wireless Networks with Mobile Hosts* (Ph.D. Thesis), Bononi, L., March 2002.
- 2002-4 *Specification and Analysis of Stochastic Real-Time Systems* (Ph.D. Thesis), Bravetti, M., March 2002.
- 2002-5 *QoS-Adaptive Middleware Services* (Ph.D. Thesis), Ghini, V., March 2002.
- 2002-6 *Towards a Semantic Web for Formal Mathematics* (Ph.D. Thesis), Schena, I., March 2002.
- 2002-7 *Revisiting Interactive Markov*, Bravetti, M., June 2002.

# User untraceability in the next-generation Internet: a proposal

Mauro Tortonesi<sup>1</sup>

Renzo Davoli<sup>2</sup>

Technical Report UBLCS-2002-8

August 2002

## Abstract

*Internet is changing. The adoption of IPv6 is only one change. The number of mobile hosts connected to the network is increasing also due to the wide application of the wireless technology. In this scenario there are also new threats to user privacy.*

*This paper analyzes how to protect the user privacy in a global view from the data link level up to the applications. In fact, several information about user habits in terms of identity, physical location and movements as well as corresponding hosts and visited sites can be inferred by eavesdropping the data exchanged on local networks at the data link layer. We describe a method to compute dynamically changing unique hardware addresses of network interfaces for the anonymization of data-link traffic and how to integrate this protection with those available at the upper layers.*

---

1. Dept. of Engineering, University of Ferrara, Italy. [mauro@ferrara.linux.it](mailto:mauro@ferrara.linux.it)

2. Dept. of Computer Science, University of Bologna, Italy. [renzo@cs.unibo.it](mailto:renzo@cs.unibo.it)

## 1 Introduction

Security and privacy have always been two sides of the same problem. A lot of work has been done so far to enhance the authentication of communicating peers [KA98a] [MC02] and to preserve the integrity and the privacy of data exchanged between them [KA98a] [DA99, HK95, FKK96] [CD01]. However, the effort for preserving the privacy about the location and the identity of a mobile host is a quite novel research issue.

The result of a projection of the Internet (and its underlying IP technology) growth shows that mobile computing will be one of the main issue tomorrow. The World Wireless Research Forum in its “Book of Visions” [WWR01] foresees that in the future wireless units (portable *phones* with capabilities that go far beyond those of third generation/UMTS devices) will be IPv6-only devices capable of using software-radio technology to access all kinds of possible wireless communicating infrastructures (GSM, UMTS, LPD, 802.11, bluetooth to cite some in use nowadays).

In this scenario privacy must be intended also as untraceability, i.e. the right for users to keep their identity and location private. In short, while in the past the main focus has been put only on the privacy of data, today we must also consider the privacy of users.

Users move carrying their portable devices and navigating through different and heterogeneous wireless infrastructures: public telecommunication networks as well as hot spots [HS] or service networks for info in public places. The users’ devices communicate through several networks under different administrative domains, with in general no guarantees of fairness and correctness about the management of data. Providers of connectivity could use any transiting packet to collect information about user habits, interests, standard move schedules.

To preserve users’ privacy at least the following information must be kept secret:

- identity. Information about the identity of the user (in terms of her name) are not directly included in the communication protocols but can be inferred by analyzing the pattern of traffic generated by the user. In fact, a host (which is owned in the common case by a single person) can be identified by its hostname or even by the unique hardware address of its NIC, thus revealing these information is in some sense similar to revealing the user identity. There are also many other sources of data that can provide significant information about the identity of the user (e.g. the address of her home agent or the prefix of her home network). In general, it is possible to collect data about habits of users by tracking their machines.
- location. Clearly it should not be possible to track where the user is currently.
- data and pattern of traffic. Not only the privacy of data must be preserved but also the pattern of traffic (i.e. visited sites, endpoint of connections, etc...), as it may carry important information that could be used to identify the user. A common academic example is the spying attack to a stock exchange agent: in this case the timing of a stock order is a priceful information itself, regardless to the content of the message. Recent studies have shown that also encrypted web traffic can carry interesting information [SSW<sup>+</sup>02].

The rest of the paper is organized as follows: in section 2 we analyze in details the untraceability problem; then we present a new technique to achieve untraceability at the data-link level in section 3; we describe limitations and open problems of the proposed technique in section 3.1 and finally we show how to extend this data-link layer solution to a global approach to the privacy problem in section 4. A comparison with related works in section 5 and the section of conclusions and future works complete the paper.

## 2 IPv6 untraceability weakness

At the moment of writing, the problem of untraceability in the new IPv6 protocol remains unresolved.

In fact, IPv6 stateful address autoconfiguration [DBV<sup>+</sup>02] does not guarantee full anonymity and untraceability for a host, as its network layer address must be negotiated with a potentially untrustful entity which maintains a table of associations between network layer and data-link layer addresses of all statefully-configured hosts. Besides, the hosts keep using their unique hardware addresses as data-link layer addresses and this makes them traceable by on-link attackers and off-link attackers who have access to the association table data.

With IPv6 stateless address autoconfiguration [TN98] the situation is even worse. The hardware address of the user's device is directly included in the interface identifier portion (the last 64 bits) of public unicast addresses, so significant information about the user's identity can be collected by simply eavesdropping the traffic in any position inside the links of the two communication endpoint or along the communication path.

Moreover, this behaviour allows a potentially dangerous correlation of the data collected at the hand-off that follows a movement of a mobile host from a (typically wireless) link to another. In fact, the interface identifier portion of the initial addresses (the address before the movement) and that of the final address (the address after the movement) of the host are coincident. So, it is possible for probes (programs that put interfaces in promiscuous mode or read-only state) to collect data about user habits, standard transit times on a network or standard paths of the user in terms of visited network cells. (See [Air] as an example of method used to spy wireless networks).

To solve these traceability problems, IETF has published in early 2001 an interesting proposal [ND01] that adds privacy extensions to the existing stateless address autoconfiguration mechanism. Unfortunately, this proposal works at the network level, and this leaves the host traceable by on-link attackers. Moreover, it has been showed that this proposal does not provide full untraceability to hosts [Pas01, Pas02], and that it does not provide an acceptable degree of privacy if the network prefix is univocally assigned according to the hardware address of the device (e.g. the dialup ISP scenario) [DS02].

From the previous analysis, it is clear that the anonymity and untraceability problem must be handled at the data-link layer. In fact, even if there were a mechanism providing perfect network layer anonymity and untraceability, it would always be possible to reconstruct the user's identity (and thence, her location) not only by analyzing patterns of the generated traffic (e.g. accessed sites, home addresses, home network prefix, etc), but also from its unique hardware address. In fact, a wireless interface, as well as a IRLan infrared node, sends its hardware address along with each packet, and it is possible to force the transmission of a packet (e.g. by a ping to a broadcast address). It is certainly true that hardware addresses are visible only to directly connected hosts (thus hosts are prone to local attacker only), but nowadays the users get connected (sometimes in an unaware manner) to several untrusted network, so this is a problem to take into consideration.

### 3 A proposal for untraceable hardware addresses

As already stated, to achieve a reasonable degree of untraceability, we need to develop a mechanism to create unique, random and dynamically-changing hardware addresses.

In fact, to be untraceable an hardware address must have the following characteristics:

- it must be unique in a scope that is at least wide as the link scope if the network layer address is obtained by stateful address autoconfiguration [DBV<sup>+</sup>02] or if other techniques are used to provide untraceability of the network layer address [ND01], and at least wide as the scope of the widest-scoped network prefix published by the routers which are on the same link of the host if network layer addresses are obtained by stateless address autoconfiguration [TN98]
- it must depend from the prefix of the network where the host is and it must change when the host moves from a network to another (in order to achieve spatial untraceability)

- it must change in time (in order to achieve temporal untraceability and to prevent attackers from being able to perform a thorough study of the user's habits, which could eventually bring them to the identification of the user)
- when the address is changed, the new address must be hard to calculate for external observers while computationally easy to calculate for the host.

Besides, if we want to implement a real-world solution which is based on these considerations, then our hardware address must satisfy the following conditions too:

- its format must be compatible with the one of the hardware addresses already in use, not to require the current implementations of upper and lower layer protocols to be modified;
- nodes that make use of generated hardware addresses should be able to communicate with all those nodes that do not support the privacy extensions at the data-link layer presented in this paper;
- it should be possible, with the authorization given by the user, to go back to the hardware address used in a specific period of time (legal traceability).

There are two possible ways to obtain a procedure that generates unique addresses: by developing a technique that guarantees the *a priori* unicity of generated addresses or simply by generating random addresses and detecting conflicts with the other hardware addresses used on the same link.

Duplicated address detection at the data-link layer (which must not be confused with the duplicated address detection procedure specified in [TN98] or with the new duplicated interface identifier detection procedure being developed by IETF - in fact both of these work at the network layer) is certainly not a common practice nowadays, at least in the most used networking technologies for local area networks, and we are quite confident that there would be at least a few problems to solve in order to develop a robust conflict detection procedure that works in a wide range of cases. Furthermore conflict detection should be avoided on mobile hosts as it is a time and energy consuming task: it requires energy as it requires more in terms of transmission time and also time as the conflict detection protocol must terminate before sending data on the channel. Energy is a critical issue on battery operated units and handoff time must be minimized for the mobility to be effective.

Thus we will focus our attention on methods that provide *a priori* unicity for the generated addresses.

Different proposals can be made:

- assigning a large-enough range of hardware addresses to every interface and using the network prefix as a parameter for the computation of the generated address.

The generated hardware address would then be:  $Crypt(NP, HW_n)$  where NP is the network prefix and  $HW_n$  is one of the hardware addresses of the interface, chosen at random in the range of addresses assigned to the interface. The lifetime of each generated address should be chosen randomly between a given minimum and maximum value, which could be administratively configured.

This solution is easy to implement and it satisfies nearly all of the conditions above, but it presents a problem: spatial untraceability requires the interfaces to be assigned a wide range of addresses (otherwise in a few changes the interface would be assigned a generated address already used recently) and this would limit the maximum number of interfaces that can be allocated in a limited address space (e.g. the 48-bits address space of EUI48).

- assuming the existence of a synchronized clock, using both the network prefix and the current time as parameters for the computation of the generated address. This is a somewhat

reasonable assumption, as the WWRF's "Book of Visions" [WWR01] forecasts that next-generation mobile devices will have GPS capabilities.

Moreover, there is no need of having strong synchronization and we are quite confident that a well-configured NTP client should provide a level of synchronization which will be more than enough for the normal use.

In fact, if  $\delta$  is the maximum bias between the local time of two interfaces, it is possible to guarantee the unicity of hardware addresses by setting a minimum address lifetime of  $2\delta$ , losing only one bit in the unique hardware address space of the interfaces.

Actually, this scheme makes use of unique hardware addresses that are 1 bit shorter than usual (e.g. 47 bit for EUI48 interfaces). Given  $N$  as the number of bits reserved for the hardware address we reserve the first  $N - 1$  for the hardware address while the last one is kept to avoid collisions due to clock de-synchronizations.

Let  $T > 2\delta$  be the lifetime of a generated address, which is equal for all the interfaces of a single data-link defined network, then the generated address will be:  $Crypt(NP, ((HW + t) \bmod(2^{N-1})) * 2 + odd(t))$  where NP is the network prefix, N is the length in bits of the hardware address, HW is the hardware address of the interface (which is unique in an address space of N-1 bits), t is the interface time in units of T size, and odd(t) is the oddity value of t (that is  $odd(t) = t \bmod(1)$ ).

The generated address is always unique, as in every network two nodes with different hardware addresses will produce different generated address.

*Crypt* must be an encryption function: mathematically for each given key it is a permutation of the set of hardware addresses. The inverse function must be computationally hard to compute. With 64 bits hardware addresses DES could be effectively applied; other algorithms can be applied to encode 48 bit. Even if a 48 bit message can be decoded in some weeks by brute force, it is ineffective for computing all the real hardware addresses needed to collect data on a number of users. Clearly the use of 64 bit hardware addresses would give a better degree of privacy but it is not compatible with the majority of network interfaces today (IEEE802 compliant).

With the authorization of the user and the knowledge of the address HW, it is possible to find out which generated hardware address has been used in a specific period of time (with an accuracy of  $2\delta$ ). So, the legal traceability requirements specified above are fully satisfied.

Notice that during the address update process there can be a de-synchronization between the interfaces that can bring some of them to have a wrong value of t. Anyway, as the de-synchronization of the interfaces never lasts more than  $2\delta$  and t never differs of more than one unit from its correct value, the proposed algorithm is still successful in providing a unique generated address. In fact, if two nodes are de-synchronized (that is, if they have different values of t), the hardware addresses of node 1 may produce a value of  $(HW1 + t1) \bmod(2^{N-1})$  which is equal to the value  $(HW2 + t2) \bmod(2^{N-1})$  produced by node 2, but the generated address of the two nodes will be different as  $odd(t1) \neq odd(t2)$ .

Notice that this procedure does not present the same counterindications of the previous one.

This generated hardware address can be used to obtain a unique and untraceable interface identifier, according to the rules specified in [HD98].

Thus, the network layer address can be configured by stateless address autoconfiguration [TN98], and there is no need to resort to any kind of privacy extensions [ND01] or to stateful address autoconfiguration [DBV<sup>+</sup>02], as the privacy and the untraceability of the network layer address is already guaranteed by the use of the generated hardware address.

The security of this procedure can be incremented by introducing a random delay in the update of the address (and by incrementing the minimum lifetime T of the generated addresses as

well), to make sure that the synchronization of the clock cannot be used to recognize an invariant pattern in the traffic.

### 3.1 Known problems

The proposal introduced above presents a few minor problems.

First of all, there is a non-zero probability that the generated address of a host will conflict with a ordinary hardware address assigned to another host that does not make use of our procedures and is on the same link of the former one. Although we expect this situation to be very rare, the address generation protocol must handle this case too.

This problem should be addressed either

- by avoiding collisions by cutting off a specific range of addresses for untraceable devices,
- or by forcing everyone to use untreaceable address on a specific network (excluding the others)
- or by detecting collisions and changing duplicated addresses.

The former approach leads to a narrow address space thus it weakens the protocols for brute-force attacks. The latter should be avoided for efficiency. The second solution has an implementation problem: the traceable host should be ostracized by the network, so a method to exclude an interface from the network is needed.

There is also a minor annoyance with stateless address autoconfiguration [TN98]. Actually, as the network prefix is an indispensable parameter for the calculus of the generated address, it is imperative for a node implementing our hardware address untraceability protocol to retrieve this information before the actual generation of the address.

A possible solution could be putting the interface card in promiscuos mode, sending a few Router Solicitation ICMPv6 messages [NNS98] on the channel medium with a broadcast source address, and waiting the correspondent Router Advertisement response from the router. Notice that there are no problems if the network prefix is administratively configured on all hosts.

Besides, as nobody has studied so far the interactions between dynamic-changing hardware addresses and stateful address autoconfiguration, an extensive testing phase is needed in this case.

Finally, this method is critical for physical networks using several global scope prefixes. In fact, it is not possible to compute a physical address for each prefix as each interface must have a single physical address. If we allowed in theory the use of multiple hardware address per interface, this would have lead to the possibility of collision between addresses for different prefixes.

A proposed workaround is simply choosing the highest network prefix value among all those assigned to the physical network (published by routers or administratively configured). Notice that there is a possible attack to this method: by agreeing to publish the same fictitious global scope (but not routed) network prefix, malicious entities could force the encryption key to be the same on several networks (thus hosts moving from one of these networks to another one would have always the same generated address, regardless from the network they are visiting).

Another way to address this problem could be to use a key cryptographically obtained from the prefixes assigned to the network instead of a particular prefix in the computation of the generated address. We could use the same cryptographic method to generate the addresses and to compute the key. E.g:  $key = Crypt(prefix_1, Crypt(prefix_2, Crypt(...)))$  where  $prefix_1, prefix_2, \dots$  are the network prefixes, sorted according to a well-defined criterion. In this way, as it is statistically very rare to find two different networks with the same set of assigned prefixes, it is also computationally hard to find, among several networks, two set of network prefix that produce the same key.

It is our opinion, anyway, that in a secure and private network it is possible in many cases to introduce the constraint to have one single global scope network prefix.

## 4 The road to full untraceability

As already stated, untraceability of the hardware address is not enough to provide a real solution to the untraceability problem.

In fact, the privacy can also be violated by reading and analyzing the data exchanged through the network and the pattern of traffic.

For what concerns the privacy of the data, they can be encrypted using standard cryptographic techniques like IPsec Encapsulated Security Payload [KA98b]. Unfortunately, this approach is not enough to guarantee an acceptable level privacy. In fact IPsec encrypts only the payload portion of the packets data, while keeping the header portion (and thence the addresses of the connection endpoint) in cleartext.

Here the problem is a structural one, because, as the IP routing architecture is completely based on the use of routable addresses, the network prefix portion of the addresses contained in the header of each packet carries a significant information about the location of the communication endpoints and the identity of the communicating users

This is a structural characteristic of IP-based networks and cannot be subverted. Hence, an alternative (trusted) anonymization mechanism is needed, maybe with a distributed method. Each packet must be sent to the anonymizer with destination address and payload independently crypted.

The packets sent to the anonymizer should be encoded this way:

$$(Encrypt(PKA, destaddr), Encrypt(PKD, payload))$$

or

$$Encrypt(PKA, (destaddr, Encrypt(PKD, payload)))$$

where *Encrypt* is a public Key encryption function (e.g. RSA), *PKA* and *PKD* are the public keys of the anonymizer and the destination host, respectively. The anonymizer decrypts the destination address and sends the payload unchanged to its final destination.

In this schema the real destination is hidden in the path from the sender to the anonymizer, the real sender is hidden in the path from the anonymizer to the destination, and the anonymizer itself is able to see the traffic pattern but not the data transferred.

Anomizers are commonly used today in the Internet as commercial services (e.g. [Ano]). They are provided by private companies, many of them small and unknown, and they are used mainly to anonymize web navigations (thus unencrypted data). Users must trust completely the anonymizing service as data and destination are encrypted together. Needless to say, unfair companies could collect the history of all the navigation of their users.

It is possible to envision two different designs for the anonymization mechanism:

- **Trusted anonymizer:** provided by a well known company, maybe as a side-service by the telecom company which provides connectivity. The service (maybe implemented in a distributed manner) tends to give a good level of anonymity if the amount of exchanged data is quite high. In fact only the aggregate load pattern is externally visible.
- **Peer-to-Peer solution:** a big number of anonymizers are spread all over the network. Potentially all the privacy-enabled hosts can act as anonymizers for the others. In this way many anonymizers can be used at the same time (e.g. a different anonymizer can be randomly choosed in a large set for each transmitted packet). In such a way the information needed by an attacker to trace the network load is extremely distributed.

## 5 Related work

Privacy of users has been discussed in many research works. The privacy of the network layer address (IP level) has been recognized as a problem for mobile users in RFC3041 [ND01]. Then

the proposed solution has been criticized by [DS02] and [Pas01, Pas02]. All these methods protect the network address thus preserve privacy on all networks but the one where the mobile host is directly connected. In fact, the hardware address (e.g. the IEEE802 address) is fixed and unchangeable for each interface: if a spying host is on the same network all the protocols above are useless, the mobile host can be recognized by inspecting its hardware address.

Also the very recent internet draft by Montenegro and Castelluccia [MC02] addresses the problem of the anonymization at the network layer. The paper shares with our solution the idea to generate unique addresses by using cryptography techniques, but the method is applied to different protocol layers.

The web-surfing anonymity is another perspective of the privacy problem. This problem has been addressed by many authors: LPWA [GGK<sup>+</sup>99] is a centralized solution providing also temporary identities for a web-mail integrated service; Onion routing [STRL01], Crowds [RR98] and Hordes [SL00] are distributed solutions. Onion routing makes use of several level of encryption for the messages: each router makes cryptographic transformation on the messages and forwards them to the next hop making the traffic correlations hard to compute for an external observer. The anonymity of a single individual when it is in a crowd is the idea used by the Crowds project. The routing idea is similar to that we proposed above as the peer-to-peer approach to anonymity. Here the method is used at the network layer as a general solution instead of a service for web only. Hordes is an evolution of Crowds; it uses multicast to increase performances (and anonymity).

Another method to preserve privacy on web is P3P [RPMD<sup>+</sup>97]. Proposed by the W3C, P3P is a way to configure and manage the privacy on the web. P3P enables an agreement between web-sites and clients. Users can define their personal data and privacy policy by defining a *persona*. The policy is defined by a specific language named *appel*. Even if P3P has been criticized to facilitate web-sites to gather data instead of protecting personal information giving to users only an illusion of privacy [Coy99], methods à la P3P are not alternative to our method but instead complementary solutions: privacy at web level (or any other application layer protocol) is effective if it is impossible to read sensible data using the underlying layers and viceversa.

## 6 Conclusion and Future Work

This paper has introduced a unified view of the privacy problem, ranging from the physical layer up to the data-link and transport layer.

Obviously, untraceability of data-link layer addresses and network traffic alone cannot provide by itself a good level of privacy. A careful design and implementation of upper layer applicative protocols is needed.

As an example, an application layer protocol that reveals the real hardware address of a host makes useless all the efforts to keep it secret. Similar considerations can be applied to HTTP cookies [KM00]: it is useless to change the addresses on the net if it is possible for a web site to store a private information client-side and reread it later. The same problem applies to stored data accessible by downloaded code (e.g. Java or ActiveX).

For what concerns the procedure for the generation of untraceable hardware addresses presented in this paper, it is certainly an interesting technique but there are still many problems to solve: to cut out a reserved hardware address space to avoid collisions or to detect and handle incidental collisions between generated addresses and ordinary addresses, to retrieve experimental data about the cost of this method in terms of memory and energy requirements. The performance of the methods must be extensively tested as well.

There is also an orthogonal question: does the use of such a high degree of protection really always make sense? Nowadays the telecommunication companies can trace the position of all their customers in every moment, but this is a minor problem as those companies are considered to be trusted: they should not analyze our traffic and habits.

The method proposed in this paper could be applied only to untrusted links, e.g. spontaneous or public networks. The user can ask for an anonymous connection and, if the protocol is unavailable, she is aware that the public network management can violate her privacy, thence she has the right to choose whether to accept the risk or not. Maybe she can decide to use the untrusted connection for specific purposes, using a virtual machine or a sandbox not to reveal private information. Surely she is revealing her hardware address and this is a privacy violation by itself.

## References

- [Air] AirSnort Homepage. <http://airsnort.shmoo.com>.
- [Ano] The Anonymizer. <http://www.anonymizer.com>.
- [CD01] Claude Castelluccia and Francis Dupont. draft-castelluccia-mobileip-privacy-00.txt: A Simple Privacy Extension for Mobile IPv6. IETF, Internet-Draft, February 2001.
- [Coy99] K. Coyle. P3P: Pretty poor privacy? A social analysis of the Platform for Privacy Preferences, June 1999. Available at: <http://www.kcoyle.net/p3p.html>.
- [DA99] T. Dierks and C. Allen. RFC2246: The TLS Protocol. IETF, January 1999.
- [DBV<sup>+</sup>02] R. Droms, J. Bound, B. Volz, T. Lemon, C. Perkins, and M. Carney. draft-ietf-dhc-dhcpv6-26.txt: Dynamic Host Configuration Protocol for IPv6 (DHCPv6). IETF, Internet-Draft, June 2002.
- [DS02] Francis Dupont and Pekka Savola. draft-dupont-ipv6-rfc3041harmful-01.txt: RFC 3041 Considered Harmful. IETF, Internet-Draft, June 2002.
- [FKK96] A. Frier, P. Karlton, and P. Kocher. The SSL 3.0 Protocol. Technical report, Netscape Communications Corp., November 1996.
- [GGK<sup>+</sup>99] Eran Gabber, Philip B. Gibbons, David M. Kristol, Yossi Matias, and Alain Mayer. Consistent yet Anonymous Web Access with LPWA. *Communications of the ACM (CACM)*, 2(42), February 1999.
- [HD98] R. Hinden and S. Deering. RFC2373: IP Version 6 Addressing Architecture. IETF, July 1998.
- [HK95] Hickman and Kipp. The SSL protocol. Technical report, Netscape Communications Corp., February 1995.
- [HS] Web sites: [www.personaltelco.net](http://www.personaltelco.net), [www.nycwireless.net](http://www.nycwireless.net), [www.80211hotspots.com](http://www.80211hotspots.com).
- [KA98a] S. Kent and R. Atkinson. RFC2401: Security Architecture for the Internet Protocol. IETF, November 1998.
- [KA98b] S. Kent and R. Atkinson. RFC2406: IP Encapsulating Security Payload (ESP). IETF, November 1998.
- [KM00] D. Kristol and L. Montulli. RFC2965: HTTP State Management Mechanism. IETF, October 2000.
- [MC02] G. Montenegro and C. Castelluccia. draft-montenegro-sucv-03.txt: SUCV Identifiers and Addresses. IETF, Internet-Draft, July 2002.

- [ND01] T. Narten and R. Draves. RFC3041: Privacy Extensions for Stateless Address Auto-configuration in IPv6. IETF, January 2001.
- [NNS98] T. Narten, E. Nordmark, and W. Simpson. RFC2461: Neighbor Discovery for IP Version 6 (IPv6). IETF, December 1998.
- [Pas01] Alberto Escudero Pascual. Location Privacy in IPv6: "Tracking the binding updates". In *Proceedings of IMDS01*, Lancaster, UK, September 2001.
- [Pas02] Alberto Escudero Pascual. Privacy Extensions for Stateless Address Autoconfiguration in IPv6: "Requirements for Unobservability". In *Proceedings of RVK02*, 2002.
- [RPMD<sup>+</sup>97] Joseph Reagle, Martin Presler-Martin, Melissa Dunn, Philip DesAutels, Lorie Cranor, and Mark Ackerman. General Overview of the P3P Architecture. Technical report, W3C Working Draft, October 1997. Available at: <http://www.w3.org/TR/WD-P3P-arch>.
- [RR98] Michael K. Reiter and Aviel D. Rubin. Crowds: Anonymity for Web Transactions. *ACM Transactions on Information and System Security*, 1(1):66–92, November 1998.
- [SL00] Clay Shields and Brian Neil Levine. A Protocol for Anonymous Communication over the Internet. In *Proc. of ACM Conference on Computer and Communication Security (CCS'00)*, Athens, Greece, November 2000.
- [SSW<sup>+</sup>02] Qixian Sun, Daniel R. Simon, Yi-Min Wang, Wilf Russel, Venkata N. Padmanabhan, and Lili Qiu. Statistical Identification of Encrypted Web Browsing Traffic. In *Proc. of the 2002 IEEE Symposium on Security and Privacy (S&P02)*, 2002.
- [STRL01] Paul Syverson, Gene Tsudik, Michael Reed, and Carl Landwehr. Towards an Analysis of Onion Routing Security. In H. Federrath, editor, *Designing Privacy Enhancing Technologies*, volume 2009 of *LNCS*, pages 96–114. Springer-Verlag, 2001.
- [TN98] S. Thomson and T. Narten. RFC2462: IPv6 Stateless Address Autoconfiguration. IETF, December 1998.
- [WWR01] WWRF. *Book of Visions 2001*. The Wireless World Research team, Paris, December 2001. available at: [www.wireless-world-research.org](http://www.wireless-world-research.org).