WOT for WAT: Spinning the Web of Trust for Peer-to-Peer Barter Relationships*

Kenji Saito
ks91@sfc.wide.ad.jp
Graduate School of Media and Governance
Keio University

January 28, 2005

Abstract

Peer-to-peer complementary currencies can be powerful tools for promoting collaborations and building relationships on the Internet.

i-WAT[13] is a proposed such currency based on the WAT System[18], a polycentric complementary currency using *WAT tickets* as its medium of exchange. Participants spontaneously issue and circulate the tickets as needed, whose values are backed up by chains of trust. *i*-WAT implements the tickets electronically by exchanges of messages signed in OpenPGP[4].

This paper clarifies the trust model of *i*-WAT, and investigates how it is related with that of PGP[17]. To implement the model by dynamically building an appropriate web of trust (WOT), the author claims that it would suffice if the behaviors of participants satisfy the following three properties:

1. *mutual signing by knowing*, or any two mutual acquaintances sign the public keys of each other,
2. *mutual signing by participation*, or the drawer and a user of an *i*-WAT ticket sign the public keys of each other, and
3. *mutual full trust by participation*, or the drawer and a user of an *i*-WAT ticket fully trust each other, and a recipient fully trusts the corresponding user of a ticket, in the context of PGP public key signing.

Likelihood of satisfaction of these properties is supported by the (dis)incentives imposed by the semantics of *i*-WAT.

A reference implementation of *i*-WAT has been developed in the form of an XMPP[6, 7] instant messaging client. We are beginning to put the currency system into practical use.

1 Introduction

1.1 Peer-to-peer complementary currencies and their potential impacts on the Internet

Distributed autonomous (or peer-to-peer) systems, such as an overlay network of people over the Internet, require coordination among participants to achieve their goals. Since each partici-

*This draft is an extended version of the paper with the same title, which is to appear in the IEICE TRANSACTIONS on Communication in April 2005.

pant may behave selfishly to maximize their benefit, *incentive-compatibility*[8], roughly restated as the goal of the system being accomplished by collection of selfish behaviors, becomes important. Because relationships among participants in such a system necessitate fair exchanges of resources, the medium of exchange must take an important role.

Money is a well-known medium of exchange, but its scarsity has caused a lot of problems. *Complementary currencies*, or alternative forms of monetary medium, have been proposed and tested to achieve an autonomous, sustainable local economy even in short of money. There have been succeeding cases, such as experiments in Wörgl in 1932 (stamp money[14]), in Comox Valley in 1983 (Local Exchange Trading System[15]) and in Ithaca since 1991 (Ithaca HOURs[11]).

Many of the outcomes are short-lived, however, because most of the existing complementary currencies are dependent on the qualities of their administrations. It would thus benefit the autonomy and sustainability of economy if we could design an administration-free complementary currency; if we want to make a peer-to-peer world, money too needs to be peer-to-peer.

If such peer-to-peer complementary currencies are applied to the Internet, it would benefit many areas including multicast cost sharing, inter-domain routing, web caching, file sharing, distributed task allocation, and other application-layer overlay networks. Freedom to pursue these possibilities depends on whether we can have a free economic medium or not.

1.2 Contribution of this paper

i-WAT[13] is a proposed such currency based on the WAT System[18], a polycentric complementary currency using *WAT tickets* as its medium of exchange. A WAT ticket is like a bill of exchange, but without a specified redemption date or place. *i*-WAT implements the tickets electronically by exchanges of messages signed in OpenPGP[4].

This paper clarifies the trust model of *i*-WAT, and investigates how it is related with that of PGP[17]. In particular, this paper deduces three properties satisfying which the participants can dynamically extend their webs of trust to accommodate trades using *i*-WAT. This can be reflected to the designs of software tools which implement the *i*-WAT protocol.

2 Background

2.1 Digital signature

Digital signature is an essential technology for designing a dependable economic medium, which can provide a proof of debits or credits.

Throughout this paper, let us use notations from [3] for formalization, with additional abstractions built upon them to fit our purposes.

Suppose Alice (A) is associated with a public/secret key pair denoted as $\langle K_A, K_A^{-1} \rangle$. To simplify the arguments to follow, let us assume that each user has exactly one key pair associated with them.

A digital signature has two objectives:

1. To prove that Alice once admitted a message m.
2. To prove that m has not been altered since then.

These can be realized by encrypting m with Alice's secret key K_A^{-1}, obtaining $\{m\}_{K_A^{-1}}$ which is only decrypted with her public key K_A. Since K_A^{-1} is a secret known only to Alice, those who could decrypt $\{m\}_{K_A^{-1}}$ can infer that it must have been encrypted by Alice. They can also be certain that m has not been altered since Alice made $\{m\}_{K_A^{-1}}$ if the result of decryption equals m.

Usually, for efficiency reasons, instead of encrypting m itself, a digital signature is made by applying a secure hash function H to m, then encrypting the hash value with the secret key. H must be carefully chosen so that it is computationally infeasible to obtain m' where $m' \neq m$ such that $H(m) = H(m')$.

Definition 1 (digital signature) *Let us write* $A \overset{signs}{\rightarrow} m$ *if and only if* A *presents both a plain-text message* m *and its encrypted form* $\{H(m)\}_{K_A^{-1}}$. *The latter is called a* signature *on the former.*

The signature can be verified by Bob if he has a copy of Alice's public key K_A. To verify the signature, he calculates $H(m)$ from m, decrypts $\{H(m)\}_{K_A^{-1}}$ with K_A, and compares the two resulted values.

One question is how Bob can be sure that his copy of Alice's public key is genuine.

Definition 2 (validating relation) $x \overset{v}{\rightarrow} y$ *if* x *possesses a copy of* y*'s public key* K_y*, and infers that the copy is genuine.*

Let us also write $x \overset{v}{\leftrightarrow} y$ *iff* $x \overset{v}{\rightarrow} y \wedge y \overset{v}{\rightarrow} x$ *(*mutually validating relation*).*

A trust model around validity of public keys is a specific definition of *validating relation* $\overset{v}{\rightarrow}$ in the system in concern. Typically, validity of a public key is supported by a *certificate*, or a signature on the key. For example, if Bob (B) sees Cameron (C) such that $B \overset{v}{\rightarrow} C \wedge C \overset{signs}{\rightarrow} K_A$, then $B \overset{v}{\rightarrow} A$ assuming that C's certificate is trustworthy. This relation is recursive, so that someone needs to self-certify at some point.

A public key infrastructure uses a tree of *certificate authorities*, or issuers of certificates, whose public keys are validated by the parent nodes, rooted by a self-certifying authority.

2.2 Web of trust

In a *web of trust*, however, responsibility for validating public keys is delegated to people one trusts, without necessitating certificate authorities. It is a network of people signing the public keys of others.

Signing relation $\overset{s}{\rightarrow}$ states that one certifies that its copy of someone's public key is genuine.

Definition 3 (signing relation) $\overset{s}{\rightarrow}$ *is defined as follows:*

1. $x \overset{s}{\rightarrow} x$
2. $x \overset{s}{\rightarrow} y$ *if* $x \overset{signs}{\rightarrow} K_y$

Let us also write $x \overset{s}{\leftrightarrow} y$ *iff* $x \overset{s}{\rightarrow} y \wedge y \overset{s}{\rightarrow} x$ *(*mutually signing relation*).*

Definition 4 (signing-apart relation) $\overset{s[n]}{\rightarrow}$ *is defined as follows:*

1. $x \overset{s[0]}{\rightarrow} x$
2. $x \overset{s[1]}{\rightarrow} y$ *if* $x \overset{s}{\rightarrow} y \wedge x \neq y$.
3. $x \overset{s[a+b]}{\rightarrow} z$ *if there exists* y *such that* $x \overset{s[a]}{\rightarrow} y \wedge y \overset{s[b]}{\rightarrow} z$.

Let us also write $A \overset{s}{\rightarrow} B \overset{s[n]}{\rightarrow} C$ in place of $A \overset{s[n+1]}{\rightarrow} C$ if $A \overset{s}{\rightarrow} B \wedge B \overset{s[n]}{\rightarrow} C$ (expansion of signing-apart relation) in order to clarify who stands in between the chain of signing relations.

Definition 5 (web of trust) *A web of trust for* x *is a set of all* y *such that* $x \overset{s[n]}{\rightarrow} y$ *where* $n \geq 0$.

A specific validation relation needs to be defined over a web of trust. PGP (Pretty Good Privacy) is an example of a cryptographic technology which defines such a relation. Let us use GnuPG[16] as our choice of implementation of OpenPGP[4] standard.

2.3 PGP trust model

Let us further define that $\mathcal{T}_x$ is the set of users x considers fully trustable, and $\mathcal{T}'_x$ is the set of users x considers marginally trustable.

In the context of PGP public key signing, *fully trustable* means that one considers that the owner of a public key has an excellent understanding of key signing, and his or her signature on a key would be as good as their own, and *marginally trustable* means that one considers that the owner of a public key understands the implications of key signing and properly validates keys before signing them[17].

The PGP trust model is a definition of *validating relation* $\overset{v}{\rightarrow}$ over a web of trust.

Definition 6 (PGP trust model) $x \overset{v}{\rightarrow} y$ *if*

1. *sufficient number of valid key owners sign* y*'s public key, i.e.*
   (a) $x \overset{s}{\rightarrow} y$, *or*
   (b) *there exist at least* f *instances of* z *such that* $z \in \mathcal{T}_x$, $x \overset{v}{\rightarrow} z \wedge z \overset{s}{\rightarrow} y$, *or*
   (c) *there exist at least* m *instances of* z *such that* $z \in \mathcal{T}'_x$, $x \overset{v}{\rightarrow} z \wedge z \overset{s}{\rightarrow} y$; *and*
2. $x \overset{s[n]}{\rightarrow} y$ *where* $n \leq h$,

where f, m *and* h *are the required number of fully trusted key owners, required number of marginally trusted key owners, and number of maximum steps in the path in the web of trust tracing* x *back from* y, *respectively.*

Let us define the marginally validating relation ($\overset{v}{\rightarrow}$) as follows (although this is not used in the design of our currency):

Definition 7 (weak PGP trust model) $x(\overset{v}{\rightarrow})y$ *if*

1. *insufficient number of valid key owners sign* y*'s public key, i.e.*
   (a) *there exist at least one but less than* f *instances of* z *such that* $z \in \mathcal{T}_x$, $x \overset{v}{\rightarrow} z \wedge z \overset{s}{\rightarrow} y$, *or*

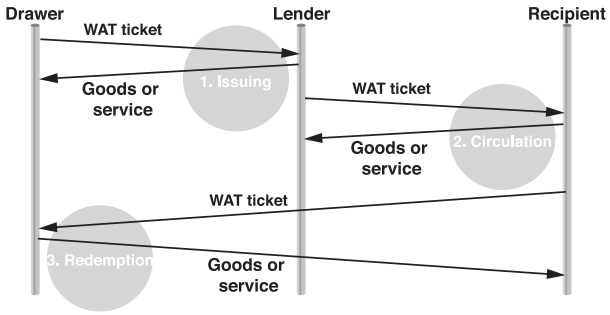

Figure 1: Three stages of trading with a WAT ticket

(b) there exist at least one but less than m instances of z such that $z \in \mathcal{T}'_x$, $x \xrightarrow{v} z \wedge z \xrightarrow{s} y$; and

2. $x \xrightarrow{s[n]} y$ *where* $n \leq h$

By default, GnuPG defines $f = 1$, $m = 3$ and $h = 5$.

2.4 The WAT System

2.4.1 Overview

The WAT System[18] is a complementary currency designed by Mr. Eiichi Morino, the founder of Gesell Research Society Japan[10]. A *WAT ticket*, a physical sheet of paper resembling a bill of exchange, is used as the medium of exchange in the system.

A lifecycle of a WAT ticket involves three stages of trade (illustrated in Figure 1):

1. Issuing – the birth of a WAT ticket

   A *drawer* issues a WAT ticket by writing on an empty form the name of the provider (*lender*) of the goods or service, the amount of debt[1], the present date, and the drawer's signature. The drawer gives the ticket to the lender, and in return obtains the goods or service.

2. Circulation – ordinary exchange

   The person to whom the WAT ticket was given can become a *user*, and use it for another trading. To do so, the user writes the name of the recipient, as well as their own, on the reverse side of the ticket. The recipient will become a new user, repeating which the WAT ticket circulates among people.

3. Redemption – the return of the WAT ticket

   The WAT ticket is invalidated when it returns, as a result of a trade, to the drawer.

Figure 2 shows the state machine of a WAT ticket.

[1]Typically in the unit kWh, which represents cost of producing electricity from natural energy sources.

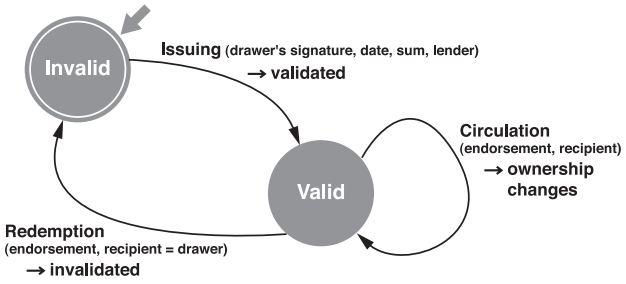

Figure 2: State machine of a WAT ticket

2.4.2 Distinctive features

Autonomy Anyone can spontaneously become a member of the WAT System with a sheet of paper if they follow the above protocol.

Compatibility A WAT ticket is compatible with any other WAT tickets in the world, so that the currency system is operable globally, as long as the drawer can be credited.

Extensibility The protocol illustrated in Figure 1 and 2 defines *the WAT Core*, the essence of the WAT System. An *extended part* can be defined for a new currency based on the WAT System, stating, for example, the region, group and duration in which the tickets are usable, as well as the unit in which the debit is quantified.

Security In case the drawer fails to meet their promise on the ticket, the lender assumes the responsibility for the debit. If the lender fails, the next user takes over. The responsibility follows the chain of endorsements. The longer the chain is, the more firmly backed up the ticket is. Therefore the length of the chain of endorsements represents the extent of trust the ticket has gained.

3 *i*-WAT: the Internet WAT System

3.1 Overview

i-WAT is a translation of the WAT Core onto the Internet. In *i*-WAT, messages signed in OpenPGP are used to implement transfers of an electronically represented WAT ticket. The exchanged messages are called *i-WAT messages*, and the ticket represented by the messages is called an *i-WAT ticket*.

An *i*-WAT ticket contains the identification number, amount of debt and public key user IDs of the drawer, users and recipients. Endorsements are realized by nesting PGP signatures as illustrated in Figure 3.

Table 1 shows the types of *i*-WAT messages. All *i*-WAT messages are signed by the senders, and are formatted in the canonical form[1] of XML[2] with nested signatures. The messages cause state transfers of an *i*-WAT ticket as illustrated in Figure 4.

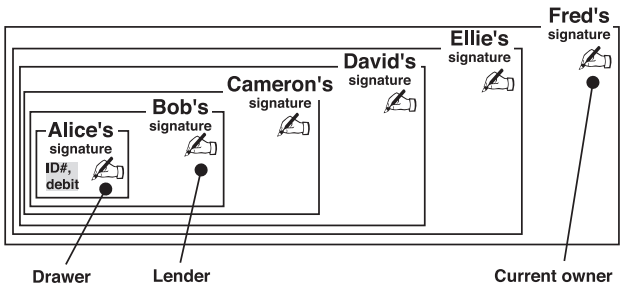

Figure 3: Signature chain in an *i*-WAT ticket

Table 1: *i*-WAT messages

message	sender	receiver	function
<draw/>	drawer	recipient (lender)	draws an *i*-WAT ticket.
<use/>	user	recipient	uses an *i*-WAT ticket.
<accept/>	recipient	drawer and user	confirms readiness to accept the *i*-WAT ticket once it is validated.
<reject/>	recipient	drawer or user*	rejects an *i*-WAT ticket.
<approve/>	drawer	user and recipient	validates an *i*-WAT ticket, and approves the transaction.
<disapprove/>	drawer	user and recipient	denies an *i*-WAT transaction.

* depending on whether the ticket has just been issued or in circulation, respectively.

3.2 Changes from the WAT System

3.2.1 Changes in the state machine

Upon translating the WAT Core onto the digital communication domain, the author has made the following changes from the state machine of a WAT ticket:

1. Trades need to be asynchronously performed. Intermediate states, such as waiting for acceptance or approval, are introduced.

2. Double-spending needs to be prohibited. The drawer is made responsible for guaranteeing that the circulating ticket is not a fraud. This means that every trade has to be approved by the drawer of the involved ticket.

3.2.2 Justification of the design

The author regards these changes as necessary modifications to design a dependable economic medium. But they, especially the latter one, may introduce bottlenecks in the system.

This issue has not been quantitatively analyzed yet. However, by a casual analysis, the author believes that the design is justifiable because anyone can spontaneously become a drawer as long as they are trusted, and the relation between their trust and the processing load is incentive-compatible:

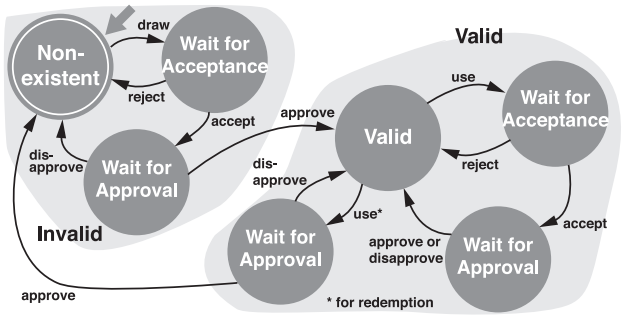

Figure 4: State machine of an *i*-WAT ticket

1. If everyone is trusted equally in a circle of friends, the load should be evenly distributed among participants.

2. Otherwise, the load should be distributed in an incentive-compatible way:

   (a) If one is late to respond (thus avoids to comply with the imposed load), or tends to fail to answer requests for redemption, they will lose trust from others. Then it will become more difficult for them to have their tickets accepted for trades in the future.

   (b) If one is quick to respond, and accepts requests for redemption with certainty, they will gain more trust from others. Since their tickets become easier to use, they may attract more load. But it will become easier for them to draw tickets at will in the future, and initiate trades spontaneously to obtain goods or services.

3.3 Protocol

3.3.1 Issuing – the birth of an *i*-WAT ticket

1. The drawer sends a <draw/> message which contains the public key user IDs of the drawer and lender, identification number and amount of debt. This message becomes the original *i*-WAT ticket after the protocol is completed.

2. The lender sends back the content of the message as an <accept/> message.

3. The drawer sends an <approve/> message to the lender.

3.3.2 Circulation – ordinary exchange

1. The user adds to the *i*-WAT ticket the public key user ID of the recipient, and sends it to the recipient as a <use/> message. This message becomes a valid *i*-WAT ticket after the protocol is completed.

2. The recipient forwards the content of the message to the drawer and user as an <accept/> message.

3. The drawer verifies the ticket, and sends an <approve/> message to the user and recipient.

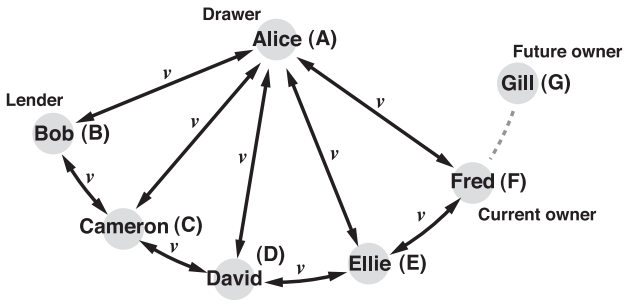

Figure 5: *i*-WAT trust model

3.3.3 Redemption – the return of the *i*-WAT ticket

1. The user sends a <use/> message to the recipient, who equals the drawer.
2. The drawer verifies the ticket, and invalidates it as the debit is now redeemed. The drawer sends an <approve/> message to the user.

4 *i*-WAT and the PGP trust model

4.1 *i*-WAT trust model

Let us define that $t(x)$ is an *i*-WAT ticket t drawn by x, $\mathcal{U}_{t(x)}$ is the set of users throughout the lifecycle (up to redemption) of $t(x)$, and $y \overset{t(x)}{\rightarrow} z$ denotes that y gives $t(x)$ to z as a result or promise of a trade.

The *i*-WAT trust model is a definition of how *mutually validating relation* $\overset{v}{\leftrightarrow}$ must hold over a network of participants.

Definition 8 (*i*-WAT trust model) *for every* $t(x)$*,*

1. *for all* y *such that* $y \in \mathcal{U}_{t(x)}$*,* $x \overset{v}{\leftrightarrow} y$
2. *for all* y, z *such that* $\{y, z\} \subseteq \mathcal{U}_{t(x)}$*,* $y \overset{v}{\leftrightarrow} z$ *if* $y \overset{t(x)}{\rightarrow} z$

Figure 5 illustrates the model by an example. This model is naturally induced from the necessity for the participants to validate *i*-WAT messages.

4.2 Spinning the web of trust – preconditions

If the PGP trust model over the network of participants does not readily support the above model, the model needs to be implemented by dynamically building an appropriate web of trust. In order to do so, the author claims that it suffices (but not necessitates) if the behaviors of the participants satisfy the following properties.

Property 1 (mutual signing by knowing) *for every* x *and* y*,*

- $x \overset{s}{\leftrightarrow} y$ *if* x *knows*[2] y

Intuitively, this states that any two mutual acquaintances sign the public keys of each other.

Property 2 (mutual signing by participation) *for every* $t(x)$*,*

- *for all* y *such that* $y \in \mathcal{U}_{t(x)}$, $x \overset{s}{\leftrightarrow} y$

Intuitively, this states that the drawer and a user sign the public keys of each other.

Property 3 (mutual full trust by participation) *for every* $t(x)$*,*

1. *for all* y *such that* $y \in \mathcal{U}_{t(x)}$, $x \in \mathcal{T}_y \wedge y \in \mathcal{T}_x$
2. *for all* y, z *such that* $\{y, z\} \subseteq \mathcal{U}_{t(x)}$, $y \in \mathcal{T}_z$ *if* $y \overset{t(x)}{\rightarrow} z$

Intuitively, this states that the drawer and a user are confident about each other that their correspondents have an excellent understanding of key signing, and a recipient is confident about the corresponding user that they have such an excellent understanding. They need to reflect such views in their PGP trust databases.

Let us assume that GnuPG's default values are used for variables f, m and h.

4.3 Spinning the web of trust – case studies

Let us justify the above claim by case studies. Throughout the studies, the network of participants in Figure 5 is used as an example. It is assumed that no external source of information is available.

Our goal is to show that the i-WAT trust model is satisfied in every stage of trades starting from likely initial states, i.e., a joining party knows someone in the network of participants, if the properties explained in section 4.2 are satisfied by the participants.

The statement that follows each claim is both a casual proof and a procedure to achieve the goal.

4.3.1 Issuing

The goal is to form the initial network of participants between Alice and Bob.

Claim 1 $A \overset{v}{\leftrightarrow} B$ *results if Alice knows Bob.*

In case Alice knows Bob

1. By *mutual signing by knowing*,
$$A \overset{s}{\leftrightarrow} B$$
2. By the definition 1a of *PGP trust model*,
$$A \overset{v}{\leftrightarrow} B$$

[2]In the context of this paper, *knows* relation is defined to be symmetrical, i.e., y knows x if x knows y.

4.3.2 Circulation

The goal is to let Gill join the existing network of participants.

Claim 2 *$G \overset{v}{\leftrightarrow} F \wedge G \overset{v}{\leftrightarrow} A$ results if Gill knows either Fred or Alice, or someone (in $\mathcal{T}_G$) or some people (in ${\mathcal{T}'}_G$) in the network of participants.*

In case Gill knows Fred

1. By *mutual signing by knowing* and *mutual signing by participation*,
$$G \overset{s}{\leftrightarrow} F \wedge F \overset{s}{\leftrightarrow} A$$
2. By *expansion of signing-apart relation*,
$$G \overset{s}{\leftrightarrow} F \wedge G \overset{s}{\leftrightarrow} F \overset{s}{\leftrightarrow} A$$
3. By the definition 1a of *PGP trust model*,
$$G \overset{v}{\leftrightarrow} F \wedge G \overset{s}{\leftrightarrow} F \overset{s}{\leftrightarrow} A$$
4. $F \in \mathcal{T}_A$ and $F \in \mathcal{T}_G$ by the properties 1 and 2 of *mutual full trust by participation*, respectively. Also, the path length between Gill and Alice is shorter than h. Therefore, by the definition 1b of *PGP trust model*,
$$G \overset{v}{\leftrightarrow} F \wedge G \overset{v}{\leftrightarrow} A$$

In case Gill knows Alice

1. By *mutual signing by knowing* and *mutual signing by participation*,
$$G \overset{s}{\leftrightarrow} A \wedge A \overset{s}{\leftrightarrow} F$$
2. By *expansion of signing-apart relation*,
$$G \overset{s}{\leftrightarrow} A \wedge G \overset{s}{\leftrightarrow} A \overset{s}{\leftrightarrow} F$$
3. By the definition 1a of *PGP trust model*,
$$G \overset{v}{\leftrightarrow} A \wedge G \overset{s}{\leftrightarrow} A \overset{s}{\leftrightarrow} F$$
4. $A \in \mathcal{T}_G$ and $A \in \mathcal{T}_F$ by the property 1 of *mutual full trust by participation*. Also, the path length between Gill and Fred is shorter than h. Therefore, by the definition 1b of *PGP trust model*,
$$G \overset{v}{\leftrightarrow} A \wedge G \overset{v}{\leftrightarrow} F$$

In case Gill knows neither Alice nor Fred The goal can still be met if

1. there is one user x such that $x \in \mathcal{U}_{t(A)}$ who knows Gill and appeared earlier than Fred, and $x \in \mathcal{T}_G$, or
2. there are three users x, y, z such that $\{x, y, z\} \subset \mathcal{U}_{t(A)}$ who all know Gill and appeared earlier than Fred, and $\{x, y, z\} \subseteq \mathcal{T'}_G$.

The proofs for the above two cases are similar; they both involve first establishing $G \overset{v}{\leftrightarrow} A$ by way of someone or some people in the middle, only that the latter is more complex.

Suppose Gill knows Cameron, David, Ellie, and marginally trust them.

1. By *mutual signing by knowing* and *mutual signing by participation*,
$$G \overset{s}{\leftrightarrow} C \wedge C \overset{s}{\leftrightarrow} A \wedge\ A \overset{s}{\leftrightarrow} F$$
2. By *expansion of signing-apart relation*, and by the definition 1a of *PGP trust model*,
$$G \overset{v}{\leftrightarrow} C \wedge C \overset{s}{\leftrightarrow} A \wedge\ G \overset{s}{\leftrightarrow} C \overset{s}{\leftrightarrow} A$$
$$\wedge\, G \overset{s}{\leftrightarrow} C \overset{s}{\leftrightarrow} A \overset{s}{\leftrightarrow} F$$
3. The above also holds if we replace C with D or E. It is given that $\{C, D, E\} \subseteq \mathcal{T'}_G$. Also, the path length between Gill and Alice is shorter than h. Therefore, by the definition 1c of *PGP trust model*,
$$G \overset{v}{\rightarrow} A \wedge C \overset{s}{\leftrightarrow} A \wedge\ G \overset{s}{\leftrightarrow} C \overset{s}{\leftrightarrow} A$$
$$\wedge\, G \overset{s}{\leftrightarrow} C \overset{s}{\leftrightarrow} A \overset{s}{\leftrightarrow} F$$
4. $C \overset{v}{\leftrightarrow} A$ by the definition 1a of *PGP trust model*. $C \in \mathcal{T}_A$ by the property 1 of *mutual full trust by participation*. Therefore, by the definition 1b of *PGP trust model*,
$$G \overset{v}{\leftrightarrow} A \wedge\ G \overset{s}{\leftrightarrow} C \overset{s}{\leftrightarrow} A \overset{s}{\leftrightarrow} F$$
5. Now that Gill and Alice mutually validate their public keys, they can establish $G \overset{s}{\leftrightarrow} A$ by *mutual signing by participation*.
$$G \overset{v}{\leftrightarrow} A \wedge\ G \overset{s}{\leftrightarrow} A \overset{s}{\leftrightarrow} F$$
6. $A \in \mathcal{T}_G$ and $A \in \mathcal{T}_F$ by the property 1 of *mutual full trust by participation*. Also, the path length between Gill and Fred is shorter than h. Therefore, by the definition 1b of *PGP trust model*,
$$G \overset{v}{\leftrightarrow} A \wedge G \overset{v}{\leftrightarrow} F$$

4.3.3 Redemption

The goal is to complete the lifecycle of the ticket in concern. This needs to be done without further expanding the existing network of participants.

Claim 3 *$G \overset{v}{\leftrightarrow} A$ results without extending the web of trust any further.*

1. By *mutual signing by participation*,
$$G \overset{s}{\leftrightarrow} A$$
2. By the definition 1a of *PGP trust model*,
$$G \overset{v}{\leftrightarrow} A$$

4.4 Justification of the preconditions

Let us casually explain how the preconditional properties are supported by the natural behaviors of people. The formal proof that the design of *i*-WAT is incentive-compatible is left out for a future work.

4.4.1 Mutual signing by knowing

If two parties know each other (well enough), it should be possible to safely exchange the fingerprints[3] of their public keys. Therefore this is only a question of the communication cost.

4.4.2 Mutual signing by participation

Because it becomes easier for other people to join the circle of friends around an *i*-WAT ticket if this property is met, both the drawer and user have incentives to sign each other's public keys after properly validating them.

4.4.3 Mutual full trust by participation

The participants are motivated to fully trust their correspondents in the context of public key signing by the same incentives as the above. Also, they are disincentivized to be negligent of the precautions for signing public keys, in order to protect themselves from possible attacks by impostors.

4.5 Possible attacks

4.5.1 Overview

Attacks may be possible by compromising the preconditional properties. When *mutual signing by knowing* or *participation* is compromised, it would result in a failure to build an appropriate web of trust. This is fail-safe, so that no harm will result for other members of the system. This is not the case when *mutual full trust by participation* is compromised. The system should be strong enough to prevent mishaps from resulting when participants show untrustworthy behaviors with respect to signing public keys.

The author claims that a web of trust to support the *i*-WAT trust model protects itself from such a threat.

[3] A fingerprint is a hash value of a key so that the key's identity can be checked with small cost.

Table 2: Cases of possible attacks

#	signing	compromised	forged key	forger	prevented by
1	$B \xrightarrow{s} A$	$B \in \mathcal{T}_C$	$\langle K_A, K_A^{-1} \rangle$	an impostor	Bob
2	$B \xrightarrow{s} A$	$B \in \mathcal{T}_C$	$\langle K_A, K_A^{-1} \rangle$	Bob (Alice is imaginary)	Bob, Cameron
3	$A \xrightarrow{s} B$	$A \in \mathcal{T}_C$	$\langle K_B, K_B^{-1} \rangle$	an impostor	Alice
4	$A \xrightarrow{s} B$	$A \in \mathcal{T}_C$	$\langle K_B, K_B^{-1} \rangle$	Alice (Bob is imaginary)	Cameron
5	$B \xrightarrow{s} C$	$B \in \mathcal{T}_A$	$\langle K_C, K_C^{-1} \rangle$	an impostor or Bob	Bob
6	$A \xrightarrow{s} C$	$A \in \mathcal{T}_B$	$\langle K_C, K_C^{-1} \rangle$	an impostor or Alice	Alice

4.5.2 Case studies

Table 2 shows the cases of possible attacks to exploit the first three people appearing in the network of participants in Figure 5.

An impostor forges Alice's public key The impostor receives goods or service from Bob in return of an empty promise. Bob is incentivized to obtain the fingerprint of Alice's public key directly from her and compare it with that of the forged public key. Otherwise, the debit will become his responsibility.

Bob creates Alice, and forges her public key Bob tries to make an *i*-WAT ticket with an empty promise, and use it against Cameron. However, Bob is disincentivized to create Alice in the first place, because he must take the responsibility of the debit once people discover that Alice is not able to repay. Or he could escape, so that Cameron is incentivized to keep in touch with Bob to make him more traceable.

An impostor forges Bob's public key The impostor receives an *i*-WAT ticket Alice issues without giving her anything in return. Although she can always disapprove further trades with the ticket, she will lose her trust because people can infer that she was a careless signer, making it more difficult for her to participate in future trades. Alice is incentivized to be careful.

Alice creates Bob, and forges his public key Issuing is always the hardest part. Alice tries to make her *i*-WAT tickets easier to circulate by skipping the step. If, for any reason, she fails to meet her promise on the ticket, Cameron must take over the responsibility as Bob does not exist. This can easily be considered an attack to Cameron. Therefore, Cameron is incentivized to keep in touch with Alice to make her more traceable in case she escapes.

An impostor or Bob forges Cameron's public key Someone pretends to be or creates Cameron, and receives the *i*-WAT ticket from Bob giving nothing in return. Bob is incentivized to be careful, and disincentivized to create Cameron because whatever the imaginary friend causes, people would first suspect Bob.

An impostor or Alice forges Cameron's public key Someone pretends to be or creates Cameron, and receives the *i*-WAT ticket from Bob giving nothing in return. Alice is incentivized to be careful because she is the first to be blamed, and disincentivized to create Cameron because when someone disappears with a valid ticket, it means Alice does not have to repay, so that people would consider that she has a motive.

5 Deployment

5.1 Overview

i-WAT allows the underlying carrier of messages to be existing e-mail or instant messaging systems. As a reference implementation, the author has developed an *i*-WAT plug-in and the hosting XMPP (Extensible Messaging and Presence Protocol)[6, 7] client called *wija*. The software is available from http://www.media-art-online.org/wija/ (the *i*-WAT plug-in is bundled with all platform-specific packages).

i-WAT and a public key exchange mechanism to support the system have been implemented as extensions to XMPP instant messaging protocol.

The reference implementation has already been in use by the WAT System communities. It has been used, for example, to exchange goods, such as books, with services, such as working hours for developing an open source software, namely *wija* itself.

5.2 Support for the preconditional properties

Our software lets users exchange their public keys directly (by way of XMPP servers) without consulting a public key server. From a user's point of view, this is performed by choosing a correspondent from a buddy list, and selecting either importing or exporting their keys. When imported, a window pops up with the fingerprint of the public key, asking the user whether to sign the key or not. This is expected to enhance the ease for *mutual signing by knowing*.

Our software currently does not directly support *mutual signing by participation*. However, the current design of the software uses the buddy list to locate the owner of a public key, so that new participants will be required to add the drawer in their buddy list if they have not already, to which the drawer would respond by adding them back. Then the above mechanism for key exchange can be used.

Our software currently does not have a support for *mutual full trust by participation*.

6 Future work

Our development team will add a user interface to *wija* to support *mutual signing* and *mutual full trust by participation*. We will experiment further how we can reduce the communication cost so that people can easily satisfy the three preconditional properties. At the same time, as we put *i*-WAT into practical use, we will see if these properties are actually useful for building up the circle of friends around an *i*-WAT ticket.

7 Related work

7.1 Magic Money

Magic Money[19] is an example of message-based currencies on the Internet based on PGP signatures. It was designed and implemented by an anonymous programmer known as Pr0duct Cypher in the early 1990s. Although there were a few enthusiasts, the use of Magic Money did not spread widely for several reasons:

1. It utilized Chaum's blind signature protocol[5] which was patented at the time. Since Magic Money was distributed as a free, open source software, its existence itself was unlawful.
2. It required presence of a server, which had to be maintained by someone.
3. It pursued untraceability while there was nothing to back up the values of the digital coins. The system was regarded as untrustworthy.

The author regards Magic Money as an important experience of deploying a complementary currency on the Internet, and has tried to do the opposites: 1) chosen GnuPG as the implementation of OpenPGP which does not use patented technologies, 2) chosen not to rely on servers (we use XMPP servers as routers), and 3) chosen to give up anonymity to some extent (public key IDs and signing relations are made known) to build up trust instead.

7.2 Geek Credit

Geek Credit[12] is an example closer to *i*-WAT. It defines *Geek Credit policy*, which is similar to the *i*-WAT state machine, but the problem of double-spending is handled differently. Geek Credit detects double-spending at redemption, so that each trading does not need to be consulted with the drawer.

While this simplifies the protocol, the risk of attacks is higher for Geek Credit than for *i*-WAT. Having not to consult the drawer also makes the trust model of Geek Credit simpler, but it means that there is no implicit support for building the web of trust dynamically other than joining the circle of friends by knowing the current owner of the ticket. Since the drawer does not have a way to check the usage of their tickets, there is no way to enforce the imposed restrictions by an extended part if there is one.

7.3 Ripple

Ripple[9] is another example of a decentralized currency system. It finds a chain of credit connections between parties to make payments. If A is trusted by B and B is trusted by C, and if A wants to make a payment to C, then A pays to B so that B pays to C.

This may work if everyone in the found chain is present on the Internet. The author does not see that as a big problem; *i*-WAT has a similar assumption of the drawers being present on the Internet most of the time.

The behaviors of the two currency systems would look similar in that if A and C do not yet know each other, the system depends on the intermediate person B to secure the trustworthiness between A and C. But *i*-WAT does so by checking the signatures of B on the public keys of A

and C, so that it does not require the presence of B when A and C want to make a transaction. Therefore *i*-WAT should be both more efficient and more tolerant of failures.

Currently, there is no working implementation of Ripple.

8 Conclusions

Peer-to-peer complementary currencies can be powerful tools for promoting collaborations and building relationships on the Internet. *i*-WAT is a proposed such currency based on the WAT System, a polycentric complementary currency using WAT tickets whose values are supported by chains of trust.

This paper clarified the *i*-WAT trust model. To implement the model by dynamically building an appropriate web of trust, the author showed that it would suffice if the behaviors of participants satisfy the following three properties:

1. *mutual signing by knowing*
2. *mutual signing by participation*
3. *mutual full trust by participation*

Likelihood of satisfaction of these properties is supported by the (dis)incentives imposed by the semantics of *i*-WAT.

The author has developed an XMPP client called *wija* in order to put *i*-WAT into practical use. The author's team has been experimenting on user interfaces for exchanging publice keys, so that participants of *i*-WAT can satisfy the above properties with little or no subjective communication cost.

Acknowledgement

The author would like to thank Mr. Eiichi Morino and other members of Gesell Research Society Japan for valuable advices and discussions on the content of this paper, as well as feedback on *wija* software.

References

[1] John Boyer. *Canonical XML Version 1.0*, March 2001. W3C Recommendation. Available electronically at http://www.w3.org/TR/xml-c14n.

[2] Tim Bray, Jean Paoli, C.M.Sperberg-McQueen, and Eve Maler. *Extensible Markup Language (XML) 1.0 (Second Edition)*, October 2000. W3C Recommendation. Available electronically at http://www.w3.org/TR/REC-xml.

[3] Michael Burrows, Martin Abadi, and Roger Needham. A logic of authentication. In *Proceedings of the Royal Society, Volume 426, Number 1871*, 1989.

[4] Jon Callas, Lutz Donnerhacke, Hal Finney, and Rodney Thayer. *OpenPGP Message Format*, November 1998. RFC 2440.

[5] David Chaum. Blind signatures for untraceable payments. In *Advances in Cryptology – Crypto '82*. Springer-Verlag, 1983.

[6] Peter Saint-Andre (Ed). *Extensible Messaging and Presence Protocol (XMPP): Core*, October 2004. RFC 3920.

[7] Peter Saint-Andre (Ed). *Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence*, November 2004. RFC 3921.

[8] Joan Feigenbaum and Scott Shenker. Distributed algorithmic mechanism design: Recent results and future directions. In *Proceedings of the 6th International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communication (DIALM '02)*, September 2002.

[9] Ryan Fugger. Money as IOUs in social trust networks & a proposal for a decentralized currency network protocol. Hypertext document. Available electronically at http://ripple.sourceforge.net/.

[10] Gesell Research Society Japan. Hypertext document. Available electronically at http://www.grsj.org/.

[11] Paul Glover. Ithaca HOURs Online. Hypertext document. Available electronically at http://www.ithacahours.com/.

[12] Alexander Komarov. Geek Credit homepage. Hypertext document. Available electronically at http://home.gna.org/geekcredit/.

[13] Kenji Saito. Peer-to-peer money: Free currency over the Internet. In *Proceedings of the Second International Conference on Human.Society@Internet (HSI 2003), Lecture Notes in Computer Science 2713*. Springer-Verlag, June 2003.

[14] Fritz Schwarz. Das experiment von Wörgl, 1951. Hypertext document. Available electronically at http://userpage.fu-berlin.de/∼roehrigw/woergl/, *(Shortened English translation by Hans Eisenkolb is available at http://www.sunshinecable.com/ ∼eisehan/woergl.htm).*

[15] Sidonie Seron. Local Exchange Trading Systems 1 - CREATION AND GROWTH OF LETS. Hypertext document. Available electronically at http://www.gmlets.u-net.com/resources/sidonie/ home.html.

[16] The Free Software Foundation. The GNU Privacy Guard. Hypertext document. Available electronically at http://www.gnupg.org/.

[17] The Free Software Foundation. The GNU Privacy Handbook. Available electronically at http://www.gnupg.org/.

[18] watsystems.net. WATSystems home page. Hypertext document. Available electronically at http://www.watsystems.net/.

[19] Mark Woodcock. Magic Money. Hypertext document. Available electronically at http://www.csee.umbc.edu/ ∼woodcock/cmsc482/proj1/ magmoney.html.