

Simulation of Complex Systems - Rumor spreading in simulated PGP trust networks

A. ANGLADE* B. BLEUZÉ† O. MEHANI‡

December 20, 2005

Contents

1	Introduction	2
1.1	What is PGP ?	2
1.2	What is the Web of Trust ?	2
1.3	Method	2
2	Model	3
2.1	Agent	3
2.1.1	Definition of an agent	3
2.1.2	3 local trust levels	3
2.1.3	3 behavior schemes	4
2.2	Scheduler	5
2.2.1	Several meetings types	5
2.2.2	Agent selection	5
2.2.3	Introduction of fakes	5
3	Validity of the model	5
3.1	Analysis of the strong set	5
3.1.1	Degree distribution	6
3.1.2	MSD distribution	6
3.2	The Leaf of Trust	6
4	Results	12
4.1	Some small networks	12
4.2	Rumor spreading	12
4.2.1	Evolution in time	12
4.2.2	Signing parties	14
4.2.3	Number and repartition of agents	15
5	Further enhancements and conclusion	17

*830222-6108

†821012-1391

‡820629-C935 / 820629-6876

1 Introduction

1.1 What is PGP ?

PGP stands for *Pretty Good Privacy*. It is a computer program which has been first released in 1991. The main developer of PGP, Phil Zimmerman, made this notice that it was almost impossible for “normal” people to ensure either the security of their communications and the identity of their correspondent¹ on the internet.

The solution proposed by PGP was to give an easy access to cryptography. People who want to communicate in a secure manner can use PGP in order to

- digitally sign documents to prove they are the actual author;
- receive encrypted documents only they can decrypt.

Somebody willing to do that will generate a *pair of keys*, a private and a public one. One can then use the *private* key to sign one’s own documents and decrypt the ones encrypted for him. One of his correspondent, willing either to verify the signature of encrypt a document will use one’s *public* key.²

For this scheme to work, the private key must not be known by anybody else, but the public key has to be widely distributed so that anybody can find and use it. This last condition is fulfilled by having *keyservers* where everybody can publish one’s own public key and get the others’.

1.2 What is the Web of Trust ?

One problem of the above system is that it is still easy for somebody to generate a fake PGP keypair claiming to belong to somebody else on his behalf. A way to make this kind of forgery impossible to work has to be found

This problem has been solved by allowing people to sign eachothers’ key when they are sure of their identity, for example in a real life meeting. The more signature a public key has, the more confident one will be that it belongs to who it claims to, and then can be used to communicate securely with this person. This solution is called the Web of Trust. This web can be seen as a network where the keys are the nodes and the signatures on the public key the vertices from the signer’s to the signee’s key.

In order to extend the number of signatures on real keys, and eventually spot fake keys, it is necessary to have as much people as possible meeting and exchanging signatures. The mechanism of *signing parties* is commonly used to do so. These events aim to gather a lot of people in a same place and have them sign eachothers’ keys. In terms of networks, this creates a fully connected component.

As this is not always possible to set this kind of events up, another method to grow the Web of Trust is to sign a public key that a sufficient number of trustworthy (according to the local signer) other people have already signed. This method is not 100% sure and one can end up signing fake keys and considering them as real ones.

This effect, a fake key getting signatures and being trusted, can be seen as rumor spreading: the more people sign it, the more other people tend to trust this key, whereas they shouldn’t. We want to study the effect of this rumor spreading on the Web of Trust and see if the keys coming from it can really be trusted as not being forgery.

1.3 Method

To study the Web of Trust and rumor spreading in it, an agent-based model seemed to be the most accurate scheme. Each agent stands for a single person signing others’ keys according to its own “opinions”.

¹It is easy to claim to be whoever one wants in order to get information one should not have access to.

²For more details on public key cryptography, see [1].

Once this model has been built, it was necessary to check its validity, and ensure that it was behaving as the real Web of Trust. This has been done by comparing the properties of both the real and our model-generated network.

Then it has been possible to study the results of introducing fake keys in the Web of Trust building process and study, according to several parameters, how these keys were considered as trustworthy (by counting the number of signatures on these keys).

2 Model

In our model, an agent is a person owning a PGP key and interacting with the other agents by signing their keys or not, according to its “trust” in the validity of these keys. The best way to implement such a model was to use object-oriented programming, because each agent is an independent entity with characteristics (for example its key and the signatures made by the others on its key) *ie.* instance variables, and reactions to situations (for example meeting a key he can sign or not) *ie.* methods. In the same way the scheduler is also an object which instance variables are the conditions of the simulation and which methods are the meetings of the agents. We then chose to implement this model in Java.

2.1 Agent

2.1.1 Definition of an agent

In our model an agent is a PGP key (associated with its owner) and it is defined by:

a key (number) which is an unique integer. Two agents can't have the same key number.

an ID which is also an integer in our model but in real life it is the name of the person who owns the key. This number is not unique, one agent can pretend to be the key of someone else. In such a situation the fake key and the real key would have a different key number (because the key is unique) but the same ID.

a list of signatures made by the other agents on that key because they trust it is valid. This list is public, which means that anyone can know the signatures made on a key (by consulting a key server, in real life).

a keyring which contains the list of keys (agents) signed by the owner of this key and the level of trust it has in each of them. This list is private. Only the owner of the key can know this information. Even the agent whose key has been signed don't know the level of trust the signer gave to it.

2.1.2 3 local trust levels

In section 1.2 we saw that in order to sign keys online (*ie.* not during real life meetings) it is essential that the agents associate a level of trust they have in the other agents. For example if I know that someone, one of my friends, signs keys carefully, I can trust his signatures fully and so when I meet a key I don't know which is signed by this friend I can consider it is valid and maybe sign it also. In this situation there are two kinds of trust: the trust in the behavior of the owner of a key (for example I trust my friend to carefully sign keys) and the trust in the validity of a key (for example I am sure this key is not a fake key).

To simplify the model we chose to merge these two kinds of “trust” in a single one: when an agent signs a key he puts the (local) level of trust it has in the validity of the key which is also the level of trust he will have in *the behavior of the owner* of this key. This is a simplification of the reality but it has then been validated (see 3 on page 5) by the fact that our model built a Web of Trust similar to what can be seen in the Real World.

So we finally chose to have **3 levels of trust**:

full when the agent trusts the key fully;

marginal when the agent is almost sure the key is valid;

unknown when the agent can't say if the key is valid or not. It means he does not have information validating the key (typically it is the first time the agent encounters this key and it is not signed by any of his friends) *and* he also doesn't have information saying that this key is a fake (he doesn't know any valid key with the same ID for example).

It is important to remember that these levels of trust are *local*, *ie.* only the agent who signs the keys knows the level of trust he puts on them.

2.1.3 3 behavior schemes

To simulate the different behaviors of the signers we chose to have three kinds of agents:

Strict only signs a key when he encounters it in real life meetings (individual meetings or keysigning parties). He is sure of the validity of a key he signs because he could verify the identity of its owner (by checking his passport or ID), so **the level of trust he puts on each key he signs is full**.

Laxist signs whichever encountered key. During a real life meeting such an agent signs fully whichever key (same behavior than the strict agent in that case). During an online meeting, a laxist agent previously checks if he has signed a key with the same ID than the new one. If it is not the case he trusts it marginally (it doesn't matter what the web of trust "thinks" of this key). If it is the case he simply chooses to trust marginally the key with the highest number of signatures. Thus a laxist agent is coherent, he doesn't sign two keys with the same ID but his verification of the validity of a key is quite simple and so can be wrong (if the fake key is more signed than the real one).

Prudent trusts the Web of Trust. During a real life meeting he signs fully whichever key he encounters. Before signing online a new key a prudent agent consults his keyring to see if the agents he trusts have signed this key. In our model we chose to simulate it by a simple rule **R** (inspired by a common rule some key-owners use in real life to determine the validity of a key): if one person I personally fully trust has signed this new key (condition C_1) or if three people I trust marginally have signed this key (condition C_2) I sign it marginally, if not (C_1 or C_2) I sign it with the level "unknown". **R** is applied if the met key has a new ID (I haven't already signed a key with the same ID). If I have already signed a key with the same ID with a full level of trust (which means in a real life meeting), I trust the old key and I don't trust the new one, otherwise I trust the new key (and I don't trust the old one) if it verifies the condition C_1 or the condition C_2 . This verification of the validity of a key can also be wrong (if my friends sign fake keys) but it is more prudent (because it is more restrictive) than the verification of the laxist agent.

It is important to notice that an agent can revoke a signature he made on a key. For example if he signs a fake key online and then meets the real key and its owner in a real life meeting, he is sure the previous key was a fake and so doesn't trust it anymore and revokes his signature on this key (in real life by contacting the key server). More generally a revocation occurs when an agent decides to trust a new key with the same ID than a key he already signed, he revokes his signature on the old key. When a signature is revoked it simply doesn't appear anymore in the list of signatures made on this key.

2.2 Scheduler

The main organ of the model is the *scheduler* which simulates the meeting of the key-holders. It first generates a list of agents and fakes then lets time pass. Every time step, a number of agents is selected from the list, and are proposed to sign another agent's key. Whether the signature is actually made depends on both the type of meeting is taking place and the local agent rules for signing.

2.2.1 Several meetings types

As explained in the introduction, there are several possible ways to get to sign a public key.

Real life meeting is the first mechanism that builds up the Web of Trust. Two agents meet each other, exchange proofs of their identity and then can sign eachother's key and fully trust it to be a real one.

Online keysigning can be considered as the most current signing event in the real Web of Trust. An agent encounters the public key of another and checks the signatures already present on this key. If there are enough signatures from trusted agents, the agent can consider this key valid and also sign it with its own key.

Key signing parties happen much more scarcely than the two other meeting cases. The scheduler selects a number of agents from the world and has them (fully) sign the key of every other agent involved in the signing party.

2.2.2 Agent selection

In order to model the difference in the behavior of real humans, who do not show the same sociability or eagerness to sign keys, a linear selection of the agents has to be avoided.

The preferred selection scheme has been to use a *roulette wheel selection*. In this scheme, every agent is given a variable probability to be selected. These probabilities are distributed among the agents in order to have some agents selected very often (those are modelling very active people) and some others selected rarely.

The name roulette wheel by itself comes from a common image of this selection scheme which consists of disposing all the agent on a wheel and give them a part of the wheel propotional to their porbability to be selected.

2.2.3 Introduction of fakes

When the list of agents is generated, some fakes are introduced. These are "pretending" to have the identity of another agent (this is done in our model by simply randomly choosing the identity of an already introduced real agent). In order to model more efficiently what's happening in the real world, every fake gets one signature from another real agent, which is considered to be the villain who forged the key and introduced it in the Web of Trust.

As the fake-keys bearing agents are introduced in the list as real agents, it may happen that they get selected for real life meetings, which *cannot* happen in real life. Thus a simple rule is applied in the model: if this fake-key-holding agent is involved in a real life meeting (or a signing party), it is simply discarded as this wouldn't have been possible in the Real World.

3 Validity of the model

3.1 Analysis of the strong set

To get relevant distances and node degrees we compared our results with real data only on the strong set, the strongly connected component, where each node is reachable by anyone else back and forth. We've found real data (see [2]) for some plots, among them the easiest to read and to

implement were the degree distribution, and the average path length distribution (which is also called *Mean Shortest Distance* or MSD distribution, on the website). It is important to remember that the graphs are directed ones.

3.1.1 Degree distribution

On the real data (fig. 1), the log log plot displays a line, which is characteristic to the power law of scale free networks. Our agent-built network should exhibit the same type of degree distribution (thanks to the kind of preferential growth it uses). It indeed does (fig. 2 on the following page). There is a cut off for the smallest degrees, but it may be due to the small number of nodes, and moreover because of the selecting roulette wheel. This wheel may not be selective enough.

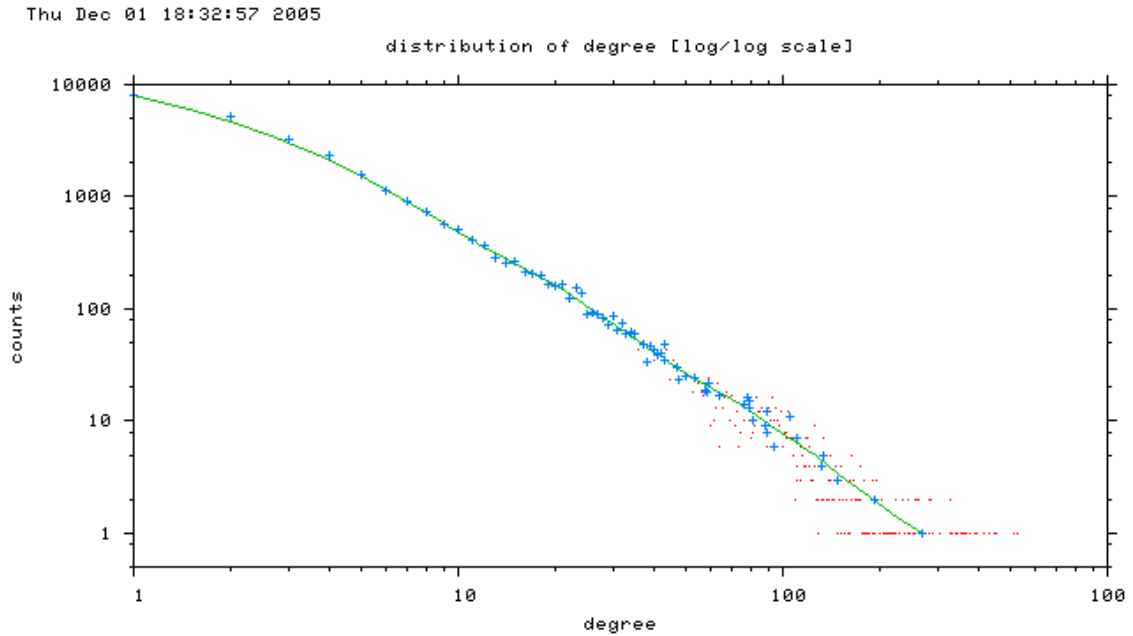


Figure 1: Degree distribution form the real strong set (30 000 nodes).

3.1.2 MSD distribution

As for the MSD distribution, its yearly evolution was also available (fig. 3 on the next page). Ours (see figure 4 on page 8) exhibits the same general aspect, but with a lower mean than the one of 5 for the real one (once again it is characteristic of scale free networks). Our mean is around 3.5. The reason for this is quite simple, we have not so many points, and only the strong set has been used to extract this information. And the most likely way for prudent agents to sign other keys is to have already signed the keys which have signed these other keys. And for a strict agent, the signing is done only in signing parties, where a cluster is formed. So the distance between members is rather short in the strongly connected component. That is why our model slightly differs from the reality. The qualitative behavior here is not in question, only the quantitative aspect of it.

3.2 The Leaf of Trust

The leaf of trust, introduced by Jörgen Cederlöf in [3] displays some interesting features of the Web of Trust. It is not the most useful tool to characterize a network, but it exhibits features such as how popular a key may be, and how connected to the world it is.

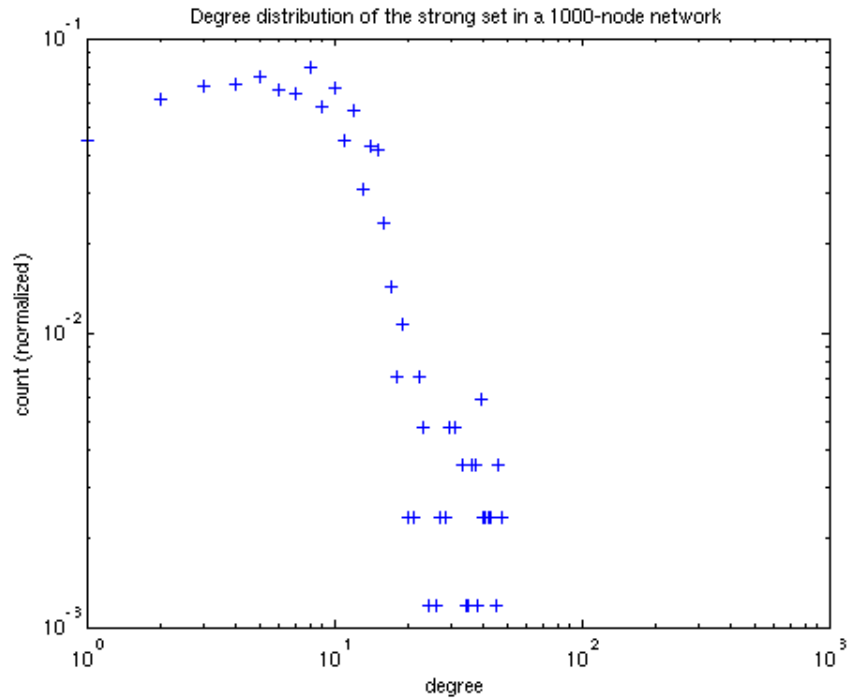


Figure 2: Degree distribution of the strong set of a simulated Web of Trust of 1000 nodes.

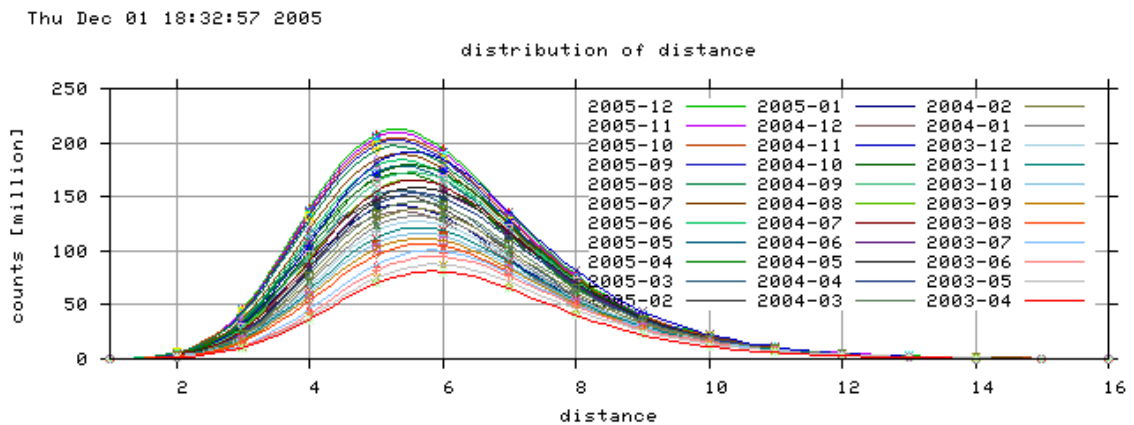


Figure 3: Evolution of the MSD distribution over the last years, centered around 6.

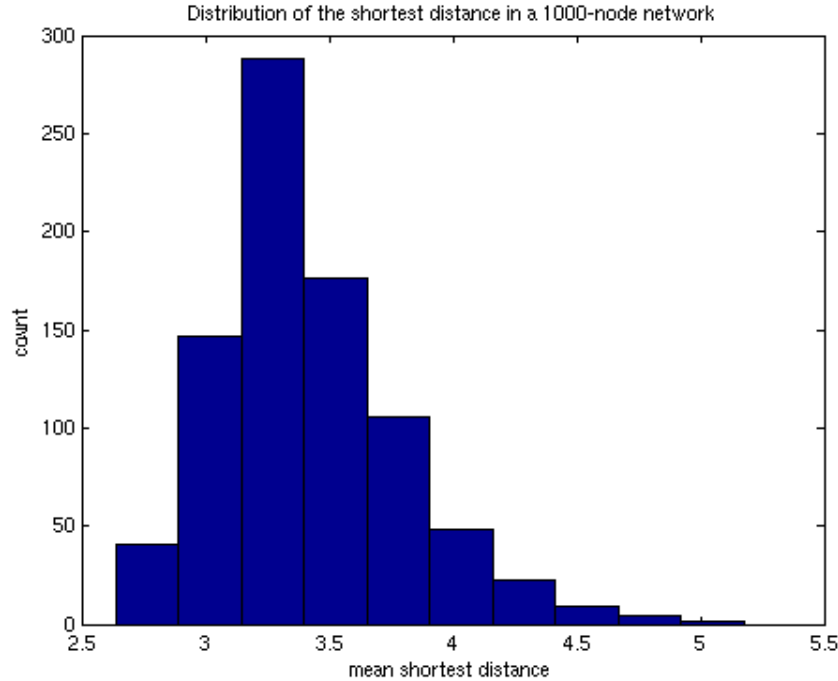


Figure 4: Our MSD distribution, for the strong set of a 1000-node network.

Figure 5 on the next page, is a matrix of the keys, sorted by ascendant MSD. One dot in the column x is a signature the key x has made on the key y . And the dots in row y are the keys x that have signed y . The colors of the dots correspond to level of trusts (these are not of direct interest to us as these are not implemented in our model). The upper-left corner is made of the most connected keys, closer to all the others. That's why the density is much higher here. And the edge of a leaf is drawn by the keys. To make a long story short, the keys signed on the left-lower edge of the leaf are at mean distance of 1 hop from the key which signed it, and from the key which is signed by it. There is one hop of difference between one edge dot and the diagonal either in the y axis and in the x axis to say it in another fashion. Jörgen Cederlöf explains this in details, and may be looked at if more explanations are needed, I hope however you won't need it. This edge is natural. If a key with MSD equal to x was signed by another one, this signing key would have a MSD of at most $x + 1$, and can be shorter thanks to its other links. So this fact enforce that no keys can be beyond the 1-hop line. A signature below it would mean that the signing key would have a MSD more than one hop below the signed key, which is not possible.

Now having a look at our leaf (figure 6 on page 10). The same one-hop line can be distinguished. It is bigger because of the closeness of the bigger part of the nodes, as already shown in the MSD distribution. And it is thinner in the lower-right corner because some are not connected to the biggest component, so there the one-hop line is very near the diagonal. The red points are the reflective signature, and the white one are the one-way ones. One can clearly see that only the white ones are upper the reflected one-hop line, which confirmed the explanation of the one-hop line, and the picture of the leaf. The big red rectangles are the 4 signing parties. each one is a cluster, and between them are often black lines. These empty areas can be explained as follows. Sometimes there is an agent connected to one node part of a cluster, and also connected to another cluster, and maybe only to the two of them, and therefore, even if it has not signed so many keys (almost empty, then black, column) and has not been signed so much (empty row), it has a pretty good MSD rank, better than the little cluster, but worse than the bigger one. It is very interesting

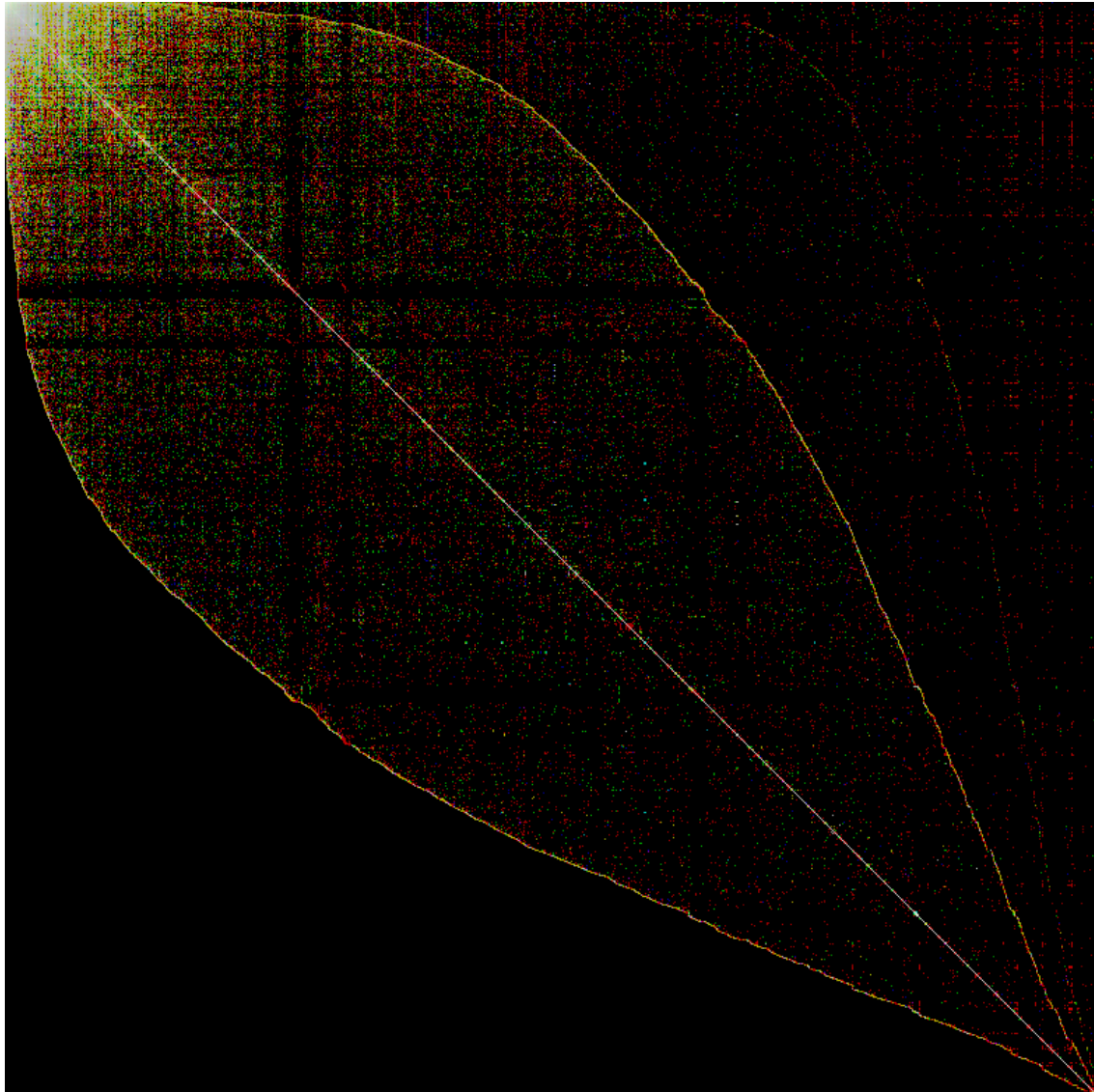


Figure 5: The complete Leaf of the real world Web of Trust (from [3]).

to note that one doesn't always have to sign or to be signed by so many keys, but has to choose well the connections to be reachable easily by the greatest number of people. In my opinion this can be a flaw of the system. Using only the key servers to know if one is at most at 3 hops to someone to trust him is an error. That's why it is recommended to only sign people in real life.

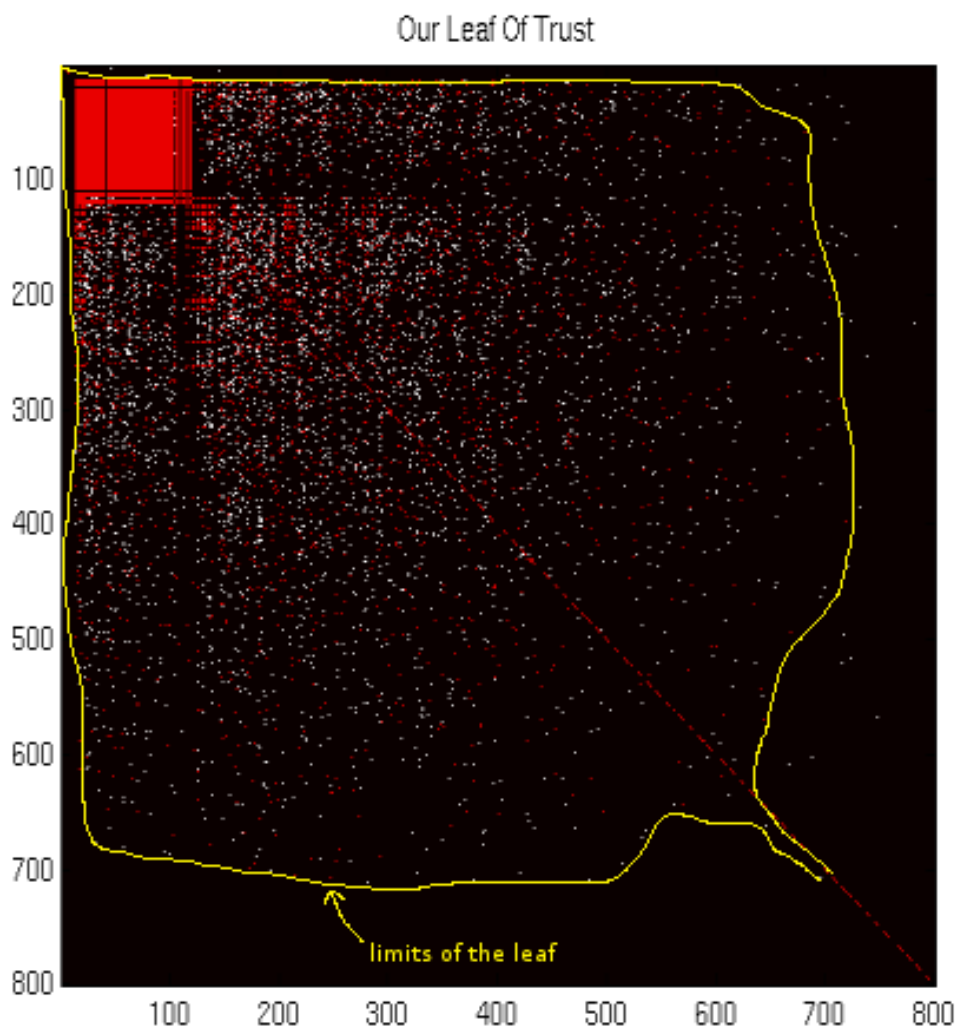


Figure 6: Leaf for a network of 800 nodes, and 4 signing parties.

One may also note that in the lines and columns where there is a cluster, the signatures spread in the y and x axis. It means that any cluster is also connected with others. To have a better look, one can refer to the zoomed picture of the 300 best nodes (fig. 7 on the next page). One will note that the best nodes are the ones connected to only a few clusters. That makes them appear as center of mass of the system, so they are like “bridges” between several “islands”. The only connections they have are above the diagonal, because of the one-hop line of course. It is important to note that a fake key couldn't reach these positions, since they can sign in signing parties, their identity being otherwise revealed.

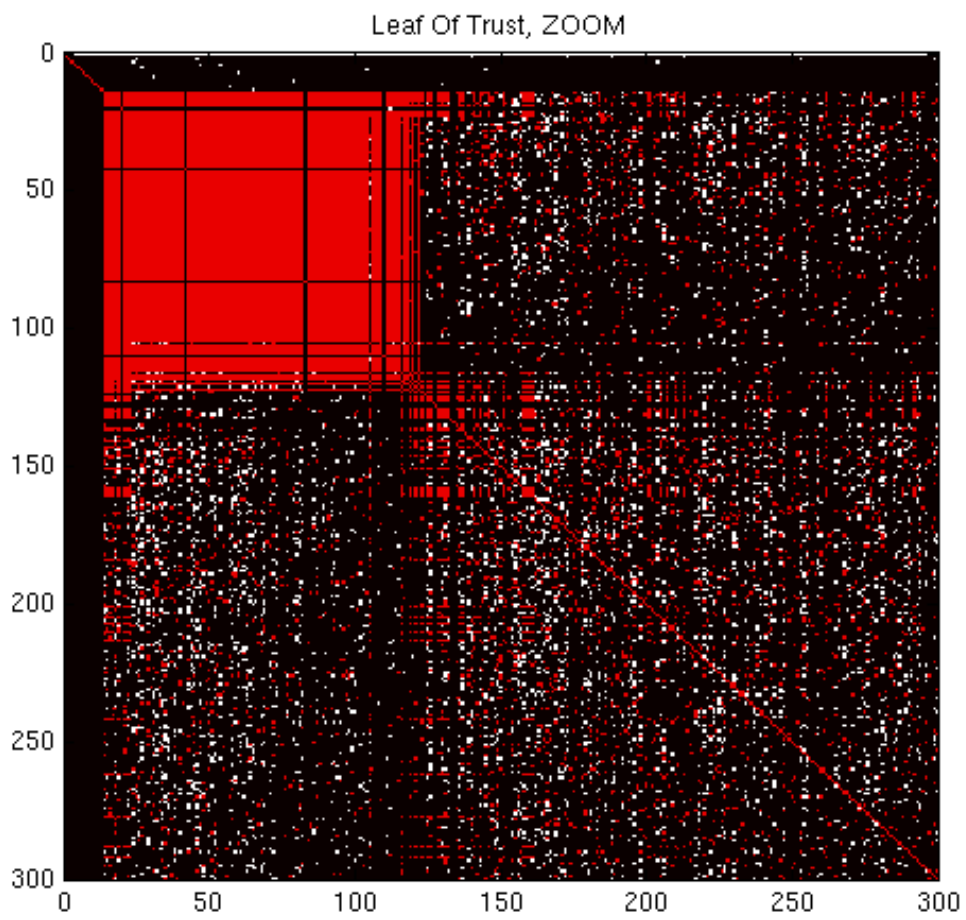


Figure 7: The 300 best (in MSD) keys in our Leaf of Trust.

Parameter	Value
Number of agents	100
Number of fakes	$n\text{Agents} \times \frac{1}{20}$
Size of the world	$n\text{Agents} + n\text{Fakes}$
Number of iterations	$\text{worldSize} \times 3000$
Number of signing parties	3
Probability of Laxist agents	$\frac{1}{3}$
Probability of Strict agents	$\frac{1}{3}$
Number of averaging runs	10

Table 1: The “default” values of the parameters of the model. It is important to note that some parameters depend on the others.

4 Results

4.1 Some small networks

The interesting results focus on the number of agents believing a fake key to be valid. This is determined by the number of signatures on the fake keys. However, it can be interesting to have a quick look at some actual networks to get a feeling of what the model builds. Some preliminary simulations were run with a reduced number of agents (13 agents, 2 fakes) for several repartitions of the agents (only prudents, only stricts, only laxists, a third of each). The networks built during these simulations are shown on figure 8 on the following page.

Nothing really unexpected is visible here. Laxist agents trust the fake keys as normal ones. Both the prudent and the “one third” networks show little trust in the fake keys. The strict agents, of course, totally distrust the fake keys (the two signatures appearing on the network are those of the strict agents which forged the fake keys).

4.2 Rumor spreading

In this model, an important number of parameters can be tweaked and it’s impossible to represent their influence all at once. In order to have significant and understandable results, the choice has been to have an initial set of parameters, it is shown on table 1. In the following, everyone of these parameters will be changed one at the time and its influence studied.

One can notice that some parameters in table 1 depend on others. This has been done in order to get rid of some side effects. For example increasing the number of agents would make the system react more slowly (as seen thereafter) so watching the result at a fixed point in time would make no sense as not only the number of agents would have changed, but also the watch-point in the relative evolution of the system. Having some parameters depending on others hopefully wipes out this inconvenience.

Except explicitly specified, when the term “relative average trust” is used in what follows, it means the average number of signatures on the fake keys divided by the total number of agents (real and fake ones) in the world. Only in section 4.2.3 on page 15 will this be different and rather refer to the average number of signatures on the fakes divided by *the average number of signatures on the real keys*.

4.2.1 Evolution in time

Figure 9 on page 14 shows the evolution with time of the rumor for several sizes of the world. For each of the 3 first graphs, the number of agents has been specified (and the number of fake keys computed following the relation shown in table 1). The last graph of the figure shows a comparison of the relative trust in the fake keys.

At first, the average trust in the fake keys grows quickly. The first signature made by the forger of the key and the ones made by the laxist agents contribute to the prudents also trusting

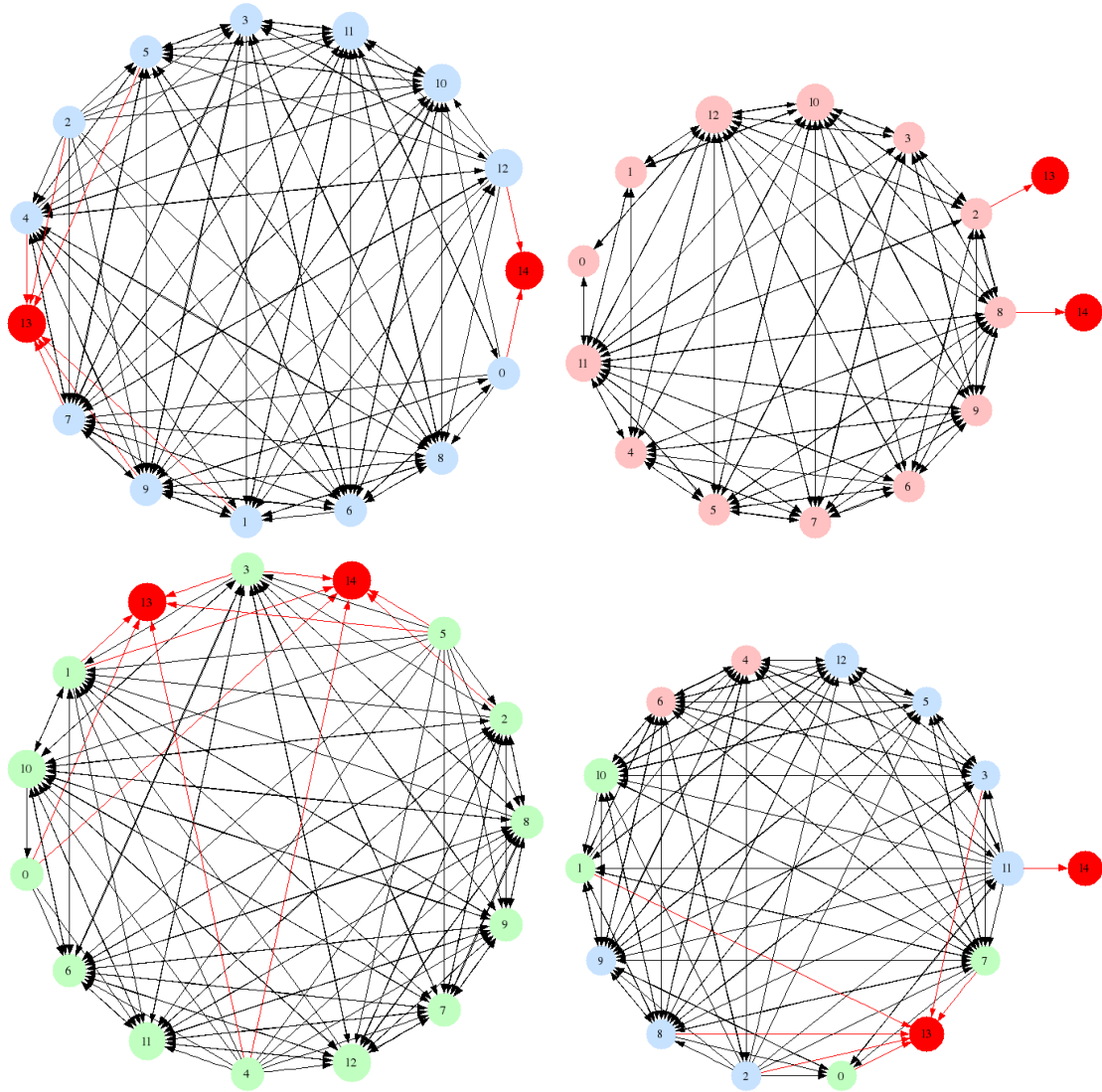


Figure 8: Some networks built using the model with varying proportions of each agent in the population. From left to right and top to bottom, the networks were respectively built with only prudent agents (blue), strict agents (pink), laxist agents (green) and a probability of $\frac{1}{3}$ each, fake keys are given a 100% red color. From the agents' signing rule, it is impossible for strict agents to sign fake keys. However the second graph clearly shows that two strict agents have signed fake keys. This is due to the fact that, when being introduced in the world, a fake key is signed by an agent supposedly being the one which forged it. In a world with only strict agents, the forger can only be one of them, hence his signature on a fake key.

this key.

However, the effect of real life meetings and signing parties is then more visible, as more and more agents which have met the real key-holders revoke their signature on the fake. It is interesting to notice that even if the general trend for the trust in fake keys is going down, it can locally go up again (for example in the first graph of figure 9 around timestep 20,000) as some agents keep on encountering the fake key without knowing the actual real key.

One can also note, on the fourth graph, the expected fact that a bigger world has more inertia than a smaller one, and the fake-key-trust decrease happens later.

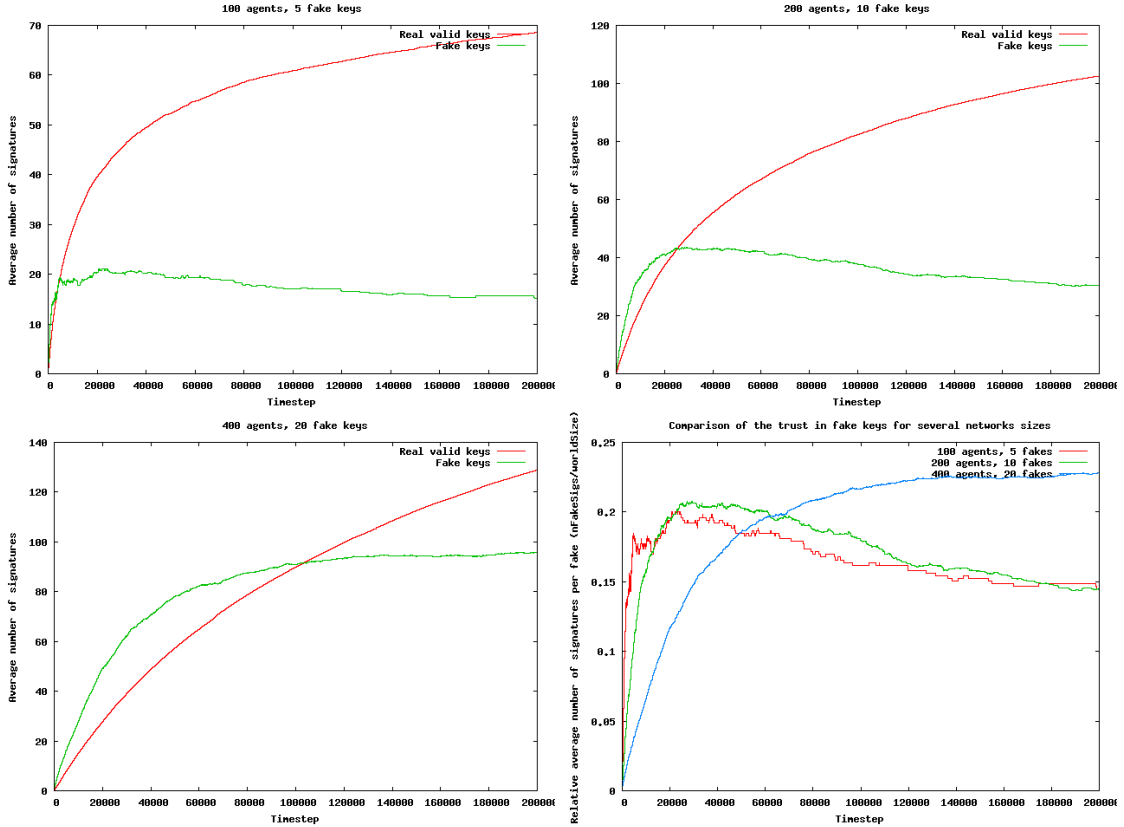


Figure 9: The average number of signatures on keys as a function of time for three initial number of agents. The first three plots compare the trust in valid keys (red) with the trust in fake keys (green) for respectively 100, 200 and 400 real agents. The last graph compares the evolution of the average number of signatures on the fake keys (divided by the number of agents to get comparable results) for the preceding cases (resp. red, green, blue).

4.2.2 Signing parties

As said in the introduction, signing parties are an important factor in the building of the Web of Trust as they create hopefully large fully connected components of fully trusted valid keys. Attention is now given to their actual influence.

First, the effect of one signing party by itself can be shown. Figure 10 on the next page shows the results of two interesting runs for very small world size (15 agents, 1 fake key). The first plot shows the usual evolution of the trust in fake keys, first increasing then slowly decreasing. The second plot, obtained in the very same conditions, shows the effect of a signing party, the fake key suddenly sees its trust going down to zero as the real agent's key as been involved in the signing

party.

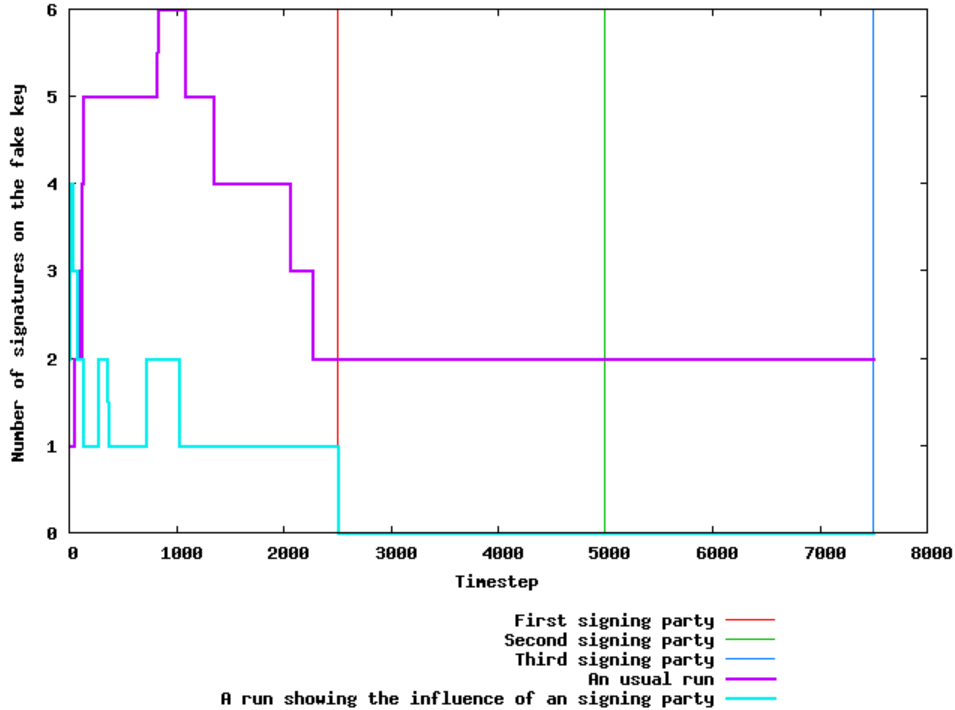


Figure 10: Two different simulations with the same parameters (15 agents, 1 fake). The second one clearly shows the influence of signing parties.

Getting back to the “usual” experimental parameters of table 1 on page 12, the number of signing parties is then gradually changed from 1 to 99 (keeping all other parameters unchanged). The results of this experiment are shown on figure 11 on the next page. Surprisingly enough, increasing the number of signing parties does not significantly reduce the trust in fake keys. This can however be explained by the fact that only a few agents (the agents with more probability to be selected) tend to attend signing parties, and these are almost the same ones over and over. This unexpected behavior may also exist in the Real World. Signing parties are often organized at the same time and place of other important meetings, in order to reach a maximum number of people. It is however notable that people attending these meetings are mostly always the same. Thus, if the first signing parties may be of crucial importance for a quick building of the network, the next ones may not add (or remove in the case of fake keys) much links in the Web of Trust network. This statement may however change if agents could be introduced at a latter time during the evolution of the model. This gives an opportunity for further enhancements of this implementation.

4.2.3 Number and repartition of agents

The influence of the size of the network, both considering the number of real agents and the number of fakes, can also be studied.

First, the number of agents is gradually changed (all other parameters, particularly the number of introduced fake keys, being set as stated in table 1). No actual influence of the number of agents in the world can be seen on the average trust in fake keys which is constant whatever the number of agents.

Fixing the number of agents back to 100, the influence of the number of fake keys, varying from 0 to 100 (thus a world size varying from 100 to 200), has then been studied. As the last

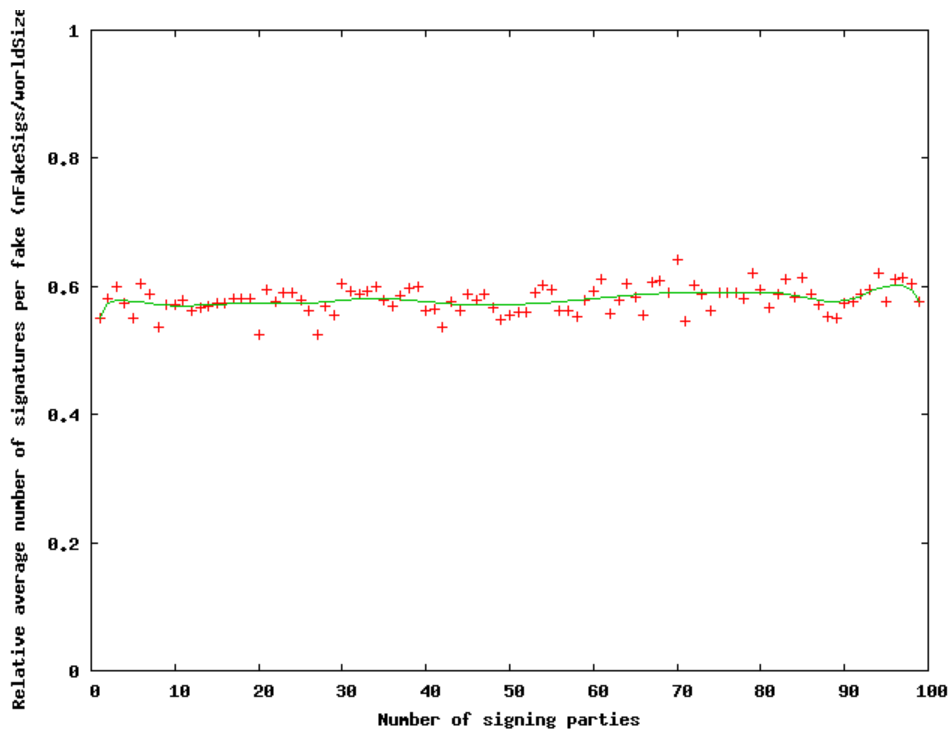


Figure 11: The influence of the signing parties on the trust given to fake keys.

result would have let it be expected, no particular effect of the number of fake keys on the relative trust can be seen either.

The composition of the population can also be changed. All the above plots have been built from simulation with equal probabilities ($\frac{1}{3}$) for each agent to be either prudent, strict or laxist. Figure 12 on the following page shows the effect of varying the proportion of strict and laxist agents.³

As expected, the number of strict agents has a drastic influence on the rumor (fig. 12 on the next page, left). This is easily understood by the fact that they only sign keys in signing parties, which prevent them from signing fake keys. Moreover, with less signatures on the fake keys, the prudent agents have less (incorrect) information which may result in their signing an invalid key.

A much more unexpected result is the steadiness of the average number of signatures on the fakes when the number of laxist agents increases (fig. 12 on the following page, right). Maybe this can prove that the Web of Trust is quite robust to rumors, even if most people are rumor-spreaders themselves. On the long run, signing parties involving laxists gradually make them revoking their signatures on the fake keys, which may be an explanation for this stability. However, the relative size of the model compared to the actual size of the Real World network may have a non-negligible importance here. Some simulations with much larger sizes of the model should be run in order to answer this question. This was unfortunately impossible to achieve in reasonable time with the current implementation of the model.

³When one proportion is varied, the other one is kept to $\frac{1}{3}$. For example, when the number of strict agents is increased, the number of laxist agents does not change, and only the number of prudent agents (which are considered the “basic” agents) is decreased.

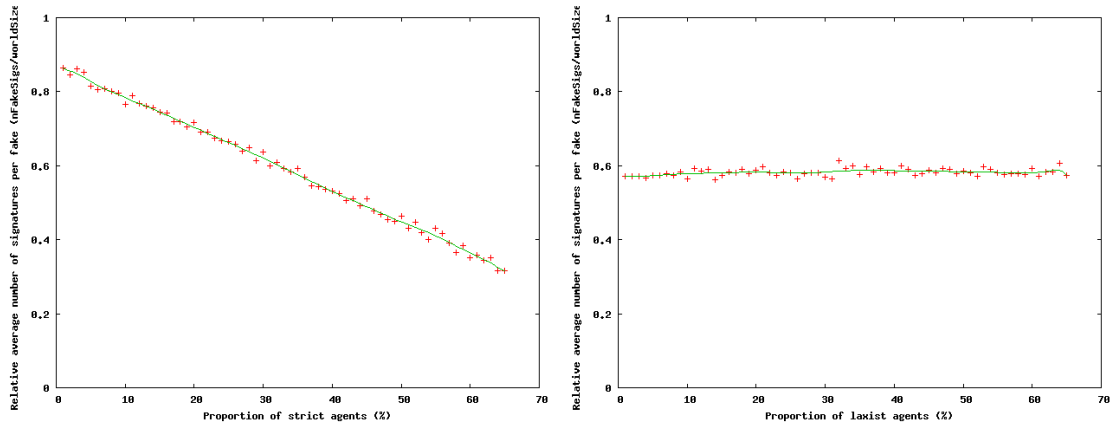


Figure 12: The more strict agents there is in the world, the less fake keys are actually trusted (leftmost graph), however, changing the number of laxist agents in the world has no influence on the average trust in fake keys (rightmost plot).

5 Further enhancements and conclusion

Some further enhancements can still be added to the model. For example, in all the simulations, fake keys were introduced before the network was built. It may be interesting to see how the Web of Trust evolves if keys (both real and fake ones) are introduced gradually in the world. It would also be interesting to run simulations with much more agents than the few hundreds used here, it has not been possible to do so due to the computation time needed.

Moreover, in the Real World, forging only one key is not an “efficient” method to have this key trusted as valid. Forgers usually make up false keys for several identities (either real person or made up ones) at a time, and have them sign each other so that they look more authentic. This behavior has not been implemented in our model, but it might be interesting to see how much more robust this practice may be compared to only introducing single, unrelated, fake keys.

In order to keep the model simple, the trust in the validity of a key and the trust in its owner’s behavior have been merged into a single parameter. A further step towards a more realistic model would be to make a difference between those two trust levels so that, for example, if an agent is known to be laxist, not much trust will be given to its signature on other keys, even if its own key is trusted to be authentic.

Finally, from the validity tests and the results which mostly show the expected trends, we can infer that the model we built for the Web of Trust has been correctly implemented and simulated. We can also state, from the results of our simulations, that the Web of Trust seems to be a trustable method to ensure the authenticity of public keys in the Real World and contain the spread of fake keys to a relatively low level.

References

- [1] Public-key cryptography, http://en.wikipedia.org/wiki/Asymmetric_key_encryption. *Wikipedia*.
- [2] Henk P. Penning. Analysis of the strong set in the pgp web of trust, <http://www.cs.uu.nl/people/henk/henkpgp/pathfinder/plot/>.
- [3] Jörgen Cederlöf. Dissecting the leaf of trust, <http://www.lysator.liu.se/~jc/wotsap/leafoftrust.html>.