Key Infection: Smart Trust for Smart Dust

Ross Anderson
University of Cambridge
Ross.Anderson@cl.cam.ac.uk

Haowen Chan
Carnegie Mellon University
haowenchan@cmu.edu

Adrian Perrig
Carnegie Mellon University
perrig@cmu.edu

Abstract

Future distributed systems may include large self-organizing networks of locally communicating sensor nodes, any small number of which may be subverted by an adversary. Providing security for these sensor networks is important, but the problem is complicated by the fact that managing cryptographic key material is hard: low-cost nodes are neither tamper-proof nor capable of performing public key cryptography efficiently.

In this paper, we show how the key distribution problem can be dealt with in environments with a partially present, passive adversary: a node wishing to communicate securely with other nodes simply generates a symmetric key and sends it in the clear to its neighbours. Despite the apparent insecurity of this primitive, we can use mechanisms for key updating, multipath secrecy amplification and multihop key propagation to build up extremely resilient trust networks where at most a fixed proportion of communications links can be eavesdropped. We discuss applications in which this assumption is sensible.

Many systems must perforce cope with principals who are authenticated weakly, if at all; the resulting issues have often been left in the 'too hard' tray. One particular interest of sensor networks is that they present a sufficiently compact and tractable version of this problem. We can perform quantitative analyses and simulations of alternative strategies, some of which we present here. We also hope that this paper may start to challenge the common belief that authentication is substantially about bootstrapping trust. We argue that, in distributed systems where the opponent can subvert any small proportion of nodes, it is more economic to invest in resilience than in bootstrapping.

1. Introduction

Wireless sensor networks are becoming increasingly important for a wide variety of applications such as factory instrumentation, climate control, environmental monitoring, and building safety. As sensor networks become cheaper and more commoditised, they will become attractive to home users and small businesses, and for other new applications.

A typical sensor network consists of a large number of small, low-cost nodes that use wireless peer-to-peer communication to form a self-organized network. They use multi-hop routing algorithms based on dynamic network and resource discovery protocols. To keep costs down and to deal with limited battery energy, nodes have fairly minimal computation, communication, and storage resources. They do not have tamper-proof hardware. We can thus expect that some small fraction of nodes in a network may be compromised by an adversary over time.

An interesting example of a sensor network technology is given by the 'Smart Dust' project which is developing tiny sensors [9]. Its goal is to make sensors so small and cheap that they can be distributed in large numbers over an area by random scattering.

The security of sensor networks may be important, especially if they are deployed in military applications or in safety-critical applications such as medical monitoring. In addition to physical destruction or barrage jamming, a range of more subtle attacks may be attempted by a capable, motivated opponent. The opponent may be passive, simply monitoring the sensor data flows in order to determine the extent and capability of the network; she may be active, and transmit deceptive messages; she may try selective jamming or network flooding; and she may subvert a number of the nodes and use them for various active and Byzantine attacks. The results of such attacks may include the loss of personal privacy, the loss of service of critical sensor systems, and the consequences of people or equipment taking erroneous action based on maliciously falsified sensor data.

When designing security for sensor networks, an important problem is *key distribution* – the problem of establishing shared secret keys between sensor nodes. Efficient key distribution for sensor networks is still an important research area. Devices with capable processors can use well-established asymmetric cryptographic techniques, based on Diffie-Hellman key agreement or RSA-based key establishment [4, 15]. However, the cheap processors used in sensor nodes often lack the resources to perform asymmetric cryptography. These devices are expected to cost at most a few tens of cents, so asymmetric cryptography may often require too much computation or too much memory. Even where it is feasible, the fundamental limit on sensor node performance is battery energy; each available microjoule must be carefully apportioned between sensing, computation and communication. Digital signatures might thus be used by an attacker to perform battery-draining denial-of-service attacks.

In response to this problem, key-distribution schemes based on symmetric cryptography have been proposed [6, 3]. Typical schemes rely on having each node pre-load a large number of keys to have a reasonable probability of sharing one with a neighbour. However, such schemes still require a large amount of memory, as well as an infrastructure to load the keys into the sensor nodes, which may be too cumbersome for some applications. Other schemes set up pairwise keys by physical contact between devices; for example, a number of burglar or fire alarm sensors are each touched briefly against a controller before emplacement [17]. Since each node must be physically brought into contact with the controller, this may limit large scale sensor deployments.

In this paper, we will focus on key distribution in commodity sensor networks where we do not assume a *global passive adversary*. Previous work on key distribution for sensor networks has assumed a strong attacker model: the adversary is assumed to be present both before and after node deployment; she can monitor all communications everywhere in the deployment site at all times. It is usual to assume also that she can subvert (maliciously reprogram) a small number of targeted sensor nodes.

Such assumptions may be appropriate for strategic sensor-network applications such as nuclear test-ban treaty monitoring, where an international agency emplaces sensing devices on the territory of a suspect state. However, they have led to the development of heavyweight security protocols that incur costly overheads of computation, of storage, or of time and effort needed for pre-deployment configuration. This overhead is almost certainly excessive for more mundane applications. In fact, for many applications, if 'security' increases deployment costs or impairs ease of use, then many users will simply switch it off. For security to be widely used, it should ship as a default and require no significant effort by even non-technical users to configure. For example, many early WiFi base stations shipped with encryption switched off by default, and many users simply did not turn it on.

We are therefore interested in developing usable security, and, when exploring the available trade-offs, one possible target is the assumption of a global passive adversary. In the world of wireless sensor networks, such an adversary is often not realistic. In many applications, we will be able to assume that she can monitor only some small proportion of the traffic during initial deployment.

We will discuss a number of scenarios in detail in section III; as an example meanwhile, consider a burglar alarm sensor network deployed in an office building containing a few hundred nodes, each with an effective radio range of 10 metres. If the opponent has already installed so many surveillance devices in the building that she can monitor all radio exchanges between nodes, then the building owner is trying to solve the wrong problem! As another example, consider a canister of 10,000 smart dust motes deployed via an artillery shell into enemy territory, and which then communicate with each other using low-power radio on a single channel. Each enemy sensor will only be able to pick up the nearest signal; distant signals will interfere with each other. Thus, if our 'white dust' is deployed at a much greater density than the enemy's 'black dust' defensive motes, most of our communications will go unmonitored – at least initially. (Of course, once the presence of white dust has been reported, several canisters of black dust may be deployed to reinforce the defenders, so that almost all white communications are monitored thereafter; but during initial deployment, the white motes have the advantage of surprise.)

It therefore appears of interest to see whether weakening our threat model – a small proportion of communications during the network deployment phase – and exploring whether this can give us useful cost savings and usability gains.

Using our more realistic attacker model, we design a lightweight security protocol suitable for use in non-critical commodity sensor networks. Our protocol is called *Key Infection* and is based on the assumption that, during the network deployment phase, the attacker can monitor only a fixed percentage α of communication channels. (We provide a detailed justification for this assumption in Section 3.) We show that, using this slightly relaxed attacker model, it is possible to perform sufficiently secure key distribution with low computation overhead (a few symmetric cryptographic operations), no memory overhead (only storage for the actual keys used in node-to-node communications), and no prior key setup. Due to the lightweight nature of the Key Infection protocol, it is highly suited for implementation in low-cost commodity sensor nodes.

This paper makes several contributions. First, we identify a more realistic attacker model that is applicable to non-critical commodity sensor networks. Second, we present *Key Infection*, a light-weight key-distribution mechanism that is so efficient that it is applicable even to smart dust sensor nodes. Third, we analyze the security of key infection, and design *Secrecy Amplification*, an additional mechanism to strengthen the security of key infection in the presence of an active attacker. Together, these provide interesting new ways to trade off security for cost and usability. Finally, the research community has so far largely considered authentication to be a bootstrapping problem. However, in many real-world applications, the major cost is in maintenance, rather than initial deployment. We hope this paper will start to shift the focus toward designing authentication mechanisms that are optimised over the total lifecycle of a network.

2. Previous Work

A sensor network is typically set up as follows. When a node hits the ground, it broadcasts its identity, say i. If a neighbouring node j hears it, it replies. The two mutually-aware nodes now set RF power at just the level needed for communication. A source-based routing protocol, based on the periodic broadcast of beacons by base stations, organizes the nodes into forests, with a base station at the root of each tree. This involves not just routing but time synchronization: to save power, the nodes typically turn off their communications, only waking up and listening for radio signals intermittently. The base station may be a normal node, or it may have a larger battery so it can cope with a larger volume of traffic and also communicate with the outside world by transmitting at a higher power.

The routing architecture depends on the nodes behaving themselves – that is, executing the software with which they were loaded before deployment – and on the integrity of the link traffic between them. If an opponent can intercept and modify this traffic, then he can create disruptions; a typical attack will target the routing mechanism so as to introduce loops or partition the network. It is also possible to disrupt the network by maliciously introducing clock skew. Previous work therefore included services such as authenticated broadcast. These rely on shared symmetric keys. The initial keys are diversified from master keys, known to the base stations: thus we may have $k_{ij} = \{i, j\}_{KM}$. The base stations act as key-distribution centers in much the same way as with Kerberos, but using protocols designed to minimize communications overhead [14].

While adequate for many environments, this security architecture is vulnerable to node destruction and subversion. As the base stations send significantly more radio traffic than the other nodes, an opponent can use direction finding to locate them, then either destroy them or subvert them (for example, by probing them to extract cryptographic keys or to load maliciously altered software). The compromise of master keys betrays all traffic that was protected by them and that was also overheard and stored by the opponent.

There are various possible countermeasures. In some applications, one can use normal nodes as base stations and have other nodes replace them after random periods of time. Another is for the first generation of base stations to possess master keys that are destroyed once a network has been established and link keys have been set up between neighbouring nodes. However, there are applications where this may be unsatisfactory. The deletion of all master keys effectively closes the network and makes the addition of new nodes impossible. This makes it difficult to expand a sensor network or to add new nodes to replace failed or battery-exhausted nodes. Also, a number of nodes fail when they hit the ground, and then be probed by the attacker. So key erasure cannot be assured.

An alternative method has been proposed by Eschenauer and Gligor [6], and extended by Chan, Perrig, and Song [3]. In these key-distribution schemes, enough symmetric keys are pre-loaded on each node that any two nodes will probably share a key after deployment. However, these schemes require a significant pre-computation phase in which the shared keys are generated based on the total number of nodes in the network and the expected density of deployment. There may therefore be significant usability issues. Furthermore, each node requires a lot of memory to store keys, most of which is effectively wasted since only a small fraction of keys are ever actually used. So there are cost issues too. It appears that random key pre-distribution is not a practical solution for low-cost commodity networks. Recently, Du et al. [5], and Liu and Ning [11, 12] leverage Blom's [1] and the Blundo et al. [2] key establishment protocols to achieve high resilience to node capture. Zhu et al. [19] assume that an attacker arrives after key establishment, and that all nodes in the network share a secret key. However, all these protocols assume secret information is set up before sensor network deployment. In this paper we explore key setup without any prior information.

3. A Real World Attacker Model

In prior work, researchers have assumed highly capable and motivated attacker – a global passive adversary, who monitors and stores all communications. It is also commonly assumed that the attacker can subvert a small chosen subset of the sensor nodes, and deploy hostile nodes of her own. She is sometimes assumed to be a global active adversary, in that she can modify and inject communications at will at any time.

This assumption is inherited from the world of crypto research, which in turn was conditioned by the experience of World War 2 where both strategic and tactical communications were mostly carried by radio and were routinely intercepted; and the world of international telephony in the post-war years where the traffic volumes were such that widespread interception by signals intelligence agencies was feasible and was indeed carried out. However, in the modern wired world, it is less realistic because of the cost of tapping high-speed backbones and because of today's huge traffic volumes. Intelligence agencies have had to sponsor legislation in many countries to install surveillance devices to get access to traffic near its end-points. In many countries, the law not only compels communications service providers (such as phone companies and ISPs) to provide such facilities, but regulates the proportion of subscribers which the provider must be able to wiretap simultaneously. Thus, even if a new application (such as a peer-to-peer network) is being deployed on a global scale on home PCs using unencrypted TCP/IP communications, it may often be a realistic assumption that no single potential opponent has had the capability to record more than (say) 1% of the initialisation traffic. Of course, if the network is later deemed to be subversive, its users may be targeted intensively. However, during an initial innocuous deployment, it is reasonable to assume that most of the traffic escapes unmonitored.

Consider now the case mentioned above, of a tactical deployment of 10,000 smart dust motes air-dropped into enemy territory. Assume an initialisation process as the shell is armed, whereby a master key KM is generated and transmitted to all the motes. On landing, mote i find that mote j is a neighbour, and uses KM to generate a session key $\{i, j\}_{KM}$, which they subsequently use. However, some of the motes are broken on impact, and since it is not economic to make them tamper-proof, the enemy can probe out KM. The enemy thus has access to all the initialisation traffic that it recorded. Is this a disaster? Probably not, as we shall examine in detail below. Unless the defender was expecting the motes to be deployed at that location – and therefore had her own defensive motes ready and waiting to record communications or insert themselves into the sensor network – it would have been impractical for her to have recorded more than a tiny fraction of the initialisation traffic. So in this case the master key did not achieve anything beyond securing the secrecy of the small fraction of communications that was actually recorded by the adversary. If this amount of communication information was too little to be of significant use to her, then we could have got the same result at lower cost by not using a master key at all, but simply exchanging session keys in the clear.

Consider now non-critical commodity sensor networks, which for cost reasons have extreme limitations on sensor hardware and also require that the pre-deployment setup must be minimal. These have a quite different threat model than tactical military networks, simply because they are less valuable as targets and little damage is done to the user if their security should fail. Hence, it is rather dubious to apply a stronger attack model to non-critical commodity sensor networks than we would apply to a tactical military deployment.

As an analogy, doors to bank vaults and doors to homes have very different attack models, and this is reflected in their construction – each should provide an optimal trade-off of security and cost for its owner. Homeowners understand that normal doors are not invulnerable to a determined adversary, but prefer to save money (and enjoy more elegant and convenient access to their homes) by assuming that nobody will attack their front door with an acetylene torch or a rocket-propelled grenade. Likewise, for commodity sensor networks, the attacker model should reflect realistic protection requirements.

Our proposed trade-off depends on only a slight weakening of the attack model – that hostile surveillance is not ubiquitous during the deployment phase of the network. This phase, the time while the nodes are doing key exchange, may last only several seconds – a very small fraction of the network's lifetime. From the attacker's viewpoint, it would be extremely expensive to deploy surveillance devices against a large number of offices, factories and other targets, and retain them in place for a period of perhaps many years, in order to capture key material. This would be especially difficult as the main obstacle to surveillance is the availability of power. Long-term surveillance in general requires either connection to mains electricity, or periodic redeployment of battery-operated devices.

Thus it is unlikely to be economical for anyone to attack commodity sensor networks by means of universal surveillance directed against premises in general. (Targeted surveillance does get attempted against high-value targets such as defence ministries and the residences of heads of government. In case their usual countersurveillance measures fail, owners of such premises are advised to use wired networks, or to invest in careful cryptographic initialisation of each node. But theirs is a tiny market and their problems are not our subject here.)

The attacker model we assume is therefore as follows:

1. The attacker does not have physical access to the deployment site during the deployment phase;
2. The attacker is able to monitor only a small proportion (α) of the communications of the sensor network during the deployment phase. After key exchange is complete, she is able to monitor all communications at will;
3. The attacker is unable to execute active attacks (such as jamming or flooding) during the deployment phase. After key exchange is complete, she is free to launch any kind of attack.

In summary, the attacker is assumed to be fully capable at all times except during the deployment phase, where she is assumed to have at most a partial, passive presence. This is realistic because deployment represents a very small window of opportunity for an adversary. The possibility of an attack during the vulnerable window is usually an acceptable risk since the window is extremely small (several seconds) compared to the overall lifetime of the network (up to several years).

In order to violate the assumed attacker model, an adversary has to achieve several things. First, she has to have the foresight to deploy surveillance equipment or adversarial nodes at the target site before the sensor network is deployed there. Second, her eavesdropping devices must remain in place, operational and undetected, until the sensors perform key exchange. Third, she needs to be able to identify, retrieve and process the the relevant eavesdropped product in order to extract the key exchange messages. In general, except for high-value strategic targets, these requirements will make it too expensive to maintain anticipatory surveillance – against which such targets are vigorously defended in any case.

When we come to non-critical commodity sensors such as light and temperature sensors for homes, anticipatory surveillance makes no sense. If an attacker wishes to violate the privacy of a home by deploying sensors in it, she will presumably harvest audio directly, instead of performing sophisticated electromagnetic eavesdropping to steal the keys of sensor nodes that just conceivably might be deployed some time in the unspecified future.

Furthermore, since the deployment time window is small, additional security measures could be taken by network owners whose threat model lies somewhere between the random householder's and the Head of Government's. For example, the site could be secured physically and monitored against adversarial entry; it could be swept for eavesdropping or adversarial devices prior to key exchange; or an intrusion detection system could be run after key exchange to ensure that no external nodes have managed to insert themselves into the network. These additional measures are simple to execute and relate well to users' existing intuitions of security.

Our attack model is in fact probably still too paranoid. However, by using this slightly weakened model, we can greatly simplify key exchange, as we shall describe in the following section.

4. Key Infection

In view of our real-world attacker model, let us consider the simplest possible approach to key establishment: each node simply chooses a key and broadcasts it in plaintext to its neighbours. Thus, node i, when it comes to rest, broadcasts a key k_i. Recall that this is a short-range transmission, with a maximum range of perhaps ten meters, and perhaps half a dozen other nodes will have landed within range. As they become active, they detect each others' presence and start organizing themselves into a network. The idea is to propagate key material as contact is made, rather like an infection spreading through a biological population.

Assume that node i's signal is heard by node j. Its response will be to generate a pairwise key k_j and send it, along with its name, to i: $\{j, k_{ji}\}_{ki}$. As we try at all times to minimize the energy cost of our protocols, this packet will be transmitted using the minimum power necessary for the link, based on measurement of the strength of the signal from i.

The key k_{ji} can be used to protect traffic between i and j. It may seem extremely counterintuitive that plaintext key broadcast can give any protection at all. However, in an area with no opponents, plaintext key exchange foils any adversary who arrives later. Even where there are opponents already present at the time of deployment, it will still give significant protection. For example, we show below that where there is one 'black' (hostile) smart dust sensor node for every 100 white nodes, and each node has on average four neighbours within range, only 2.4% of links will be compromised.

The situation can be improved considerably by a small change in the protocol. Instead of each white node broadcasting a single initial key as loudly as it can, it starts off transmitting very quietly and steadily increases the power until a response is heard. A link key is established with the responder, and then the broadcast resumes with a new initial key. This 'key whispering' protocol ensures that two white nodes W_1 and W_2 within range of each other and of a black node B will exchange a secure key provided that the black node is further away from either W_1 or W_2 than the distance between W_1 and W_2. In this case, the number of links that the black node can eavesdrop falls to 0.8%. We will now describe the simulation in more detail.

5. Analysis

Key infection is trivially secure if the attacker arrives *after* the key infection phase. In this section, we consider the case where an attacker already has some black dust nodes installed *before* our customer installs the white dust nodes. We compute an upper bound on the ratio of communication links that the black dust nodes may compromise. Assume the maximum range of the radio is R. Let the smart dust nodes be distributed over an area of size s, let N_b be the number of black dust nodes, and let N_w be the number white dust nodes. Assume the black nodes are distributed uniformly at random in the area of size s.

For now, we only consider the case where the black nodes collude. So in order to eavesdrop a key setup between two white nodes, the attacker needs at least one black node in the radius of each one of the two white nodes. (In the non-colluding case, we do not consider the sharing of information among black nodes, so the outcome is always more favourable for white.)

If both nodes transmit at maximum strength, then given a link e, the effective eavesdropping area is at most πR^2, and hence the probability that this link is bad, i.e., can be eavesdropped by at least a black node, is at most $\pi R^2 N_b/s$.

For the whispering case, if a link has length r, then both nodes will transmit their signals at strength that exactly reaches distance r. Therefore, a black node has to lie in the intersection of the two circles of radius r where the distance between the two centers of the two circles is r. The effective eavesdropping area is thus at most the area of this intersection which is $2r^2(\frac{\pi}{3} - \frac{\sqrt{3}}{4}) \doteq 1.2r^2$, and the probability that the link is compromised is at most $1.2r^2 N_b/s$.

To evaluate our protocols, we simulated the key infection of a random and uniformly distributed white dust, contaminated with 1%, 2% or 3% of black dust, and averaged the results over 100 simulations. We simulated $10,000$ white dust nodes, with 100 eavesdropping black sensor nodes. We assumed that both the white dust node and the black dust node have a receiver range of 10 meters. We considered various node densities d, which we characterize by the average number of neighbour nodes.

We first compared the point-to-point key exchange and whisper mode extension. Table 1 lists the percentage of compromised links for the basic neighbour infection protocol, and the whisper mode extension. We found that the whisper mode extension results in approximately three times fewer compromised links.

In the above, we assumed that the black nodes have the same receiver sensitivity as the white nodes, which appears reasonable given the economies of scale of single-chip receiver technology. It follows that they would have larger batteries – or a wired network – so they can transmit farther. This seems a reasonable strategy for a pre-emplaced defensive network. However, what the above simulation indicates is that the combatant who can produce the smaller, denser dust has a significant advantage.

6. Multihop and Multipath Key Establishment

As the previous section shows, an attacker who introduces black dust nodes before we deploy our white dust nodes can subvert some fraction of communication links. In this section, we design and analyze *Secrecy Amplification*, a technique that utilizes multipath key establishment to make her job significantly harder. We also simulate various strategies for key establishment in which nodes introduce each other, in an attempt to bypass nodes whose links are compromised from the start (by being too close to black nodes) or which are subverted later.

6.1. Secrecy amplification

The first strategy is 'secrecy amplification' in which we combine keys propagated along different paths. Suppose that the nodes W_1, W_2, and W_3 are neighbours. W_1 and W_2 set up the key k_{12}, W_1 and W_3 the key k_{13}, W_2 and W_3 the key k_{23}. To amplify the secrecy of key k_{12}, W_1 can ask W_3 to exchange an additional key with W_2 (here N_1 is a unpredictable nonce generated by W_1, N_2 is a unique nonce generated by W_2 (used for confirmation of key k'_{12}), and we assume that principals can distinguish names, nonces and keys).

$$\begin{array}{ll}
W_1 \rightarrow W_3: & \{W_1, W_2, N_1\}_{k_{13}} \\
W_3 \rightarrow W_2: & \{W_1, W_2, N_1\}_{k_{23}} \\
W_2 \text{ computes}: & k'_{12} = H(k_{12} \,||\, N_1) \\
W_2 \rightarrow W_1: & \{N_1, N_2\}_{k'_{12}} \\
W_1 \rightarrow W_2: & \{N_2\}_{k'_{12}}
\end{array}$$

After this protocol terminates, W_1 and W_2 update their key k_{12} by hashing it with the value just received: $k'_{12} = H(k_{12} \,||\, N_1)$. If k_{12} was secure before the protocol, the new k'_{12} will also be secure afterwards. But if the initial link key k_{12} was compromised, the new one k'_{12} will not be, so long as neither k_{13} nor k_{23} is. The last two messages of the protocol are needed for key confirmation, to ensure to W_1 and W_2 that the other party correctly received the key.

Tables 2 and 3 show the results of our experiments. We simulate the regular neighbour key infection and compare it to the secrecy amplification. The tables list the ratio of compromised links for a varying density α of black dust: 1%, 2%, and 3%. Table 3 also uses the whispering mode for neighbour key infection.

So naive three-party secrecy amplification gives an improvement of about 20%. Is this worthwhile? Well, it is often almost free. The reason for this is that nodes which use optical communications – tiny lasers and MEMS corner reflectors – have mostly unidirectional links, and so need routing algorithms for finding a loop back to a transmitting node in any case.

Secrecy amplification is not limited to paths of two hops. For example, to protect against black dust B between the two white dust nodes W_1 and W_2, one can recruit a chain of white nodes W_3, W_4, W_5 that are out of range of B. The source routing algorithms used in many sensor networks give partial location information, and can probably be improved by using link transmitter power as a distance metric. They can also be easily adapted to search for 'a path from A to B that does not pass through C, D or E'. Thus, even when B is directly between W_1 and W_2, there is some hope of finding a chain W_3, W_4, W_5 that enables W_1 to set up a secure key with W_2. The next table has simulations of secrecy amplification undertaken using a multihop return path. This is sig-

	α = 1%		α = 2%		α = 3%	
d	basic	whisper	basic	whisper	basic	whisper
2	1.13%	0.40%	2.34%	0.81%	3.48%	1.19%
3	1.75%	0.61%	3.51%	1.25%	5.06%	1.81%
4	2.38%	0.83%	4.61%	1.61%	6.75%	2.44%
5	2.92%	1.01%	5.76%	2.00%	8.40%	3.02%

Table 1: This table compares the standard key infection with the whisper-mode key infection. The first column lists *d*, the average number of neighbours of a node. The remaining columns list the ratio of compromised links (links that a black dust mote could eavesdrop on and extract both key infection messages).

	α = 1%		α = 2%		α = 3%	
d	basic	SA	basic	SA	basic	SA
2	1.20%	0.97%	2.29%	2.00%	3.38%	2.93%
3	1.81%	1.37%	3.44%	2.67%	5.42%	3.93%
4	2.30%	1.80%	4.45%	3.71%	6.50%	5.55%
5	2.93%	2.37%	5.73%	4.68%	8.73%	6.75%

Table 2: This table shows the improvement of secrecy amplification (SA) over the basic key infection.

nificantly better – where complexity and other constraints permit it.

6.2. Multihop keys

The second strategy is setting up multihop keys. If W_1 links to W_2 which links to W_3, then it may make sense for W_1 and W_3 to invoke W_2's help to set up a key that W_2 immediately forgets, against the eventuality that W_2 is compromised in the future. This has two purposes. First, it supports end-to-end, rather than link-level, cryptography. It is energy efficient for base-to-node communications to be encrypted using end-to-end keys rather than translated at intervening nodes; this also protects against subsequent node compromise. Second, multihop keying also protects multihop secrecy amplification against node compromise.

Where memory is not too restricted, multihop keying may be a very natural mechanism to use. Consider the unsolved problem mentioned by Sirois and Kent [16], that secure routing protocols that only use link-level protection (such as Nimrod with IPSEC) are vulnerable to node subversion. The point is that, in a ‘proper’ network, a router has the memory to set up a key with every other router for which it keeps a routing table entry.

In sensor networks such as Smart Dust, memory size and the energy cost of messages are the critical resource limits. However, sensor networks have quite limited types of traffic – mostly application messages between base stations and nodes, local routing / key management messages, and broadcasts of signals such as time beacons. Base-to-node traffic should be end-to-end encrypted anyway, as noted above; and as for routing, a number of multihop keying strategies are available, which require nodes to store more keys than they have neighbours but very much fewer than there are nodes.

6.3. Interaction with routing algorithms

Some existing work on secure ad-hoc routing assumes a particular routing strategy. Our work does not; existing prototypes use strategies based on dynamic source routing [13, 8] although our key infection protocol can also support other mechanisms.

These other mechanisms may have to be used to recover from attacks. For example, if sufficient nodes are subverted for the network to be partitioned – that is, there are pairs of motes that can no longer route to each other despite being physically connected by a multihop path – then a recovery phase may be initiated. This may involve backup nodes, if they exist; a re-run of the initial network discovery algorithm; or a strategy such as sticky random routing. There, we use a single path until it becomes unavailable, whether as a result of congestion or (in our application) damage, and then switch to an alternate [10]. Our multipath key infection protocol automatically discovers paths that may be used for this as needed. (This is one point at which the analogy with biological infection breaks down. In biology, the immune response normally stops you catching the same disease twice; if you are a smart dust mote, the more keys you ‘catch’ from a colleague, the better.)

In addition, multihop keying enables motes to try different logical paths along the same physical path, in order to identify and isolate a faulty or subverted node. If base sta-

	α = 1%		α = 2%		α = 3%	
d	basic	SA	basic	SA	basic	SA
2	0.38%	0.34%	0.75%	0.66%	1.25%	1.08%
3	0.59%	0.49%	1.15%	0.93%	1.75%	1.50%
4	0.81%	0.61%	1.67%	1.33%	2.27%	1.87%
5	1.04%	0.81%	2.03%	1.53%	3.15%	2.28%

Table 3: This table shows the improvement of secrecy amplification (SA) over the basic key infection. In this case, the basic key infection uses whispering.

	α = 1%		α = 2%		α = 3%	
d	basic	m-path	basic	m-path	basic	m-path
2	0.61%	0.38%	1.40%	0.48%	2.23%	1.11%
3	0.55%	0.26%	1.11%	0.58%	1.76%	0.91%
4	0.40%	0.16%	0.94%	0.30%	1.57%	0.80%
5	0.35%	0.04%	0.75%	0.12%	1.29%	0.40%

Table 4: This table compares the basic two-hop key infection, with the multipath extension. We simulate nodes that perform key infection with neighbours that are two hops away. In the columns marked "basic", we assume that the return path of the key infection is the same as the forward path. In the columns marked "m-path", the return message takes a different path, if available.

tion W_1 communicates with a sensor W_5 via the path W_2, W_3, W_4, then normally routing updates would be broadcast using a key shared with all these nodes [14]. However, more selective multicasts can be tried as a first recourse in the event of failure.

Finally, although most current sensor networks do not need to do mobile routing, the topology does still change – as a result of battery exhaustion and node destruction. In the future, we will need routing strategies that work for much more mobile principals, such as swarms of insect robots (which might be flying around gobbling up hostile dust motes).

7. Economic Issues

When analyzing the economics of a game, one approach is to look at the initial and marginal costs borne by each party. Thus, a company using a larger-scale production process than a competitor may have higher initial costs but lower marginal costs. Thus it may be dangerous for a small company to develop a market to the point at which it becomes attractive to a large competitor.

A similar effect appears here. If initial keying is used, this imposes a cost on the party deploying the dust (e.g., having a mechanism to distribute a seed key to the nodes in a canister just before deployment). It also imposes a cost on the attacker; in addition to having to pre-emplace black nodes with a certain density, he also has to physically recover a seed key from a white node. In practice, it may well be cheaper for white to key her nodes than for black to probe out a key, so a network-wide bootstrap key will be used. However, our discussion may perhaps have convinced the reader that this is only a small part of the equation. (Even if initial keying is used, and if the motes are moderately tamper-resistant, the use of the mechanisms described here may push down the value of recovering the bootstrap key below the cost of probing it out.)

Given an opponent with the capability to subvert nodes after deployment, the balance between attack and defense will move towards an equilibrium that will depend on the cost-benefit ratios of attack and of security maintenance. If these favor the defender, the attacker will give up; otherwise there will be an equilibrium at which the defender has to go all out [7, 18].

One implication is that, if the deployment is to be at all long lived, the initial costs for both parties will be dwarfed by the running costs. The designer of the white dust should focus on the cost of security maintenance rather than on over-investing in the initial trust bootstrap mechanisms. This mistake, though common enough, is from the viewpoint of an economic model somewhat like placing too much trust in security-by-obscurity. Security maintenance and renewability is often more important, in long-lived systems, than the height of the initial barrier placed before intruders.

8. Conclusions

We proposed a novel and quite counterintuitive way of managing keys in sensor networks: each node bootstraps it-self by broadcasting an initial key in the clear. Nodes then exchange keys and build up trust structures as they do network and resource discovery. It turns out that, under often reasonable assumptions, this is almost as secure as using pre-loaded initial keys. In other words, initial keying in such networks buys less than one might think.

Up till now, the research community has focused on mechanisms for establishing initial keys. This paper has shown how the benefits of initial keying can be analyzed separately from the benefits of later-stage key management activities, such as key updating, the use of alternative trust routes, and the invocation of backups – whether in response to perceived attacks, or periodically. Such resilience and recovery mechanisms are often more important than bootstrapping, but are widely ignored as operational matters about which there is little of academic interest to be said. We hope that contemplating systems without an initial trustworthy keying phase may be a good way to challenge this mindset.

Furthermore, the protection of sensor networks is a sufficiently compact problem to be accessible to network modelling techniques. We can explore, in a quantitative way, the benefits of differing schemes for initial keying, key updating and recovery from failure. Key infection is thus of interest not just as a curiosity, but as a tool for understanding more complex systems.

There are many environments in which an opponent can compromise any principal, but cannot compromise all of them. For example, a peer-to-peer system may be modeled as a large self-organizing network of principals, any small proportion of which can be subverted at will by the opponent. As such systems attract hostile action once they have grown to a certain size, the bootstrapping issues are not so important as what happens later.

What cryptography mainly achieves in such systems is to stop security compromises becoming worse over time. Whatever bad nodes managed to join the system at its inception remain there, until they show themselves (and hopefully get removed by higher layer mechanisms). If they choose not to show themselves, then in some applications (such as routing, where it's integrity that matters) they can perhaps be ignored.

Analogies with biological systems and even with human social structures may also be useful. The trust propagation and maintenance mechanisms described here mirror those in human society. You can almost certainly not remember the time when you decided to trust your mother! Initial trust bootstrapping is not such a big issue in many human organisations as stiffening members against later subversion. If our future is to be built on huge ad-hoc networks of communicating smart objects, from swarms of robot insects down to nanites circulating in our bloodstream, then the mechanisms we use to command and control them may be much more similar to societal mechanisms than to existing industrial control technology. The command and control of sensor networks appears to be a first step along the way to developing them.

Acknowledgment:

This work was done while the first author was visiting UC Berkeley on sabbatical in 2001-2, supported by NSF 9979852. The other authors are supported by NSF CNS-0347807 (CAREER) and by a gift from Bosch.

References

[1] R. Blom. Non-public key distribution. In *Advances in Cryptology: Proceedings of Crypto '82*, pages 231–236, 1982.

[2] C. Blundo, A. D. Santis, A. Herzberg, S. Kutten, U. Vaccaro, and M. Yung. Perfectly-secure key distribution for dynamic conferences. In *Advances in Cryptology - Crypto '92*, pages 471–486, 1992.

[3] H. Chan, A. Perrig, and D. Song. Random key predistribution schemes for sensor networks. In *IEEE Symposium on Security and Privacy*, May 2003.

[4] W. Diffie and M. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, IT-22(6):644–654, Nov. 1976.

[5] W. Du, J. Deng, Y. Han, and P. Varshney. A pairwise key predistribution scheme for wireless sensor networks. In *Proceedings of the Tenth ACM Conference on Computer and Communications Security (CCS 2003)*, pages 42–51, Oct. 2003.

[6] L. Eschenauer and V. D. Gligor. A key-management scheme for distributed sensor networks. In *Proceedings of the 9th ACM Conference on Computer and Communication Security*, pages 41–47, Nov. 2002.

[7] J. Hirshleifer. *Economic Behaviour in Adversity*. Chicago UP, 1987.

[8] Y.-C. Hu, A. Perrig, and D. B. Johnson. Ariadne: A secure on-demand routing protocol for ad hoc networks. In *Proceedings of the Eighth ACM International Conference on Mobile Computing and Networking (Mobicom 2002)*, Sept. 2002.

[9] J. M. Kahn, R. H. Katz, and K. S. Pister. Mobile networking for smart dust. In *International Conference on Mobile Computing and Networking (MobiCom '99)*, Seattle, WA, August 1999.

[10] F. P. Kelly. Modeling communication networks, present and future. *Philosophical Transactions of the Royal Society*, A354:437–463, 1996.

[11] D. Liu and P. Ning. Establishing pairwise keys in distributed sensor networks. In *Proceedings of the Tenth ACM Conference on Computer and Communications Security (CCS 2003)*, pages 52–61, Oct. 2003.

[12] D. Liu and P. Ning. Location-based pairwise key establishments for static sensor networks. In *ACM Workshop on Security in Ad Hoc and Sensor Networks (SASN '03)*, Oct. 2003.

[13] R. Perlman. *Network Layer Protocol with Byzantine Agreement*. PhD thesis, The MIT Press, Oct. 1988. LCS TR-429.

[14] A. Perrig, R. Szewczyk, V. Wen, D. Culler, and J. D. Tygar. SPINS: Security protocols for sensor networks. In *Proceedings of ACM International Conference on Mobile Computing and Networks (MobiCom 2001)*, pages 189–199, July 2001.

[15] R. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, Feb. 1978.

[16] K. Sirois and S. Kent. Securing the nimrod routing architecture. In *Proceedings of the Symposium on Network and Distributed Systems Security (NDSS '97)*. Internet Society, Feb. 1997.

[17] F. Stajano and R. Anderson. The resurrecting duckling: Security issues for ad-hoc wireless networks. In *Security Protocols—7th International Workshop*, volume 1796 of *Lecture Notes in Computer Science*, pages 172–194. Springer-Verlag, Berlin Germany, 2000.

[18] H. R. Varian. System reliability and free riding. In *Workshop on Economics and Information Security*, May 2002.

[19] S. Zhu, S. Setia, and S. Jajodia. LEAP: Efficient security mechanisms for large-scale distributed sensor networks. In *Proceedings of the Tenth ACM Conference on Computer and Communications Security (CCS 2003)*, pages 62–72, Oct. 2003.