Appeared in the Proceedings of the Second International Workshop on Sensor and Actor Network Protocols and Applications, Boston, Mass., USA, August 2004, pages 21-29.

Privacy-Sensitive Monitoring With a Mix of IR Sensors and Cameras

Abhishek Rajgarhia, Fred Stann, {rajgarhi, fstann}@usc.edu,
John Heidemann {johnh@isi.edu}

Abstract

The goal of monitoring an area for security often conflicts with the inhabitants' right to privacy. This paper argues that technical choices can be made to balance the privacy-security tradeoff. We present a privacy-sensitive security monitoring system which balances privacy and security by deploying cameras only in public areas and by distributing information about people's movements in private areas so that no single computer can track an individual. We show how this distributed information can be utilized to correlate an interesting event such as a theft with an image taken by the cameras deployed at the public areas. We then survey potential users of our system to evaluate preferences concerning cameras and motion sensors, and we present analysis of three alternative locations of storing data.

1. Introduction

Advances in computing, wireless and sensing technology have enabled ubiquitous computers, sensor networking and the ability to surround our environments and workplaces with sensors. We have also seen a proliferation of cameras in public places to the point where there are many outdoor urban environments where it is impossible to avoid being subject to video surveillance. Not surprisingly, many people are concerned about privacy implications of ubiquitous sensing [1]. Ubiquitous surveillance cameras raise the unsettling specter of "Big Brother." Yet the goal of surveillance is often improved security and most people would accept cameras at banks and ATMs, for example to discourage robbery or muggings. Thus the goal is to balance privacy and security and make informed choices about trade-offs, rather than simply making a binary decision.

The premise of this paper is that there are technical choices that can balance the privacy-security trade-off. We explore a Privacy-Sensitive Security Monitoring System that seeks to balance these issues by

(a) Distributing information about people's movements so no single computer can track an individual.
(b) Placing cameras only in public areas such as lobbies where security staff are already present.
(c) Providing a means whereby an explicit action, such as a court order or theft, can correlate a location and time with an image of an individual.

In our system, cameras in public areas periodically collect images and timestamp them. Private areas, such as hallways and office suites are densely deployed with Passive Infra - Red (PIR) motion sensors. These sensors collect data about the times at which private areas were accessed. All data is stored locally at a sensor node. When a theft occurs the system carries out a *spatio-temporal search* of the distributed data collected and stored by the network and tries to correlate a location and time of event such as a theft with a picture. This picture is then reliably transferred to a display node using Reliable Multi-Segment Transport (RMST) [14].

Thus the contribution of this paper is the development of a system that balances privacy and security as described above. We evaluate this system through experiments, and we confirm the privacy benefits of reducing camera deployment by surveying potential subjects of such a system.

We first demonstrated our operational monitoring system at Sensys 2003 [10]. This paper makes several contributions beyond that demonstration. First, it provides a detailed description of the system. Second, a premise of our work is that people prefer motion sensors to cameras in the workplace. We surveyed 60 people and report those findings in Section 4. Finally, we analyze the communications cost of localized storage compared to centralized and data-centric storage (section 5).

2. Design

In our system an office environment is instrumented with two kinds of sensors: cameras and motion detectors (passive infra-red sensors). We place cameras only in public areas, perhaps by the front desk where there may already a receptionist or guard. We place motion detectors throughout the building, approximately every office doorway. Both types of sensors continuously take readings and store readings locally for several days or more. (Local storage is very inexpensive; flash memory is sufficient for small motion detections while cameras can use hard disks. At today's storage prices, $5 of storage per sensor is sufficient for months worth of detections. In addition, local storage avoids the energy cost of sending possibly never used events from battery powered nodes to a central site.) This configuration of sensors provides two aspects of privacy. First, since sensed information is

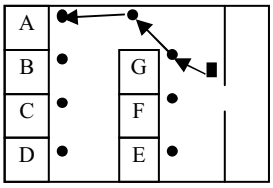

Figure 1: Typical office layout

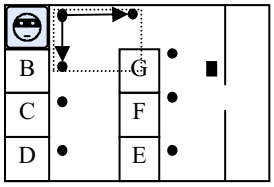

Figure 2: Room A's node initiating search

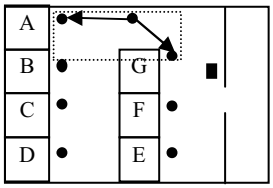

Figure 3: Query being propagated

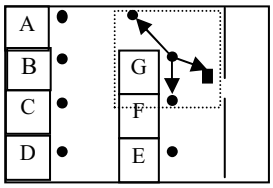

Figure 4: Query reaches camera node

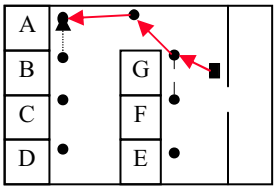

Figure 5: Reponses begin propagated

distributed, no single computer or sensor can track an individual. We presume that distributed access to multiple sensors must be more explicit and can be more easily controlled. Second, people already have reduced expectation of privacy in public areas. Many buildings have receptionists or guards observing who enters buildings. Thus use of cameras in such public areas should not be as onerous as in private areas. (We verify this assumption in Section 4 by surveying people's expectations.) Finally, we desire a mechanism by which, in response to an explicit action (such as a court order or due to a theft) we can correlate a location and time with an image of an individual by connecting a *chain of events* through the sensor network from the location of the theft to a camera. We assume private areas are deployed densely enough with sensors so that a human moving through the sensor field triggers multiple sensors. The system is designed for monitoring at times when the floor is sparsely populated such as at night. We also assume that the sensor nodes know their location in terms of a two-dimensional frame of reference. The system is designed to achieve two aims.

- To correlate a location and time of an event such as a theft with an image stored in a sensor node monitoring a public area, on demand.
- Transfer the related picture reliably over the sensor net, to a display node.

We now describe how the chain of events is formed. In regular operation, each sensor detects readings and stores the data locally. If a theft occurs, nodes initiate a spatio-temporal search through the sensor net. The query originates at the node where the theft occurred and is tagged with the time (or times) of detections at that node. The query then is made against nearby nodes for any detections in the recent past. This query propagates through the network in space (moving from node to node) and time (adjusting the time window of interest). In this way we can establish a chain of events and track the intruder back to a sensor node which contained the intruder's picture. When a chain of events is established from the point of theft to a node with a camera, the query has identified an image potentially correlated with that event. We then send that image over the sensor network to a display node.

Figure 1 shows a typical building floor layout. A-G are private areas monitored by sensor nodes having only PIR sensors. Suppose the thief traverses the path shown by the arrows in Figure 1 and that the theft occurs in area A. Figures 2-4 show how the query traverses the sensor net. Since the theft occurs at A, the query originates at A. The sensor node at A tags the query with times of detection and its own location. This query is sent to nearby nodes. The nodes receiving the query calculate the distance between themselves and the source node and assuming some reasonable speed (1.25 m/s) calculate the time taken by a human being to traverse the distance. They subtract this time from the timestamp in the query and use a jitter to calculate a time window. The jitter is used to compensate for variations in speed. They query their local database for detections in this window. If they find detections, the nodes propagate the query, as done by the node to A's right in Figure 3, by replacing the source node's location by their own location and the timestamp by the times of detection in their local database. Eventually the query reaches the sensor node which has a camera as shown in Figure 4.

We use directed diffusion [6] as the query dissemination protocol. Attribute matching [12] is used to ensure that a query is received only by nearby nodes. Reliable Multi-Segment Transport (RMST) [14] is used to transfer the image to a display node.

Figure 5 shows how the responses are transmitted along the backward path taken to the query. Dotted lines show negative response and solid lines show positive response.

3. Implementation

The network is composed of Intel Stargate boards which have ARM processors and run linux-2.4.19. The node which monitors the entrance has Video4linux support, USB support and the OV511 driver installed. In addition to Stargates we have stand-alone Mica2 motes which interface with PIR sensors, as shown in Figure 7. Mica2 motes are attached to the Stargates, so that they can send and receive diffusion packets and packets from stand-alone motes. When a sensor is triggered the motes unicast an S-MAC packet to a nearby Stargate, indicating presence of a person. The S-MAC address of the Stargate is provided to each mote by hand. The Stargate boards run diffusion-3.2.0.

We assume that every node knows its location in terms of a 2-D frame of reference. In our current system, coordinates are manually configured. When a theft occurs, a spatio-temporal search is initiated at the site of the theft.

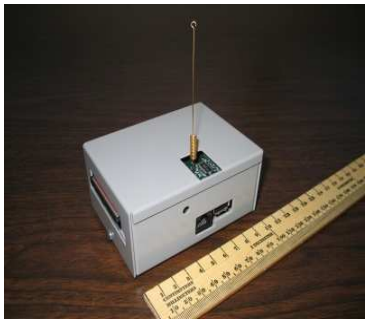

Figure 6: Stargate

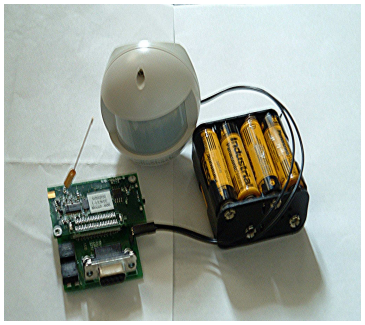

Figure 7: PIR sensor interfaced with mote

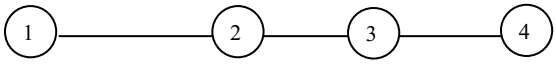

Figure 8: Successive nodes in path taken by query

The search involves sending of queries and responses through the network. One-phase-pull is used in the search because it has lower control traffic compared to two-phase-pull [4]. A small change is made to diffusion. In the current implementation of diffusion, if an application expresses an interest, interest messages are transmitted repeatedly by diffusion. The mean time interval between two transmissions is 30 seconds. This is clearly not suitable for our system because, since we query only nearby nodes the time taken to get responses to a query from nearby nodes is about one round-trip-time plus the time taken to process the query. This time is much lesser than 30 seconds. Hence, the responsibility of retransmitting interest messages is taken over by the application. The application sends out interest messages, once every round-trip-time. Some jitter is added to avoid synchronization.

3.1 Propagation of Queries

A node situated at (x,y) is interested in receiving queries from nearby nodes. Hence it has a standing local interest in which it specifies its coordinates.

If the node at the site of theft is located at (x_s,y_s) and the PIR sensor at the site of theft was triggered at time t, a query in the form of an interest message is sent out. The interest message specifies the coordinates (x_s,y_s), a network-wide query id and a rectangular region around (x_s,y_s). The query is routed by Directed Diffusion to nodes in this region. The interest message is tagged with all the times at which the PIR sensor of the node at the site of theft was triggered.

When a node at (x,y) receives a query it runs the following algorithm.

1. Check cache of query ids seen to see if the query is duplicate. If the query is duplicate, goto step 10
2. Add query id to cache of queries seen.
3. Retrieve coordinates of node which sent out the query. Let it be (x_s,y_s). Find distance d, between itself and (x_s,y_s).
4. Assuming a reasonable speed (say 1.25 m/s), calculate time t_d a human might have taken to traverse distance d.
5. Let T=empty set of timestamps
6. For each time t with which the query is tagged Look up local database for any local detections in interval $(t-t_d+j, t-t_d-j)$, where j is a small jitter. For each t1, in the interval at which an event had occurred, add t1 to T.
7. If T is empty publish a negative response to (x_s,y_s). Goto step 10.
8. Otherwise T is not empty and so we found one or more detections linking the chain of events through the local node. Propagate the query tagging it with (x,y), a rectangular region around (x, y) and T.
9. Publish a positive response to (x_s,y_s), tagging the response with (x,y) and T.
10. End.

Every time a node has a positive response to a query, it propagates it tagging with different timestamps and rectangular regions. Hence the query moves in space and time.

3.2 Handling Replies

To be robust to message loss the system must detect and retry lost queries. Two complications arise in handling replies: first, knowing how long to delay before retrying, and second, handling unevenly spaced sensors (when the initial query region misses the nearest sensor). When a node directly queries its neighbor directly it is easy to estimate how long this query should take: approximately the time to transmit the query, compute the reply, and send the reply. All of these times are small and can be bounded so we can set a short timer for an appropriate interval *T* and resend the query if no response arrives. But to establish a chain of events through the sensor network, computing a reply at one node can require querying another node recursively. Thus the "compute a reply" step can involve an arbitrary number of sub-queries and so an arbitrary delay. For example, in Figure 8, imagine that node 1's query must proceed to node 4 before finding a camera. This query will require at least 3*T* time to complete.

To resolve this problem, intermediate nodes can send a "response in progress" message in reply to a query. When node 1 sends its query, it sets a timer for *T*. If it has not heard a reply from 2, it resends the query and 2 replies immediately with "response in progress". Interests are sent out a fixed number of times, irrespective of the number of

responses heard. This is in the hope of finding more number of nearby nodes, which may have correlated detections.

For ease of deployment, sensors may be placed unevenly. Hence when a query is sent out by a node, it may not be heard by any node, because the rectangular region to which the query is routed doesn't contain any other nodes. Hence if a node, doesn't hear any response for its query within 3*T*, it sends the query to a larger rectangular region. When the size of the region reaches a threshold, the node gives up.

3.3 The Camera Node

The camera node keeps taking pictures every second. When a camera node gets a query and it was triggered at time *t* in the window of interest that it calculates for the query, it transmits the latest picture that was taken before or at *t*-3 seconds using RMST to a display node. This is done because we want the picture that was taken a while before the sensor was triggered to account for the delay that is introduced when sending a S-MAC packet from the stand-alone mote to the sensor node and for the camera's limited field of view. After a night's monitoring, if it is discovered that there were no thefts in the building, the images can be deleted, to reclaim space from the camera node. For one night's monitoring 200MB space is required for the images. The images can be stored on disk. Disk space can be saved by using a simple compression algorithm, which discards pictures with no major differences.

4 Survey of Privacy Perceptions

	Sensors	Cameras
All	2.1 (1.6; 1.7-2.5)	3.1 (1.7; 2.6-3.5)
Men	2.5 (1.6; 1.9-3.1)	2.8 (1.9; 2.1-2.5)
Women	1.7 (1.5; 1.1-2.3)	3.3 (1.5; 2.7-3.8)

Table 1: Comparision of level of concern for motion sensor monitoring vs. camera monitoring. Values are mean (standard deviation; low 95% confidence interval-high 95% confidence interval)

Our work is based on the assumption that people are less bothered by being monitored by motion sensors in private areas than cameras in private areas. To validate this assumption we carried out a survey, asking people to rate the level of their concern regarding various privacy and security aspects on a scale of 0 (not concerned) to 5 (very concerned). To minimize order effects, two separate orderings of questions were used.

Thirty men and thirty women participated in the survey. The subjects were composed of college students and professionals. The survey consisted of 22 questions to calibrate different levels of privacy, ranging from innocuous things such as filling out census forms, to invasive actions such as theft. The full survey is reproduced in Appendix A.

Table 1 summarizes the results for the two key questions: "security cameras record your picture continuously at work" and "motion sensors record your presence continuously at work". We report mean scores and 95% confidence intervals for all people surveyed and for men and women. In both cases, mean scores indicate that cameras were considered more invasive than simple motion sensors. There was substantial variation in the responses, however. Surprisingly, when confidence intervals are considered, a difference in perception based on gender. For men, although there is a difference in the means, it is not statistically significant since 95% confidence intervals overlap the mean. For women, however, the difference is statistically significant. It is possible that with a larger survey both groups would demonstrate statistically significant differences.

5. Analysis of Communication Costs

In this section we compare the communication cost of our search, which uses local storage of event data with two approaches; sending all data to a central site and using Data-centric storage [11]. We measure communications cost as an estimate of node lifetimes if motion sensors are battery powered. While there is certainly an ongoing cost to keep a network active, recent energy-conserving MAC protocols (for example, S-MAC [16, 17], T-MAC [15], or TRAMA [9]) minimize ongoing listen costs but cannot reduce transmit costs.

The communication cost is composed of two components. Operating cost or the cost incurred in sending event information to the destination and search cost or the cost incurred in searching the data stored in the network, in event of a theft. Let n be the number of sensor nodes in the network. Over a time period t, let there be e events and s searches. We assume the diameter of the network is $\sqrt{n}$. Note that with our hardware the motion sensors are on motes, physically distinct from the Stargates.

5.1 Central Storage

Then worst-case operating cost is given by.

$$C_{\text{central-operating}} = e\sqrt{n}$$

Since all the data is available at one place, the search cost is constant. There are s searches. Hence,

$$C_{\text{central-total}} = e\sqrt{n} + s$$

We observe that the cost of a centralized system is dominated by the cost of moving events to a central location. Thus, in addition to providing a weaker privacy model it is more expensive in communications cost.

5.2 Data-Centric Storage

The communication cost incurred depends on the function used to map event data to location. If we name all event data with a generic name such as "EVENT_DATA" and map this string to a geographical location, then all event data will

be sent to one single location and this will reduce to a central storage approach. If we use the geographical coordinates of the location of the event, event data will be stored locally and this reduces to our approach. We now analyze the worst-case communication cost if data-centric storage is used. In the data centric-storage approach, in the worst case, a cost of $e\sqrt{n}$ will be incurred for transmitting data about events to their respective destinations.
Hence,

$C_{DCS\text{-}operating} = e\sqrt{n}$

Further cost will be incurred for the search. Let r be the number of nodes in the rectangular region to which a query is sent and let L be the length of the maximal path from entrance to any possible site of theft. Let p be the probability that any node will have a positive response for a query. Then the worst-case communication cost for a search which uses data-centric storage is given by

$C_{DCS\text{-}search}$ = (Cost of transmitting queries) + (Cost of positive responses) + (Cost of transmitting negative responses)

At a distance of i hops from the node which started the search, the number of nodes which will be active (i.e. which will propagate queries) is $p^i r^i$. Each of these nodes have to send the query to r separate locations and to send the query to a particular location it will take $\sqrt{n}$ messages. Hence,

Cost of transmitting queries = $\sum_{i=0}^{L} r\sqrt{n}\,p^i r^i$

At a distance of i hops from the node which started the search, there will be $p^i r^i$ nodes with positive responses. Each of these nodes will send the positive response to the node which sent them the query. Each positive response will take $\sqrt{n}$ message transmissions. (We will assume that nodes which had positive responses at *i-1* hops from the node which started the search will group the positive responses of the nodes at i hops from the node which started the search with their own positive response.) Then the cost of transmitting positive responses is given by

Cost of transmitting positive responses = $\sum_{i=1}^{L} \sqrt{n}\,p^i r^i$

At a distance of i hops from the node which started the search, there will be $(1-p)p^{i-1}r^i$ nodes with negative responses. Each negative response takes $\sqrt{n}$ messages in the worst case. Hence,

Cost of transmitting negative responses =
$\sum_{i=1}^{L} \sqrt{n}(1-p)p^{i-1}r^i$

Adding costs of transmitting queries, positive responses and negative response

$C_{DCS\text{-}search} = \sum_{i=0}^{L} r\sqrt{n}\,p^i r^i + \sum_{i=1}^{L} \sqrt{n}\,p^{i-1}r^i$

Since there are s searches, the total cost of using data-centric storage in the worst case is given by

$C_{DCS\text{-}total} = C_{DCS\text{-}operating} + s\cdot C_{DCS\text{-}search}$

$C_{DCS\text{-}total} = e\sqrt{n} + s\left(\sum_{i=0}^{L} r\sqrt{n}\,p^i r^i + \sum_{i=1}^{L} \sqrt{n}\,p^{i-1}r^i\right)$

We observe that data centric storage is strictly more expensive than centralized storage overall. (Although, as with the observations in [11], DCS may have better distribution of energy consumption by eliminating hot spots.)

5.3 Local Storage

In local storage, the communication cost for e events is e (1 per event).

$C_{local\text{-}operating} = e$

$C_{local\text{-}search}$ = (Cost of transmitting queries) + (Cost of positive responses) + (Cost of transmitting negative responses)

Cost of transmitting queries = $\sum_{i=0}^{L} r p^i r^i$

For local storage the cost required to transmit each query in r instead of $\sqrt{n}$.

Since for a positive/negative response a unicast is sufficient.

Cost of transmitting positive responses = $\sum_{i=1}^{L} p^i r^i$

Cost of transmitting negative responses = $\sum_{i=1}^{L} (1-p)p^{i-1}r^i$

Adding costs of transmitting queries, positive responses and negative response

$C_{local\text{-}search} = \sum_{i=0}^{L} r p^i r^i + \sum_{i=1}^{L} p^{i-1}r^i$

Since there are s searches,

$C_{local\text{-}total} = C_{local\text{-}operating} + s\cdot C_{local\text{-}search}$

$$C_{\text{local-total}} = e + s(\sum_{i=0}^{L} r p^i r^i + \sum_{i=1}^{L} p^{i-1} r^i)$$

Comparing $C_{\text{local-total}}$ to $C_{\text{DCS-total}}$, we observe that both storage and search require a factor of $\sqrt{n}$ less messages. This result is because our application is optimal for local storage: data generation is strictly local, and queries are geographically scoped.

We can prove that if *pr*=1 and *s*=1 then local storage is better than central storage.

Put *pr*=1 in expression for $C_{\text{local-search}}$. Then

$C_{\text{local-search}} = r(L+1) + rL = 2rL + r < 2r(L+1)$

Hence,

$C_{\text{local-total}} < 2r(L+1) + e$

Since $\sqrt{n}$ is the diameter of network and we query only nearby nodes, $r<(\sqrt{n}-1)/2$ and the length of the path is bounded by *e*. Hence we can assume the $L+1<e$.

Hence,

$r(L+1) < e(\sqrt{n}-1)/2$

or

$2r(L+1) + e < e\sqrt{n}$.

Recall that for *s*=1, $C_{\text{central-total}} = e\sqrt{n} + 1$

Hence if *pr*=1 and *s*=1,

$C_{\text{local-total}} < C_{\text{central-total}}$

6. Related Work

Many location-based applications have privacy as one of their design goals. In this section we look at some of these systems and see how their approach and goals differs from our those of our system. We also look at a system which employs a spatio-temporal search.

6.1 Privacy Aware Location-Based Systems

There are many approaches to address the issue of privacy in location-based applications.

- Put privacy policies in place which govern access to the location information at a central server.
- Employ data perturbation and anonymization before data leaves the sensor network. E.g. Privacy-Aware Location Sensor networks [2].
- Design the system so that user or a trusted agent can control who receives location information. E.g. Cricket Location-Support System [8].

As noted by Grtueser et. al [2] anonymization and access control mechanisms become less effective, if the data collecting agents are owned by a third party. Our approach differs in that no single node keeps track of where a person moves. Motion sensors detect someone moving, but cannot identify who. The camera sensors record who is present in the lobby, but not where they go. Only by querying the network as a whole is sensitive information generated and we assume that that functionality can be monitored and controlled. Myles, Friday and Davies [7] describe a system in which a location server uses validators and a XML encoded privacy policies of the application to decide whether information should be released to the application. In this system users have to trust the location server to adhere to their policy requirements and have to trust that the location system is not vulnerable to attack. Our system differs because we don't employ a central server. Privacy-Aware Location Sensor networks employ data perturbation, which requires users to trust the sensors [2]. Gruteser and Grunwald [3] analyze the feasibility of employing spatio-temporal anonymization to location information. Our system doesn't require the employment of these schemes because as specified above sensitive information is generated by querying the network as a whole. In the Cricket-Location Support System [8] devices calculate their positions by listening to strategically placed beacons. The location information is then published to applications via an API. The devices control the publication. Spreitzer and Theimer [13] describe an architecture in which every user has a trusted agent which governs access to location information. Our system doesn't seek to actively track any device or individual.

6.2 Spatio-Temporal Search

Huang, Lu and Roman describe Mobicast [5] in a spatio-temporal multicast protocol for sensor networks useful for tracking "mobile phenomena", in which active sensors at any point it time warn sensors close to the object being tracked that the object is headed their way. Our spatio-temporal search is different because we don't seek to actively monitor objects but to exploit spatio-temporal dependencies to generate possible paths which an object might have taken after an interesting event such as a theft takes place.

7 Further Work

We have tested a prototype of our system in a laboratory setting and have demonstrated it at ACM Sensys 2003. It was found that the search algorithm could successfully correlate a theft event with a picture of the intruder. We consider it important to deploy the system in a real-world setting.

8 Conclusion

In this paper we argued for deployment of more privacy-sensitive security monitoring systems. We presented the design, implementation and analysis of a privacy-sensitive security monitoring system, which balances privacy with security by using motion sensors in private areas instead of cameras. In a survey we found that women, showed a significant preference for motion sensor monitoring than camera monitoring. We have successfully tested a prototype of our application in laboratory conditions and proved that under certain circumstances our system incurs lesser communication costs than other approaches of data storage.

9 Acknowledgements

We would like to thank Dr. Michael Noll and Dr. Lynn Miller for their help with formulating and administering the survey

References

[1] David H. Flaherty, Video surveillance by public bodies: a discussion (Investigation P98-012)," Office of the Information and Privacy Commissioner for British Columbia, March 31, 1998.

[2] Marco Gruteser, Graham Schelle, Ashish Jain, Rick Han, and Dirk Grunwald. Privacy-Aware Location Sensor Networks. In *Proceedings of USENIX 9th Workshop on Hot Topics in Operating Systems (HOTOS IX)* 2003.

[3] Marco Gruteser and Dirk Grunwald. Anonymous Usage of Location-Based Services Through Spatial and Temporal Cloaking. In *Proceedings of the First International Conference on Mobile Systems, Applications, and Services*, May 2003.

[4] J. Heidemann and F. Silva. Matching Data Dissemination Algorithms to Application Requirements. In *Proceedings of the First International Conference On Embedded Networked Sensor Systems,* Los Angeles, CA, USA, Nov. 2003.

[5] Q. Huang, C. Lu, G.-C. Roman. Spatiotemporal Multicast in Sensor Networks. In *Proceedings of the First International Conference On Embedded Networked Sensor Systems,* Los Angeles, CA, USA, Nov. 2003.

[6] C. Intanagonwiwat, R. Govindan, and D.Estrin. "Directed Diffusion: A scalable and Robust Communication Paradigm for Sensor Networks." In *Proceedings of ACM/IEEE International Conference on Mobile Computing and Networking* ,pages 56-67, Boston, MA, USA, August 2000.

[7] Ginger Myles, Adrian Friday, and Nigel Davies. Preserving Privacy in Environments with Location-Based Applications. *IEEE Pervasive Computing*, 2(1):56-64, 2003.

[8] Nissanka B. Priyantha, Anit Chakraborty, and Hari Balakrishnan. The Cricket Location-Support System. In *Proceedings of the sixth annual international conference on Mobile computing and networking*, pages 32-43. ACM Press, 2000.

[9] Venkatesh Rajendran, Katia Obraczka, and J.J. Garcia-Luna-Aceves. Energy Efficient, Collision-Free Medium Access Control for Wireless Sensor Networks. In *Proceedings of the First ACM SenSys Conference*, pp. 181-193. Los Angeles, California, USA, ACM. November, 2003.

[10] Abhishek Rajgarhia, Fred Stann, and John Heidemann. Privacy-Sensitive Monitoring With a Mix of IR Sensors and Cameras. *Technical Report ISI-TR-582*, USC/Information Sciences Institute, November, 2003.

[11] Sylvia Ratnasamy, Deborah Estrin, Ramesh Govindan, Brad Karp, Scott Shenker, Li Yin, Fan Yu. Data-Centric Storage in SensorNets. In *ACM First Workshop on Hot Topics in Networks*, 2001

[12] Fabio Silva, John Heidemann and Ramesh Govindan. Network Routing Application Programmer's Interface (API) and Walk Through .

[13] Mike Spreitzer and Marvin Theimer. Providing Location Information in a Ubiquitous Computing Environment. In *Proceedings of the Fourteenth ACM Symposium on Operating System Principles*, pages 270-283, 1993.

[14] Fred Stann and John Heidemann: "RMST: Reliable Data Transport in Sensor Networks." In *Proceedings of the First International Workshop on Sensor Net Protocols and Applications*, Anchorage, Alaska, USA, April 2003.

[15] Tijs van Dam and Koen Langendoen. An Adaptive Energy-Efficient MAC Protocol for Wireless Sensor Networks. *In Proceedings of the First ACM SenSys Conference*, pp. 171-180. Los Angeles, California, USA, ACM November, 2003.

[16] Wei Ye, John Heidemann, and Deborah Estrin. An Energy-Efficient MAC protocol for Wireless Sensor Networks. In *Proceedings of the IEEE Infocom*, pp. 1567-1576. New York, NY, USA,USC/Information Sciences Institute, IEEE. June, 2002.

[17] Wei Ye, John Heidemann, and Deborah Estrin. Medium Access Control with Coordinated, Adaptive Sleeping for Wireless Sensor Networks. *Technical Report N. ISI-TR-567*, USC/Information Sciences Institute, January, 2003 accepted to appear IEEE/ACM Transactions on Networking.

Appendix A: Detailed Survey Results

Question	All	Men only	Women only
A stranger taps your telephone line and records your conversations.	4.1 (1.7)	4.0 (1.7)	4.2 (1.7)
An unauthorized person has access to your e-mail at work.	4.2 (1.4)	4.0 (1.7)	4.3 (1.1)
A stranger peeps into your window at home.	4.2 (1.2)	4.1 (1.1)	4.3 (1.2)
Person you call records your phone number with caller-id.	2.0 (1.6)	2.2 (1.7)	1.8 (1.5)
Security cameras record your picture continuously at a shopping mall.	1.9 (1.7)	2.2 (1.8)	1.6 (1.5)
You get abusive phone calls at work-place.	3.5 (1.8)	3.2 (1.9)	3.8 (1.6)
A stranger is loitering around your work-place.	3.1 (1.7)	2.7 (1.7)	3.4 (1.6)
Motion sensors record your presence continuously at a shopping mall.	1.5 (1.5)	1.7 (1.5)	1.3 (1.5)
A census worker has you fill out a census form.	1.2 (1.3)	1.5 (1.4)	1.0 (1.1)
Office supplies from your organization are stolen.	2.8 (1.5)	2.8 (1.3)	3.0 (1.6)
A shop-keeper asks you for your credit-card number.	3.0 (1.7)	3.1 (1.7)	2.8 (1.7)
A visitor arrives unannounced at your work-place.	1.9 (1.3)	1.8 (1.3)	1.9 (1.4)
Motion sensors record your presence continuously at work.	2.1 (1.6)	2.5 (1.6)	1.7 (1.5)
A stranger peeps into your work-place.	2.6 (1.5)	2.7 (1.5)	2.6 (1.6)
A shop-keeper asks you for your Social Security number.	4.0 (1.3)	4.0 (1.4)	3.8 (1.1)
A stranger looks through your wallet or purse (but doesn't take anything).	3.9 (1.4)	3.8 (1.4)	4.0 (1.4)
Your personal belongings are stolen.	4.5 (1.2)	4.4 (1.2)	4.5 (1.3)
Security cameras record your picture continuously at work.	3.1 (1.7)	2.8 (1.9)	3.3 (1.5)
An unauthorized person has access to your computer at work.	3.7 (1.7)	3.8 (1.7)	3.6 (1.7)
Sensitive information/documents from your organization are stolen.	4.0 (1.7)	3.9 (1.7)	4.1 (1.7)
An unauthorized person has access to your office.	3.5 (1.7)	3.3 (1.6)	3.8 (1.7)
You are physically assaulted at your work-place.	3.8 (1.8)	3.5 (1.9)	4.0 (1.7)

Table 2: Detailed survey results. Values are mean (Standard deviation)