Edge Eavesdropping Games

Amos Beimel* Matthew Franklin†

July 2, 2006

Abstract

Motivated by the proactive security problem, we study the question of maintaining secrecy against a mobile eavesdropper that can eavesdrop to a bounded number of *communication channels* in each round of the protocol. We characterize the networks in which secrecy can be maintained against an adversary that can eavesdrop to t channels in each round. Using this characterization, we analyze the number of eavesdropped channels that complete graphs can withhold while maintaining secrecy.

Keywords: unconditional security, passive adversary, mobile adversary, graph search games.

1 Introduction

Many cryptographic protocols are secure if an unknown *fixed* set of processors of bounded size is dishonest. Proactive security [13, 9] considers a more realistic scenario, where a mobile adversary can control a different set of processors of bounded size in each period. Protocols in the proactive model have to cope with a stronger adversary, which, for example, might have controlled every processor by some point during the protocol execution. In protocols secure in the proactive model, each processor has to "spread" the secret information it holds.

Franklin, Galil, and Yung [6] studied maintaining secrecy against a mobile eavesdropper which can eavesdrop to a bounded number of processors in each round of the protocol. Unfortunately, we discovered that the main characterization given in [6] of maintaining secrecy against a mobile eavesdropper is incorrect. We describe the flaw in their proof and the correct characterization, see Section 1.2. The main focus of this paper is a similar question, where a mobile eavesdropper can eavesdrop to a bounded number of *communication channels* in each round of the protocol. As eavesdropping to communication channels is easier than eavesdropping to processors, this is a natural question. Although the two problems are similar, there are differences between the two problems, for example in the number of rounds that an adversary can learn the secret information in a complete graph while eavesdropping to minimal number of vertices or edges respectively.

To model the question of maintaining the secrecy of a system against a mobile adversary that can eavesdrop to communication channels, we consider the following abstract game, similar to [6],

*Department of Computer Science, Ben-Gurion University. On sabbatical at the University of California, Davis, partially supported by the Packard Foundation.

†Department of Computer Science, University of California, Davis. Partially supported by NSF and the Packard Foundation.

called the distributed database maintenance game. There is a protocol trying to maintain the secrecy of one bit b in the system. The first stage in the game is an initialization stage in which each edge gets an initial value. (This abstracts an intermediate state of a more complex protocol.) In Round i, each vertex receives messages, and sends messages generated based on the messages it received in the previous round and a "fresh" random string. The secret bit b can be reconstructed in each round of the protocol from the messages sent in the system in that round. The mobile adversary eavesdrops to t channels of its choice in each round. We require that an unbounded adversary cannot learn the secret from the messages it heard. The adversary can only eavesdrop to channels; it cannot change, insert, or delete messages.

Following [6], because of the close connection with "graph search games" [14, 11], we refer to the eavesdropping to a channel as placing a "guard" on this edge, and we say that a graph is "cleared" at the end of a "search" (finite sequence of subsets of edges the adversary eavesdrops) if the adversary has collected enough information to infer the secret bit b.

The search terminology we use is somewhat unnatural as it "takes the side" of the adversary. That is, the adversary defeats the secrecy of a protocol if it clears the graph. In contrast, a protocol maintaining privacy should prevent the adversary from clearing the graph. Furthermore, we will define when a vertex is contaminated after i rounds of the protocol, and the protocol maintains privacy if there is at least one contaminated vertex at the end of the protocol. The reason we use the graph search terminology is that it describes the interplay between the protocol and the adversary as contamination tends to spread unless the adversary guards the edges adjacent to contaminated vertices.

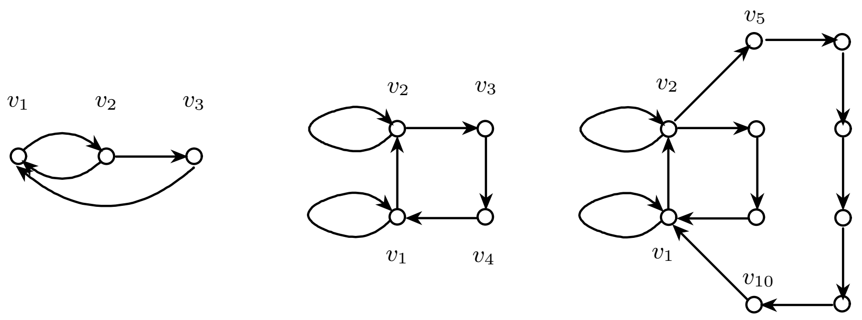
We consider two variants of the edge eavesdropping game, depending on whether the underlying communication network is modeled as a directed or an undirected graph. When the network is modeled as an undirected graph, each edge is a full-duplex channel, and a single eavesdropper can monitor the message flow in both directions. When the network is modeled as a directed graph, each edge allows communication in one direction only, and a single eavesdropper can monitor the message flow in that direction only. Note that a full-duplex channel can be represented as a pair of directed edges, but then two eavesdroppers are required to monitor the message flow in both directions.

To see some of the subtleties of edge eavesdropping games, consider the three graphs described in Figure 1. A single guard can clear these graphs, and thus the distributed database maintenance game on these networks is defeated by an adversary controlling a single mobile eavesdropper. An explanation of these examples can be found in Example 3.5 and in Section 4.3.

1.1 Our Results

Our first result (Theorem 3.2) is a characterization of when a search clears a graph. Given a directed or undirected graph G and given a search of length ℓ, we construct an undirected layered version of the graph where the number of layers is the length of the search. In the layered graph there are $\ell + 1$ copies of each vertex, and there is an edge between the ith copy of u to the $(i + 1)$th copy of w iff there is an edge between u and v in G. We prove that a search clears a graph iff it cuts the first layer from the last layer in the layered graph. That is, we prove that:

1. If there is no search with t guards that cuts the first layer from the last layer, then there is a simple protocol that can maintain privacy against any adversary that can eavesdrop to t channels in each round.



The Graph G_1 The Graph G_2 The Graph G_3

Figure 1: Three graphs that can be cleared with one guard.

2. If there is a search that cuts the first layer from the last layer in the layered graph, then no protocol can maintain privacy against this search. This is proved by a reduction to the impossibility of unconditional key exchange.

These results demonstrate a zero-one phenomenon: either the adversary can learn the everything or it cannot deduce any information about the secret. This zero-one phenomenon is typical to secure computation against an unbounded adversary, e.g., [4].

Inspired by this characterization, we say that an undirected path in the layered graph is contaminated if all edges in the path are unguarded; a vertex is contaminated after i rounds of the search if there is a contaminated path (through any layers) from the first layer to the copy of the vertex in layer i. That is, contamination "flows" both forwards and backwards in time.

We give a second characterization (Theorem 3.6) of when a search clears a graph based on the sets of contaminated vertices in each round of the protocol. This characterization is more useful for analyzing the possibility and impossibility of clearing graphs. Based on this second characterization, we prove an upper bound on the length of the search (Theorem 3.8): If an adversary can clear a graph while eavesdropping to at most t edges in each round, then it can clear the graph in at most 2^n rounds while eavesdropping to at most t edges in each round. We do not know if super-polynomial search length is sometimes necessary.

A search is "monotonic" if once a vertex is cleared, it will remain clear for the entire search. We explore the usefulness and limitations of a generic monotonic searches. On the positive side, we show that monotonic search is essentially optimal for directed and undirected complete graphs. A complete *directed* graph with n vertices can be cleared by $n^2/2$ guards in two rounds when n is even (by monotonic search). We prove that $n^2/2$ guards are required to clear this graph no matter how many rounds the adversary is allowed (by any search). For a complete *undirected* graph with n vertices, we show that it can be cleared by $n^2/4 + n/2$ guards in $O(\sqrt{n})$ rounds (by monotonic search). Furthermore, we prove that $n^2/4 + n/2$ guards are required to clear this graph no matter how many rounds the adversary is allowed (by any search), and $\Omega(\sqrt{n})$ rounds are required to clear the graph even if the adversary uses $n^2/4 + O(n)$ guards (by any search). In contrast, with $3n^2/8 + n/4$ guards, the complete undirected graph can be cleared in two rounds (by monotonic search).

1.2 Comparison to the Vertex Eavesdropping Game

The problem we consider is similar to the vertex eavesdropping games considered in [6]. In the vertex eavesdropping game, a mobile adversary eavesdrops to processors – it monitors their internal state, the computations they perform, and the messages they send and receive. A search is a finite sequence of subsets of vertices; a search succeeds ("clears the graph") if the adversary learns enough information to infer the secret bit b in the distributed database maintenance game. Unfortunately, the main characterization given in [6] of successful searches is incorrect. The correct characterization is similar to the edge eavesdropping games: Given a directed or undirected graph, and given a search, construct the *undirected* layered version of the graph where the number of layers is the length of the search (and with all self-loops added, i.e., an edge from each node in each non-final layer to the same node in the next layer). A search clears a graph iff it cuts the first layer from the last layer in the *undirected* layered graph.

The mistake in [6] is that they considered the *directed* layered version of the graph instead of the undirected case. In particular, the flaw is in the proof of Lemma 4 of [6], i.e., Alice cannot simulate the behavior of every node in V_s by herself. A graph demonstrating this problem is described in Appendix A. The characterization of [6] is correct if we require that each vertex is *deterministic* during the execution of the protocol.

Although, the vertex eavesdropping game and the edge eavesdropping game seem similar, there are differences between them. For example, the search of complete graphs is simple in the vertex eavesdropping game: the complete graph with n vertices can be cleared with n guards in one round, and cannot be cleared by fewer guards in any number of rounds. By contrast, the search of undirected complete graphs in the edge eavesdropping game is more complicated as it requires $\Omega(\sqrt{n})$ rounds even if near optimal number of guards are used. See Sections 4 and 5 for a detailed treatment.

In [6] it was shown that for *directed* layered graphs, super-polynomial search length is sometimes necessary: There exists a family of graphs $\{G_n\}$ such that each G_n has $O(n^2)$ vertices, however, clearing the directed layered graph of G_n requires $\Omega(2^n)$ rounds using the optimal number of guards. This should be contrasted with classic search games, in which linear number of rounds are sufficient to clear a graph with optimal number of guards [12, 1] (for background on search games on graphs [14, 11]). However, due to the problem in the characterization of [6], the above sequence of graphs *does not* imply that in vertex eavesdropping games super-polynomial search length is sometimes necessary. It is not known if super-polynomial search length is ever necessary for the vertex eavesdropping game or for the edge eavesdropping game.

1.3 Historical Background

Ostrovsky and Yung [13] considered mobile faults under the control of a Byzantine adversary to achieve general secure distributed computation against virus-like waves of attack. Defense against mobile Byzantine faults was subsequently called "proactive" security [9], and was considered in numerous papers. The classic problem of Byzantine Agreement was studied in the mobile Byzantine fault setting by Garay [7] and Buhrman et al. [2]. The distributed database game was analyzed for the vertex eavesdropping game by Franklin et al. [6]. A more elaborate and fully functional distributed storage, with all operations secure against a mobile Byzantine adversary, was treated by Garay et al. [8]. In a work done in parallel to our work, Prasad, Srinathan, and Pandu Rangan [15] considered proactive perfectly secure message transmission against an unbounded adversary.

All the above works consider faults of processors, while we consider eavesdropping to communication channels. In [3, 10, 5], the problem of using a multicast network coding to transmit information securely in the presence of an adversary which can eavesdrop to a *fixed* set of edges of bounded size was studied.

Organization. In Section 2 we present our model and some background results. In Section 3 we characterize when a graph can be cleared with t guards. In Section 4 we consider monotonic searches, discuss how they can be used to clear complete graphs, and show that for some graphs they are not optimal. Finally, in Section 5 we prove lower bounds on the number of guards needed to clear complete directed and undirected graphs.

2 Preliminaries

We consider a network described either by an undirected graph or by a directed graph. In the directed case we assume, for technical reasons, that the out-degree, $|\mathrm{OUT}(v)|$, and in-degree, $|\mathrm{IN}(v)|$, of each node v is at least 1. The network is synchronous, and protocols in the network proceed in rounds. In Round i, each vertex v does the following: (1) receives the messages sent by neighboring vertices in Round $i-1$, (2) chooses a random string r_v^i for Round i, (3) computes new messages based on the messages sent to it in Round $i-1$ and the random string r_v^i, and (4) sends the messages it computes in Round i.

We consider the distributed *database maintenance game* (*database game* for short). There is a protocol trying to maintain the secrecy of one bit b. The first stage in the game is an initialization stage in which each edge gets an initial value; there is a initialization function $I(b) = \langle m_{u,v}^0 \rangle_{\langle u,v\rangle \in E}$ that generates initial messages for the edges as a randomized function of the secret bit b. In Round i, where $i \geq 1$, the state of each vertex v is $\langle m_{u,v}^{i-1} \rangle_{u \in \mathrm{IN}(v)}, r_v^i$, that is, the messages it received in the previous round and a random string for the current round. Vertex v computes messages $\mathbf{m_v^i} = \langle m_{v,w}^i \rangle_{w \in \mathrm{OUT}(v)}$, where $\mathbf{m_v^i}$ is a function of the vertex state,[1] and sends $m_{v,w}^i$ to w. The secret can be reconstructed in each round of the protocol; there is a reconstruction function ϕ such that $\phi(\langle m_{u,v}^i \rangle_{\langle u,v\rangle \in E}, i) = b$.

In the model we define, the messages that a vertex sends in Round i depend only on the messages sent to it in Round $i-1$, and on a "fresh" random string for the round, thus, effectively each vertex forgets all information from previous rounds. The reason for this requirement is that otherwise the secrecy in the database game can be maintained in the local memory of some vertex. If we want to allow local memory, that is, remembering the history, then the adversary must be able to read it. Technically, this is done by adding self-loops in the graph. Thus, depending on the graph, we allow or disallow each vertex to remember its history. However, the adversary cannot eavesdrop to the local memory of a vertex during the momentary period of receiving the messages, computing the new messages, and sending them.

A mobile adversary is trying to learn the bit b. The adversary eavesdrops to a possibly different subset of *edges* in each round. In a directed graph, an adversary that eavesdrops to an edge $\langle u, v\rangle$ in Round i, learns the message sent by u to v in Round i. In an undirected graph, an adversary that eavesdrops to an edge $\langle u, v\rangle$ in Round i, learns two messages sent in Round i: the message sent by u to v and the message sent by v to u. The adversary cannot change, insert, or delete messages.

[1] In the undirected case, $\langle u, v\rangle$ and $\langle v, u\rangle$ are the same edge. However, $m_{u,v}$ and $m_{v,u}$ denote different messages.

A *search* – a behavior of an adversary – is a sequence of subsets of edges $W_1, W_2, \ldots, W_\ell$, where in Round i the adversary eavesdrops to the edges in W_i and learns no additional information on the messages exchanged on other edges. Similarly to other search games, if the adversary eavesdrops to an edge in Round i, then we say that it *guards* the edge in Round i.

The adversary is adaptive, it decides on W_i – the communication channels it eavesdrops in Round i – based on the messages it heard on $W_1, W_2, \ldots, W_{i-1}$ in previous rounds and on its random string r. The *view* of the adversary, after an execution, is its random input, the search $W_1, \ldots, W_\ell$ it chose to eavesdrop, and the messages it heard in this search. An unbounded adversary has not gained information on the secret bit b, if its view is equally distributed when the bit is 0 and when the bit is 1.

Definition 2.1 *The adversary does not gain information on the secret bit b in Protocol $\mathcal{P}$ if for every possible view h:*

$$\begin{aligned}&\Pr[\text{ The view of the adversary is } h \,|\, \text{The secret bit is } 0]\\ = \; &\Pr[\text{ The view of the adversary is } h \,|\, \text{The secret bit is } 1],\end{aligned}$$

where the probability is taken over $\left\{r_v^i : v \in V, 1 \leq i \leq \ell\right\}$, the random strings of the vertices, and over the random string used by the initialization function $I(b)$.

An adversary uses t guards if, for every search $W_1, W_2, \ldots, W_\ell$ that it can use, $|W_i| \leq t$ for every $1 \leq i \leq \ell$.

Definition 2.2 *A system can* maintain its secrecy *in a graph G against t guards if there is a protocol $\mathcal{P}$ for the vertices in G such that every adversary that uses t guards does not gain information on the secret bit. Otherwise, we say that t guards can clear G.*

We next describe a simple protocol, considered in [6], for the database game. In each round of the protocol we maintain the following property

$$b \;=\; \bigoplus_{\langle u,v\rangle \in E} m_{u,v}^i. \tag{1}$$

This describes the reconstruction function of the protocol. The basic step in the protocol is the simple sharing of a bit b, generating k bits $b_1, \ldots, b_k$ by randomly choosing the first $k-1$ bits independently such that each bit is uniformly distributed, and setting $b_k \leftarrow b \oplus \bigoplus_{1 \leq i \leq k-1} b_i$. In the initialization stage, Protocol $\mathcal{P}_{\rm xor}$ generates the messages $\langle m_{u,v}^0 \rangle_{\langle u,v\rangle \in E}$ as the sharing of the secret bit b. In Round i of Protocol $\mathcal{P}_{\rm xor}$, each vertex computes the bit $b_v^i \leftarrow \bigoplus_{u \in \mathrm{IN}(v)} m_{u,v}^{i-1}$, and shares b_v^i generating the bits $\langle m_{v,w}^i \rangle_{w \in \mathrm{OUT}(v)}$, that is,

$$\bigoplus_{u \in \mathrm{IN}(v)} m_{u,v}^{i-1} \;=\; b_v^i \;=\; \bigoplus_{w \in \mathrm{OUT}(v)} m_{v,w}^i. \tag{2}$$

As we assume that each vertex has at least one in-going edge and at least one out-going edge, this process is possible. Clearly, the reconstruction described in (1) is correct in the initialization stage. A simple calculation shows, using induction, that the reconstruction described in (1) is correct in each round of the protocol. In the next section we show that this simple protocol is "universal": if there exists a protocol that can maintain secrecy against t guards, then Protocol $\mathcal{P}_{\rm xor}$ can maintain secrecy against t guards.

3 Characterization Theorems for Clearing Graphs

We give two theorems that characterize graphs that can be cleared with t guards. To understand the evolution of the clearing process throughout the rounds of the protocol, we define a layered graph version of the communication graph. In this graph there are two vertices Source and Target that are added for technical reasons.

Definition 3.1 *Given a directed or an undirected graph $G = \langle V, E\rangle$ and an index ℓ, we construct an undirected layered graph $L(G,\ell) = \langle V^\ell, E^\ell\rangle$ as follows. The vertices of $L(G,\ell)$ are $V^\ell \stackrel{def}{=} (V \times \{1,\dots,\ell+1\}) \cup \{\text{Source}, \text{Target}\}$. The edges of $L(G,\ell)$ are*

$$\begin{aligned} E^\ell \stackrel{def}{=} & \{\langle (u,i),(v,i+1)\rangle : \langle u,v\rangle \in E, 1 \le i \le \ell\} \\ & \cup \ \{\langle \text{Source},(v,1)\rangle : v \in V\} \ \cup \ \{\langle (v,\ell+1), \text{Target}\rangle : v \in V\}\,. \end{aligned}$$

Given a search $W_1, W_2, \dots, W_\ell$, we say that an edge $\langle (u,i),(v,i+1)\rangle$ in $L(G,\ell)$ is guarded when G is a directed graph if $\langle u,v\rangle \in W_i$. We say that an edge $\langle (u,i),(v,i+1)\rangle$ in $L(G,\ell)$ is guarded when G is an undirected graph if $\langle u,v\rangle \in W_i$ or $\langle v,u\rangle \in W_i$. If an edge is not guarded, then we say that the edge is unguarded. An undirected path in $L(G,\ell)$ is *contaminated* if all edges in the path are unguarded. Note that this path can go forwards and backwards in the layers. A search $W_1, W_2, \dots, W_\ell$ of length ℓ *cuts* the undirected layered graph $L(G,\ell)$ if there is no contaminated path in $L(G,\ell)$ from Source to Target. For example, consider Graph G_1 shown in Figure 1. In Figure 2 we describe its layered graph $L(G_1,3)$ and a search that does not cut it, and in Figure 3 we describe its layered graph $L(G_1,4)$ and a search that cuts it.

3.1 First Characterization Theorem

Theorem 3.2 (First Characterization Theorem) *Let G be a graph. A system can maintain its secrecy in the graph G against t guards iff for every $\ell \in \mathbb{N}$, every search $W_1, W_2, \dots, W_\ell$ with t guards does not cut $L(G,\ell)$.*

In light of Theorem 3.2, if a search cuts the undirected layered graph $L(G,\ell)$, we may say that the search *clears* G. The theorem is implied by the following two lemmas.

Lemma 3.3 *Let G be a graph, and $W_1, W_2, \dots, W_\ell$ be a search that cuts $L(G,\ell)$. Then, for every protocol $\mathcal{P}$, the adversary that eavesdrops to W_i in Round i, for $1 \le i \le \ell$, learns the secret after at most ℓ rounds.*

Proof: Fix any protocol $\mathcal{P}$. We assume, for sake of contradiction, that the adversary that eavesdrops to W_i in Round i, for $1 \le i \le \ell$, does not learn the secret in the first ℓ rounds, and construct an information-theoretic secure protocol in which two parties, Alice and Bob, can exchange a secret key on a public channel (without any prior secret information), which is impossible by the fundamental result of Shannon [16].

We next define two sets with respect to $W_1, W_2, \dots, W_\ell$.

$$R \stackrel{\text{def}}{=} \{\text{Source}\} \cup \{(v,i) : \text{ there is a contaminated path from Source to } (v,i)\}\,,$$

and $B \stackrel{\text{def}}{=} V^\ell \setminus R$. Notice that all edges that connect vertices from R to B are guarded.

Informally, to exchange a key Alice and Bob execute $\mathcal{P}$, where Alice simulates the vertices in R and Bob simulates the vertices in B, and the messages that should be sent on guarded edges, that is, the messages the adversary hears, are broadcasted on the public channel. Formally, to transmit a bit b, Alice uses the initialization function $I(b)$ to generate messages $\langle m^0_{u,v}\rangle_{\langle u,v\rangle\in E}$. Now, Alice and Bob simulate $\mathcal{P}$ round by round. In the ith round, Alice simulates the vertices in $R_i \stackrel{\text{def}}{=} \{v \in V : (v,i) \in R\}$ and Bob simulates all other vertices, namely $B_i \stackrel{\text{def}}{=} V \setminus R_i$. We will show that the simulation maintains the following property:

Property 1 *Each party knows all messages sent in Round $i-1$ to the vertices that it simulates in Round i.*

Now, Alice (respectively, Bob) chooses random strings r^i_v for every $v \in R_i$ (respectively, $v \in B_i$), computes the messages v sends in $\mathcal{P}$, broadcasts on the public channel all the message that are sent on guarded edges, and remembers all other messages.

Property 1 is maintained for Alice (respectively, for Bob), as all edges from B_{i-1} to R_i (respectively, all edges from R_{i-1} to B_i) are guarded, and, therefore, the messages sent on them are broadcasted on the public channel. This implies that the key-exchange protocol can proceed. On one hand, there is no contaminated path in $L(G,\ell)$, and after the ℓth round of the simulation all vertices in G are in $B_{\ell+1}$. So, Bob can compute the reconstruction function $\phi(\langle m^\ell_{u,v}\rangle_{\langle u,v\rangle\in E},\ell)$ and learn the message sent by Alice. On the other hand, the view of Eve after the key exchange protocol is exactly the view of the adversary that eavesdrops to $W_1, W_2, \ldots, W_\ell$ in $\mathcal{P}$, so Eve learns nothing about b. This is a contradiction to the fundamental result of Shannon [16] that there no unconditionally secure key exchange protocol that only uses a public channel. Thus, in the original protocol $\mathcal{P}$, the adversary can learn the secret. $\square$

Notice that in Lemma 3.3 the adversary is deterministic and non-adaptive as it deterministically chooses the search it uses before the execution of the protocol. Furthermore, the adversary exactly learns the secret information.

Lemma 3.4 *Let $\mathcal{P}_{\text{xor}}$ be the XOR protocol, and assume that for some ℓ there is no search with t guards that cuts $L(G,\ell)$. Then, any adversary that uses t guards does not gain information on the secret in the first ℓ rounds of $\mathcal{P}_{\text{xor}}$.*

Proof: To understand the idea of the proof, first consider a deterministic adversary which chooses a search $W_1, W_2, \ldots, W_\ell$ with t guards before the execution of the protocol (that is, its choice of W_i does not depend on the messages it heard in previous rounds). Since the search does not clear the graph, there is a contaminated path in the layered graph from SOURCE to TARGET. The adversary cannot learn the secret bit b, since the value of the secret bit b can be flipped by flipping the values of the messages sent on a contaminated path. This is a valid execution of the protocol in which the adversary sees the same view, however with the secret $\overline{b}$.

To consider a randomized, adaptive adversary, fix any view h for the adversary, that is, fix a random string r of the adversary, a search $W_1, W_2, \ldots, W_\ell$ with t guards, and the messages sent on the edges of the search. To prove the lemma, we show that there is a one-to-one and onto function from possible executions of $\mathcal{P}_{\text{xor}}$ when the view is h and the secret bit is 0 to possible executions of $\mathcal{P}_{\text{xor}}$ when the view is h and the secret bit is 1. Thus, the number of these executions is the same for both values of the secret, and, as every possible execution of protocol $\mathcal{P}_{\text{xor}}$ has the same probability, the probability of the view is the same for both values of the secret.

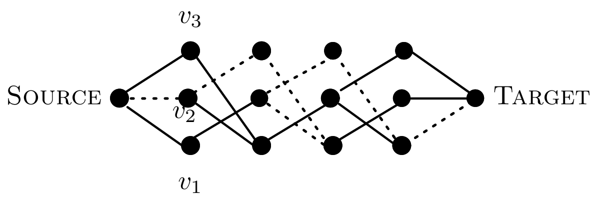


Figure 2: The contaminated path in $L(G_1, 3)$ with respect to the search that guards the edge $\langle v_1, v_2 \rangle$ for three rounds. The edges in the path are the dotted edges.

Consider any execution of $\mathcal{P}_{\text{xor}}$ when the view is h and the secret bit is 0. There must be a contaminated path from SOURCE to TARGET in $L(G, \ell)$ with respect to $W_1, W_2, \ldots, W_\ell$. Consider the lexicographically first simple contaminated path in $L(G, \ell)$. We map the execution with secret 0 to the following execution of the protocol $\mathcal{P}_{\text{xor}}$ with the secret 1: We flip the values sent of the path as follows.

- For $\langle \text{SOURCE}, (v, 1) \rangle$, the first edge in the path, flip the initial value $m^0_{u,v}$ for the first $u \in \text{IN}(v)$.
- For every "forward" edge $\langle (u, i), (v, i+1) \rangle$ in the path, flip the message sent by u to v in Round i.
- For every "backward" edge $\langle (u, i), (v, i-1) \rangle$ in the path, flip the message sent by v to u in Round $i - 1$.

We claim that this is a legal execution of $\mathcal{P}_{\text{xor}}$, that is, for every v and every i, Equation (2) holds – the exclusive-or of the messages v receives in Round $i - 1$ is equal to the exclusive-or of the messages v sends in Round i. This is true since the path is simple, and, therefore, the mapping flipped the values of two edges for every vertex in the path (and changed no messages sent on edges not in the path). Since the mapping flipped the value of exactly one initial message, the value of the secret in the new execution has changed to 1, thus, this is indeed an execution with secret 1.

As the mapping flipped the values only on unguarded edges, in each round of the protocol, the adversary sees the same messages, thus, it cannot notice this change, and it continues to eavesdrop to the same search. Finally, this transformation is one-to-one and onto since if we apply this transformation twice, then the result is the original execution. □

Example 3.5 Consider Graph G_1 described in Figure 1 in the Introduction. The search that guards the edge $\langle v_1, v_2 \rangle$ for three rounds does not clear the graph as SOURCE, $(v_2, 1)$, $(v_3, 2)$, $(v_1, 3)$, $(v_2, 2)$, $(v_3, 3)$, $(v_1, 4)$, TARGET is a contaminated path from SOURCE to TARGET in $L(G_1, 3)$. Notice that this path goes forwards and backwards in the layer graph. This path is described in Figure 2. There is no contaminated path in $L(G_1, 3)$ that only goes forward; this search illustrates the importance of "backward" edges in the layered graph. Nevertheless, Graph G_1 can be cleared with one guard in 4 rounds as follows: In Round 1 guard $\langle v_2, v_3 \rangle$, in Round 2 guard $\langle v_2, v_1 \rangle$, in Round 3 guard $\langle v_2, v_3 \rangle$, and in Round 4 guard $\langle v_1, v_2 \rangle$. This layered graph with this search is described in Figure 3.

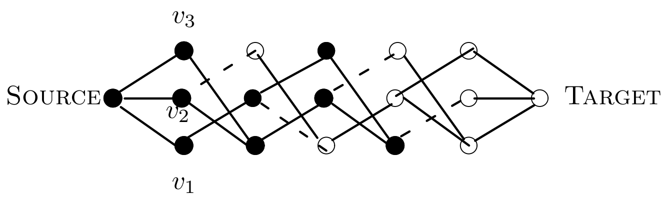


Figure 3: A search that clears G_1 with one guard. We describe the layered graph $L(G_1, 4)$; the guarded edges are the dashed edges and the contaminated vertices are the black vertices.

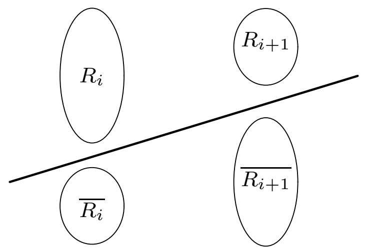


Figure 4: A description of the ith layer of the cut in $L(G, \ell)$ for a search. The sets R_i and R_{i+1} are the sets of the vertices reachable by an unguarded paths in layers i and $i+1$, respectively. The edges in the cut are the edges between R_i and $\overline{R_{i+1}}$ and the edges between $\overline{R_i}$ and R_{i+1}.

3.2 Second Characterization Theorem

Recall that a *cut* in an undirected graph $H = \langle V, E \rangle$ is a set of edges defined by a set $R \subset V$ containing all edges between R and $\overline{R}$. Theorem 3.2 implies that a search clears a graph G iff it induces a cut in $L(G, \ell)$ such that all edges in the cut are guarded. That is, there is a search that clears a graph iff there is a cut in the graph $L(G, \ell)$ that, for every i, contains at most t edges between layer i and layer $i + 1$. This is formalized in the next theorem, and illustrated in Figure 4.

Theorem 3.6 (Second Characterization Theorem) *Let G be a graph. The graph G can by cleared by t guards iff there is some $\ell \in \mathbb{N}$ and a sequence of subsets of vertices $R_1, \ldots, R_{\ell+1}$ (that is, $R_i \subseteq V$ for $1 \leq i \leq \ell + 1$) such that*

1. *$R_1 = V$, $R_{\ell+1} = \emptyset$, and*
2. *for every $1 \leq i \leq \ell$ the set $\left(R_i \times \overline{R_{i+1}}\right) \bigcup \left(\overline{R_i} \times R_{i+1}\right)$ contains at most t edges of G.*

Proof: By Theorem 3.2 it suffices to prove that such sequence of sets $R_1, \ldots, R_{\ell+1}$ exists iff there exists a search with t guards that cuts $L(G, \ell)$.

First, we assume that such sequence of sets $R_1, \ldots, R_{\ell+1}$ exists. We define the search $W_1, \ldots, W_\ell$ with t guards, where W_i contains the edges in E that are in $\left(R_i \times \overline{R_{i+1}}\right) \bigcup \left(\overline{R_i} \times R_{i+1}\right)$. We claim that this search cuts $L(G, \ell)$, that is, every path from Source to Target in $L(G, \ell)$ contains a guarded edge. Define

$$R \stackrel{\text{def}}{=} \{\text{Source}\} \cup \{(v, i) : 1 \leq i \leq \ell, v \in R_i\}.$$

The edges in the cut between R and $V^\ell \setminus R$ in $L(G,\ell)$ are exactly the edges guarded by our search. Every path from Source to Target in $L(G,\ell)$ passes through this cut, thus, the path contains a guarded edge and is not contaminated.

Now assume that there is a search $W_1,\ldots,W_\ell$ with t guards that cuts $L(G,\ell)$. For every i, where $1 \le i \le \ell+1$, define

$$R_i \stackrel{\text{def}}{=} \{v : \text{ there exists a contaminated path from Source to } (v,i) \text{ in } L(G,\ell)\}\,.$$

We say that R_i is the set of contaminated vertices in Round i. First, $R_1 = V$, since all edges from Source to the first layer are unguarded. Second, $R_{\ell+1} = \emptyset$, since all edges from the last layer to Target are unguarded and there is no contaminated path from Source to Target. We need to prove that, for every i, the set $\left(R_i \times \overline{R_{i+1}}\right) \bigcup \left(\overline{R_i} \times R_{i+1}\right)$ contains at most t edges of G. Recall that in each round of the protocol, at most t edges are guarded, thus, it suffices to prove that the edges of E in $\left(R_i \times \overline{R_{i+1}}\right) \bigcup \left(\overline{R_i} \times R_{i+1}\right)$ must be guarded in Round i:

- For every $v \in R_i$ and $w \in \overline{R_{i+1}}$, the edge $\langle v,w\rangle$ (if exists) must be guarded in Round i, otherwise the contaminated path ending at (v,i) together with $\langle (v,i),(w,i+1)\rangle$ is a contaminated path ending at $(w,i+1)$.
- For every $v \in \overline{R_i}$ and $w \in R_{i+1}$, the edge $\langle v,w\rangle$ (if exists) must be guarded in Round i, otherwise the contaminated path ending at $(w,i+1)$ together with $\langle (w,i+1),(v,i)\rangle$ is a contaminated path ending at (v,i).

To conclude the second direction, given a search with t guards that cuts $L(G,\ell)$, we showed that the sets of contaminated vertices satisfy the condition of the theorem. $\square$

Example 3.7 Consider a directed cycle with n vertices, i.e., the graph $G = \langle V,E\rangle$ where $V = \{v_0,\ldots,v_{n-1}\}$ and $E = \left\{\langle v_i, v_{(i+1) \bmod n}\rangle : 0 \le i \le n-1\right\}$. This graph can be cleared by one guard sitting on the same edge for $n-1$ rounds. For concreteness, assume that $W_i = \{\langle v_{n-1}, v_0\rangle\}$ for $i = 1,\ldots,n$. Define $R_i = \{v_{i-1},\ldots,v_{n-1}\}$, for $1 \le i \le n+1$. Clearly, $R_1 = V$ and $R_{n+1} = \emptyset$. For $1 \le i \le n-1$, the only edge from the set $R_i = \{v_{i-1},\ldots,v_{n-1}\}$ to the set $\overline{R_{i+1}} = \{v_0,\ldots,v_{i-1}\}$ is $\langle v_{n-1}, v_0\rangle$ and there are no edges from $\overline{R_i} = \{v_0,\ldots,v_{i-2}\}$ to $R_{i+1} = \{v_i,\ldots,v_{n-1}\}$. Furthermore, $\langle v_{n-1}, v_0\rangle$ is the only edge in $\left(R_n \times \overline{R_{n+1}}\right) \bigcup \left(\overline{R_n} \times R_{n+1}\right)$. It can be checked that the sets $R_1,\ldots,R_{n+1}$ are exactly the sets of contaminated vertices in the above search.

As a consequence, we prove that 2^n rounds are sufficient to clear a graph with minimal number of guards.

Theorem 3.8 *If a graph G can be cleared with t guards, then it can be cleared with t guards in at most 2^n rounds.*

Proof: By Theorem 3.6, there is a sequence of subsets $R_1,\ldots,R_{\ell+1}$ such that $R_1 = V$, $R_{\ell+1} = \emptyset$, and $\left(R_i \times \overline{R_{i+1}}\right) \bigcup \left(\overline{R_i} \times R_{i+1}\right)$ contains at most t edges from E. Consider a shortest sequence satisfying these conditions. We claim that there are no indices $i_1 < i_2$ such that $R_{i_1} = R_{i_2}$, otherwise, $R_{i_1-1} \times \overline{R_{i_2}} = R_{i_1-1} \times \overline{R_{i_1}}$ and $\overline{R_{i_1-1}} \times R_{i_2} = \overline{R_{i_1-1}} \times R_{i_1}$, and thus their union contains at most t edges in E and $R_1,\ldots,R_{i_1-1}, R_{i_2},\ldots,R_{\ell+1}$ is a shorter sequence which satisfies the above conditions. Therefore, each set R_i can appear at most once in the search, and the length of the search is at most 2^n. $\square$

Finally, we observe that the problem of determining if a graph can be cleared with t guards is in $\mathcal{PSPACE}$. We do not know if this problem can be solved efficiently.

A Monotonic Search

$R_1 \leftarrow V$; $i \leftarrow 1$
While $R_i \neq \emptyset$ do:
  Choose a set $A_i \subseteq R_i$ and set $R_{i+1} \leftarrow R_i \setminus A_i$
  Guard the following set of edges W_i:
    $W_i = \left\{\langle u,v\rangle : u \in R_i, v \in \overline{R_{i+1}}\right\} \cup \left\{\langle u,v\rangle : u \in \overline{R_i}, v \in R_{i+1}\right\}$
  $i \leftarrow i+1$.

Figure 5: A monotonic search strategy for clearing a graph.

Corollary 3.9 *The decision problem if a graph can be cleared with t guards is in $\mathcal{PSPACE}$.*

Proof: To prove the corollary, we describe a non-deterministic algorithm that uses polynomial space. The algorithm guesses the sets $R_1, \ldots, R_{\ell+1}$, such that $R_1 = V$, $R_{\ell+1} = \emptyset$, and verifies that the set $\left(R_i \times \overline{R_{i+1}}\right) \bigcup \left(\overline{R_i} \times R_{i+1}\right)$ contains at most t edges in G for every i. Since the search can be long, the algorithm guesses the sets R_i round by round; in Round i the algorithm only remembers R_i and R_{i+1}. By Theorem 3.6, the algorithm can guess such sets $R_1, \ldots, R_{\ell+1}$ iff t guards can clear G. □

4 A Monotonic Search Strategy for Clearing Graphs

In this section we consider a special case of searches that clear a graph called monotone searches. By Theorem 3.6, to specify a strategy for the adversary, we can specify the contaminated vertices in each round.

Definition 4.1 (Monotone Searches) *We say that a search is* monotonic *if $R_\ell \subset R_{\ell-1} \subset \cdots \subset R_2 \subset R_1$, that is, once a vertex is cleared, it will not become contaminated later.*

In Figure 5, we formally describe monotonic searches. The advantage of monotonic searches is that they are short; there can be at most n rounds until the adversary clears the graph. However, they are not necessarily optimal, as they can require more guards than general searches (see Section 4.3). In this section we present examples of a monotonic searches that clear directed and undirected complete graphs. As complete graphs are symmetric, it suffices to specify the size of the each set R_i without specifying the exact set of vertices.

4.1 Monotonic Search in Complete Directed Graphs

A complete directed graph, denoted C_n, is a graph with all the possible n^2 edges (including self loops).

Claim 4.2 *When n is even, $n^2/2$ guards can clear the complete directed graph with n vertices C_n in two rounds. When n is odd, $(n^2+1)/2$ guards can clear C_n in three rounds.*

Proof: To clear the graph when n is even, partition the n vertices in C_n to two disjoint sets V_1 and V_2 of size $n/2$ each. In the first round, guard all the $n^2/2$ edges from V to V_1. In the second round, guard all the $n^2/2$ edges from V_2 to V. In this case, $R_1 = V$, $R_2 = V_2$, and $R_3 = \emptyset$.

To clear the graph when n is odd, partition the n vertices in C_n to three disjoint sets: V_1 and V_2 of size $(n-1)/2$ each, and a single vertex v. In the first round, guard all the edges from V to V_1. There are $n|V_1| = n(n-1)/2 < (n^2+1)/2$ such edges. In the second round, guard all edges from V_1 to V_2 and all edges from $V_2 \cup \{v\}$ to $V_1 \cup \{v\}$. There are $(n-1)^2/4 + (n+1)^2/4 = (n^2+1)/2$ such edges. In the third round, guard all the edges from V_2 to V. There are $n|V_2| = n(n-1)/2 < (n^2+1)/2$ such edges. In this case, $R_1 = V$, $R_2 = V_2 \cup \{v\}$, $R_3 = V_2$, and $R_4 = \emptyset$. □

In Section 5, we show that the number of guards in these searches is optimal by showing a matching lower bound.

4.2 Monotonic Search in Complete Undirected Graphs

A complete undirected graph, denoted U_n, is a graph with all the possible $\binom{n+1}{2}$ edges (including self loops). To simplify calculations, in this section n is even. We first show that $3n^2/8+n/4$ guards can clear U_n in two rounds. To clear the graph, partition the n vertices in U_n to two disjoint sets V_1 and V_2 of size $n/2$ each. In the first round, guard all the edges with at least one endpoint in V_1. There are $\binom{n/2+1}{2} + n^2/4 = 3n^2/8 + n/4$ such edges. In the second round, guard all the edges with at least one endpoint in V_2. Again, there are $3n^2/8 + n/4$ such edges.

Claim 4.3 *When n is even, $n^2/4 + n/2$ guards can clear the complete undirected graph with n vertices U_n in $O(\sqrt{n})$ rounds.*

Proof: We first describe a search of length n in U_n using $n^2/4+n/2$ guards. Let $V = \{v_1, \dots, v_n\}$ be the vertices of the graph. In the ith round of the search we choose $R_i \setminus R_{i+1} \stackrel{\text{def}}{=} A_i = \{v_i\}$ and $R_i = \{v_i, \dots, v_n\}$. The guarded edges are

$$\{\langle v_i, v_j \rangle : 1 \le j \le n\} \cup \{\langle v_j, v_k \rangle : 1 \le j \le i-1, i+1 \le k \le n\}.$$

The number of guarded edges in Round i is, thus, $n + (i-1)(n-i) = i(n-i+1)$. The expression is maximized when $i = n/2$, and is $n^2/4 + n/2$. Thus, $n^2/4 + n/2$ guards are sufficient to clear a complete undirected graph in n rounds. In Section 5, we show that this is optimal by showing a matching lower bound.

We next show that, with the same number of guards as in the previous search, the adversary can clear the complete undirected graph in $O(\sqrt{n})$ rounds. (In Section 5.3, we show that if the adversary uses $n^2/4 + O(n)$ guards, then $\Omega(\sqrt{n})$ rounds are necessary to clear the graph.) The idea to reduce the number of rounds is that when $|R_i|$ is small or big, the adversary can take bigger sets A_i than the singletons considered in the previous search.

Let $R_1, \dots, R_{\ell+1}$ be sets defining a monotonic search of U_n, let $A_i \stackrel{\text{def}}{=} R_i \setminus R_{i+1}$, and $S_i \stackrel{\text{def}}{=} \overline{R_i}$. Notice that $R_i = R_{i+1} \cup A_i$, $S_{i+1} = S_i \cup A_i$, and $S_i \cup A_i \cup R_{i+1} = S_i \cup R_i = V$. The edges guarded in Round i of the monotonic search are

$$\begin{aligned} &\{\langle u,v \rangle : u \in R_i, v \in S_{i+1}\} \cup \{\langle u,v \rangle : u \in S_i, v \in R_{i+1}\} \\ = \quad &\{\langle u,v \rangle : u \in (R_{i+1} \cup A_i), v \in (S_i \cup A_i)\} \cup \{\langle u,v \rangle : v \in S_i, u \in R_{i+1}\} \\ = \quad &\{\langle u,v \rangle : u \in A_i, v \in V\} \cup \{\langle u,v \rangle : v \in S_i, u \in R_{i+1}\}. \end{aligned}$$

Thus, the number of edges guarded in Round i is at most

$$|A_i||V| + |S_i||R_{i+1}| \quad = \quad |A_i|n + |S_i|(n - |S_i| - |A_i|). \tag{3}$$

In each round, we want to choose the largest set A_i such that the number of guards, as bounded in (3), does not exceed $n^2/4 + n/2$. In the first round, $|S_1| = 0$, thus, the requirement is $|A_1|n \leq n^2/4 + n/2$, that is, we can take $|A_1| \approx n/4$. In the second round $|S_1| \approx n/4$, thus, the requirement is $|A_2|n + \frac{n}{4}(\frac{3n}{4} - |A_2|) \leq n^2/4 + n/2$, that is, we can take $|A_2| \approx \frac{n}{12}$. Similar calculations show that in the ith round we can take $|A_i| \approx \frac{n}{2i(i+1)}$, and, in this case, $|R_i| \approx \frac{n}{2}(1 + \frac{1}{i})$. After $O(\sqrt{n})$ rounds, $|R_i| \approx \frac{n}{2} - \sqrt{n}$. Then, with choosing A_i as a singleton for $O(\sqrt{n})$ additional rounds, the adversary gets $|R_i| = n/2$. Finally, by (3), with additional $O(\sqrt{n})$ rounds the adversary can get $|R_i| = 0$, by using a "reverse" search strategy. That is, if the adversary used sets A_i of size $a_1, a_2, \ldots, a_{O(\sqrt{n})}$ to clear the first $n/2$ vertices, then by using sets A_i of size $a_{O(\sqrt{n})}, \ldots, a_2, a_1$ the adversary clears the last $n/2$ vertices. As $|A_i|$ has to be an integer, there are some technical details to consider. The exact details are omitted. □

4.3 Monotonic Searches are Not Optimal

We show that monotonic searches can require more guards than non-monotonic searches. This phenomenon is also true for the vertex eavesdropping game [6], but not for the classic search games on graphs [12, 1].

In Figure 6 we describe an example of a simple directed graph, Graph $G_0 = \langle V_0, E_0 \rangle$ where $V_0 = \{v_1, v_2\}$ and $E_0 = \{\langle v_1, v_2 \rangle, \langle v_2, v_1 \rangle, \langle v_2, v_2 \rangle\}$. This graph can be cleared with one guard using a non monotonic search:

- Guard $\langle v_2, v_1 \rangle$ in the first round,
- Guard $\langle v_2, v_2 \rangle$ in the second round,
- Guard $\langle v_1, v_2 \rangle$ in the third round.

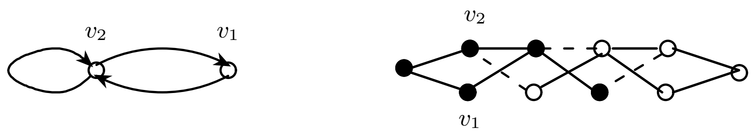
In Figure 6 we describe the layered graph $L(G_0, 3)$ and the above search. This is a non-monotonic search since v_1 is cleared in the first round and becomes contaminated in the second round.

We next claim that every monotonic search that clears G_0 uses at least two guards. In every search that clears G_0 with one guard, the first vertex that must be cleared is v_1. The only way to keep v_1 clear with one guard is to keep the guard on the edge $\langle v_2, v_1 \rangle$, thus, not clearing v_2 and not clearing G_0.

We next describe how to clear Graphs G_2 and G_3, described in Figure 1 in the Introduction, with one guard using non-monotonic searches. To clear Graph G_2, guard $\langle v_2, v_3 \rangle$ for two rounds, guard $\langle v_1, v_1 \rangle$ in the 3rd round, guard $\langle v_2, v_2 \rangle$ in the 4th round, and guard $\langle v_4, v_1 \rangle$ for the last two rounds.[2]

Notice that G_3 contains the graph G_2 and, in addition, a path with seven edges. To clear Graph G_3, guard $\langle v_2, v_5 \rangle$ for six rounds (this clears the path), then, in six rounds, clear the G_2 part of G_3 using the search described in the previous paragraph. In these rounds the path becomes contaminated, so we clear it again by guarding $\langle v_{10}, v_1 \rangle$ for 6 rounds. It can be checked that this search clears G_3.

[2] To illustrate the importance of "backwards" edges in the layered graph, note that modifying this search by guarding $\langle v_3, v_4 \rangle$ immediately after $\langle v_1, v_1 \rangle$ and before $\langle v_2, v_2 \rangle$ yields a seven-round search that cuts all of the simple "forward" paths in $L(G_2, 7)$ but fails to clear G_2.



The Graph G_0 The layered graph $L(G_0, 3)$

Figure 6: The graph G_0 and its layer graph. The guarded edges in the layer graph are the dashed edges, and the contaminated vertices are the black vertices in the layered graph.

5 Lower Bounds on Clearing a Complete graph

We show that $\lceil n^2/2 \rceil$ guards are necessary to clear C_n (no matter how many rounds the adversary uses to clear the graph). Furthermore, we show that $n^2/4 + n/2$ guards are necessary to clear a complete undirected graph and $\Omega(\sqrt{n})$ rounds are necessary to clear this graph with $n^2/4 + O(n)$ guards.

5.1 Lower Bounds on Guards to Clear a Directed Complete Graph

Theorem 5.1 *An adversary needs at least $\lceil n^2/2 \rceil$ guards to clear a complete directed graph.*

Proof: Assume that there is a search that clears the graph C_n with t guards. We claim that $t \geq n^2/2$. Let $R_1, \ldots, R_{\ell+1}$ be a sequence satisfying the conditions of Theorem 3.6. In particular, $|R_1| = n$ and $|R_{\ell+1}| = 0$. Let i be the minimal index such that $|R_{i+1}| < n/2$. Thus, $|R_i| \geq n/2$. We claim that the number of edges guarded in Round i is at least $n^2/2$. In C_n, all edges in $\left(R_i \times \overline{R_{i+1}}\right) \bigcup \left(\overline{R_i} \times R_{i+1}\right)$ exist, and the sets $R_i \times \overline{R_{i+1}}$ and $\overline{R_i} \times R_{i+1}$ are disjoint. Thus, the number of edges is exactly

$$|R_i||\overline{R_{i+1}}| + |\overline{R_i}||R_{i+1}| \quad = \quad |R_i|(n - |R_{i+1}|) + |R_{i+1}|(n - |R_i|) \tag{4}$$

Since $|R_{i+1}| < n/2$, this expression is an increasing function of $|R_i|$, thus, since $|R_i| \geq n/2$, it is at least $n/2(n - 2|R_{i+1}|) + |R_{i+1}|n = n^2/2$. As the number of guards is an integer, the theorem follows. □

5.2 Lower Bounds on Guards to Clear an Undirected Complete Graph

The following theorems provide lower bounds on the number of guards and rounds needed to clear complete undirected graphs.

Theorem 5.2 *An adversary needs at least $n^2/4 + n/2$ guards to clear a complete undirected graph.*

Proof: The proof of the lower bound for the complete undirected graphs is similar to the proof of the directed case. Indeed, a slightly weaker lower bound of $n^2/4$ can be obtained by a simple reduction to the directed case. However, in the undirected case, exactly counting the number of edges in the cut is somewhat more complicated. Besides proving the tight lower bound, the proof is used in Section 5.3 to prove a lower bound of $\Omega(\sqrt{n})$ on the number of rounds if near optimal number of guards is used. Assume that there is a search that clears the graph U_n with t guards. We claim that $t \geq n^4/2 + n/2$. Let $R_1, \ldots, R_{\ell+1}$ be a sequence satisfying the conditions of Theorem 3.6.

In the directed case, the number of the edges in $\left(R_i \times \overline{R_{i+1}}\right) \bigcup \left(\overline{R_i} \times R_{i+1}\right)$ was calculated in (4). In the undirected case, we might count each edge twice since we might count $\langle v, w\rangle$ and $\langle w, v\rangle$ as different edges. However, every edge is counted at most twice. Thus, in the undirected case, the number of edges in the above two sets is at least

$$\frac{|R_i|(n - 2|R_{i+1}|) + |R_{i+1}|n}{2}. \tag{5}$$

Similar to the directed case, this implies that the number of edges guarded is at least $n^2/4$.

We next improve this lower bound to $n^2/4 + n/2$. Towards this goal, we show that at least n edges are not counted twice.

Claim 5.3 *If $v \in R_i \vartriangle R_{i+1}$, then the edge $\langle v, v\rangle$ is counted exactly once in (5). If $v \in R_i \vartriangle R_{i+1}$ and $w \in \overline{R_i \vartriangle R_{i+1}}$, then the edge $\langle v, w\rangle$ is counted exactly once in (5).*

Proof: All expressions in (5) and in the claim are symmetric with respect to R_i and R_{i+1}, thus, we can assume without loss of generality that $v \in R_i \setminus R_{i+1}$.

First, $\langle v, v\rangle \in R_i \times \overline{R_{i+1}}$ (that is, $\langle v, v\rangle$ is counted at least once) and $\langle v, v\rangle \notin \overline{R_i} \times R_{i+1}$ (that is, $\langle v, v\rangle$ is not counted more than once).

Second, consider the edge $\langle v, w\rangle$. There are two cases:

Vertex w is in $\overline{R_i \cup R_{i+1}}$. On one hand, $\langle v, w\rangle \in R_i \times \overline{R_{i+1}}$. On the other hand, $\langle v, w\rangle \notin \overline{R_i} \times R_{i+1}$ and $\langle w, v\rangle \notin (R_i \times \overline{R_{i+1}}) \bigcup (\overline{R_i} \times R_{i+1})$. That is, in this case, the edge $\langle v, w\rangle$ is counted exactly once.

Vertex w is in $R_i \cap R_{i+1}$. On one hand, $\langle w, v\rangle \in R_i \times \overline{R_{i+1}}$. On the other hand, $\langle v, w\rangle \notin (R_i \times \overline{R_{i+1}}) \bigcup (\overline{R_i} \times R_{i+1})$ and $\langle w, v\rangle \notin \overline{R_i} \times R_{i+1}$. That is, also in this case, the edge $\langle v, w\rangle$ is counted exactly once.

■ (of Claim 5.3)

The contribution of the edges described in Claim 5.3, in addition to contribution in (5), is at least

$$\frac{|R_i \vartriangle R_{i+1}|(1 + |\overline{R_i \vartriangle R_{i+1}}|)}{2} = \frac{|R_i \vartriangle R_{i+1}|(1 + n - |R_i \vartriangle R_{i+1}|)}{2}. \tag{6}$$

All together, the number of guards in Round i must be at least

$$\frac{|R_i|(n - 2|R_{i+1}|) + |R_{i+1}|n}{2} + \frac{|R_i \vartriangle R_{i+1}|(1 + n - |R_i \vartriangle R_{i+1}|)}{2}. \tag{7}$$

At the beginning of the protocol, $|R_1| = n$ and at the end of the protocol $|R_\ell| = 0$. Let i be the minimal index such that $|R_{i+1}| < n/2$. Thus, $|R_i| \geq n/2$, and $\frac{|R_i|(n-2|R_{i+1}|)+|R_{i+1}|n}{2} \geq n^2/4$.

Denote $r \stackrel{\text{def}}{=} |R_i \vartriangle R_{i+1}|$. To complete the proof of the lower bound, we need to prove that $|R_i \vartriangle R_{i+1}|(1 + n - |R_i \vartriangle R_{i+1}|) = r(1 + n - r) \geq n$. Notice that $1 \leq r \leq n$ (since $|R_i \vartriangle R_{i+1}| \geq |R_i \setminus R_{i+1}| \geq |R_i| - |R_{i+1}| \geq 1$). The function $r(1 + n - r)$, when $1 \leq r \leq n$, is minimized in the borders when $r \in \{1, n\}$. Thus, it is at least n, completing the proof that $n^2/4 + n/2$ guards are necessary. □

5.3 Lower Bounds on Rounds to Clear an Undirected Complete Graph

We next present a lower bound on the number of rounds needed to clear the complete undirected graph with near optimal number of guards. Namely, we prove that if $n^2/4 + \gamma n$ guards are used to clear U_n, for some $\gamma \geq 1/2$, then the number of rounds is $\Omega(\sqrt{n/\gamma})$.

Theorem 5.4 *Every search clearing a complete undirected graph using at most $n^2/4 + \gamma n$ guards, for some $\gamma \geq 1/2$, must use at least $\Omega(\sqrt{n/\gamma})$ rounds.*

Proof: The idea of the proof is to prove that when $n/2 - 2\sqrt{\gamma n} \leq |R_i| \leq n/2 - \sqrt{\gamma n}$, the size of R_i cannot decrees too much in each round. That is, $|R_i| - |R_{i+1}| = O(\gamma)$; thus, it takes at least $\Omega(\sqrt{\gamma n}/\gamma) = \Omega(\sqrt{n/\gamma})$ rounds from the first round when $|R_i| \leq n/2 - \sqrt{\gamma n}$ until the first round when $|R_i| \leq n/2 - 2\sqrt{\gamma n}$. The proof uses the upper bound given in (7) together with the assumption that the number of guards is at most $n^2/4 + \gamma n$.

The proof is technical and we break it to 3 claims. We start by showing that if $|R_{i+1}|$ is relatively small, then $|R_i|$ is not too large.

Claim 5.5 *Let $\gamma > 0$. Assume that at most $n^2/4 + \gamma n$ guards are used to clear U_n. If $|R_{i+1}| < n/2 - \sqrt{\gamma n}$, then $|R_i| \leq n/2 + \sqrt{\gamma n}$.*

Proof: Assume towards contradiction that $|R_i| > n/2 + \sqrt{\gamma n}$. Thus, by (5), the number of guards in Round i is at least

$$\begin{aligned}\frac{|R_i|(n-2|R_{i+1}|) + |R_{i+1}|n}{2} &\geq \frac{(n/2 + \sqrt{\gamma n})(n-2|R_{i+1}|) + |R_{i+1}|n}{2} \\ &= \frac{n^2}{4} + \frac{\sqrt{\gamma n}(n-2|R_{i+1}|)}{2} \\ &> \frac{n^2}{4} + \sqrt{\gamma n}\sqrt{\gamma n} = \frac{n^2}{4} + \gamma n\end{aligned}$$

(the last inequality is implied by the fact that $|R_{i+1}| < n/2 - \sqrt{\gamma n}$). This is a contradiction to our assumption that only $\frac{n^2}{4} + \gamma n$ guards are used. ■ (Of Claim 5.5)

We next give a crude upper-bound on $|R_{i+1}|$, this bound will be needed in the proof of Claim 5.7 to prove a better upper-bound.

Claim 5.6 *Let $1/2 \leq \gamma < n/100$. Assume that at most $n^2/4 + \gamma n$ guards are used to clear U_n. If $n/2 - 2\sqrt{\gamma n} \leq |R_i| \leq n/2 + \sqrt{\gamma n}$, then $|R_{i+1}| \geq n/2 - 5\sqrt{\gamma n}$.*

Proof: Let $|R_i| = n/2 - \alpha$ for some α such that $-\sqrt{\gamma n} \leq \alpha \leq 2\sqrt{\gamma n}$. Assume towards contradiction that $|R_{i+1}| < n/2 - 5\sqrt{\gamma n}$. We next bound $|R_i \vartriangle R_{i+1}|$:

$$3\sqrt{\gamma n} < |R_i| - |R_{i+1}| \leq |R_i \vartriangle R_{i+1}| \leq |R_i| + |R_{i+1}| \leq n - 4\sqrt{n\gamma},$$

which implies

$$|R_i \vartriangle R_{i+1}|(1 + n - |R_i \vartriangle R_{i+1}|) \geq 3\sqrt{\gamma n}(n - 3\sqrt{\gamma n}) \tag{8}$$

(since, the function $r(1+n-r)$, when $3\sqrt{\gamma n} \leq r \leq n - 4\sqrt{\gamma n}$ is minimized in the borders, that is, when $r = 3\sqrt{\gamma n}$).

Furthermore, $\alpha \le 2\sqrt{\gamma n}$, thus,

$$-n\alpha + 3n\sqrt{\gamma n} - 9\gamma n \ge -2n\sqrt{\gamma n} + 3n\sqrt{\gamma n} - 9\gamma n = n\sqrt{\gamma n} - 9\gamma n > \gamma n \tag{9}$$

(the last inequality holds since $\gamma < n/100$). Thus, by (7), (8), and (9), the number of guards in round i is more than

$$\begin{aligned} &\frac{\left(\frac{n}{2} - \alpha\right)(n - 2|R_{i+1}|) + |R_{i+1}|n + 3\sqrt{\gamma n}(1 + n - 3\sqrt{\gamma n})}{2} \\ &\qquad \ge \frac{n^2}{4} + \frac{-\alpha n + 3n\sqrt{\gamma n} - 9\gamma n}{2} > \frac{n^2}{4} + \gamma n, \end{aligned}$$

a contradiction. ■ (of Claim 5.6)

Claim 5.7 *Let $1/2 \le \gamma < n/100$. Assume that at most $n^2/4 + \gamma n$ guards are used to clear U_n. Let α be an integer such that $\sqrt{\gamma n} + 40\gamma \le \alpha \le 2\sqrt{\gamma n}$ and let i a round where $|R_i| \ge n/2 - \alpha$ and $|R_{i+1}| < n/2 - \alpha$. Then, $|R_{i+1}| \ge n/2 - \alpha - 40\gamma$.*

Proof: As $|R_{i+1}| < n/2 - \alpha < n/2 - \sqrt{\gamma n}$, by Claim 5.5, $|R_i| \le n/2 + \sqrt{\gamma n}$. Hence, by Claim 5.6, $|R_{i+1}| \ge n/2 - 5\sqrt{\gamma n}$, and

$$\frac{|R_i|(n - 2|R_{i+1}|) + |R_{i+1}|n}{2} \ge \frac{n^2}{4} - 10\gamma n.$$

Let $r \stackrel{\text{def}}{=} |R_i \vartriangle R_{i+1}|$. The assumption that the number of guards is at most $n^2/4 + \gamma n$ and (7) imply that

$$\frac{n^2}{4} - 10\gamma n + \frac{r(1 + n - r)}{2} \le \frac{n^2}{4} + \gamma n.$$

That is, $r(1 + n - r) \le 22\gamma n$. If $40\gamma \le r \le n - 40\gamma$ then

$$r(1 + n - r) \ge 40\gamma(n - 40\gamma) \ge 40\gamma(n - 40n/100) > 22\gamma n$$

(since $\gamma \le n/100$). This implies that either $r \le 40\gamma$, or $r \ge n - 40\gamma$. If $r \ge n - 40\gamma$, then $|R_i| + |R_{i+1}| \ge |R_i \vartriangle R_{i+1}| = r \ge n - 40\gamma$. Thus, by Claim 5.5,

$$|R_{i+1}| \ge n - 40\gamma - |R_i| \ge n/2 - \sqrt{\gamma n} - 40\gamma,$$

contradicting our assumption that $|R_{i+1}| < n/2 - \sqrt{\gamma n} - 40\gamma$. Thus, $r \le 40\gamma$. Therefore,

$$|R_i| - |R_{i+1}| \le |R_i \vartriangle R_{i+1}| = r \le 40\gamma,$$

and,

$$|R_{i+1}| \ge |R_i| - 40\gamma \ge n/2 - \alpha - 40\gamma.$$

■ (of Claim 5.7)

To complete the proof of the theorem, first note that if $\gamma \ge n/100$, then at least one round is necessary and the theorem holds. Thus, we can assume that $\gamma < n/100$. Let $b \stackrel{\text{def}}{=} \left\lfloor \sqrt{n/\gamma}/40 \right\rfloor$ and j

be an index such that $1 \leq j \leq b$. Finally, let i_j be the first round when $|R_{i_j+1}| < n/2 - \sqrt{\gamma n} - 40j\gamma$. By our choice of parameters

$$n/2 - \sqrt{\gamma n} - 40b\gamma \geq n/2 - 2\sqrt{\gamma n}.$$

By Claim 5.7,

$$|R_{i_j+1}| \geq n/2 - \sqrt{\gamma n} - 40(j+1)\gamma.$$

This implies that $i_1 < i_2 < \cdots < i_{b-1} < i_b$, and there must be at least $b = \Omega(\sqrt{n/\gamma})$ rounds from the round $|R_{i+1}| \leq n/2 - \sqrt{\gamma n} - 40\gamma$ until the first round that $|R_{i+1}| \leq n/2 - 2\sqrt{\gamma n}$. □

References

[1] D. Bienstock and P. Seymour. Monotonicity in graph searching. *J. of Algorithms*, 12(2):239–245, 1991.

[2] H. Buhrman, J. A. Garay, and J. H. Hoepman. Optimal resiliency against mobile faults. In *25th International Symp. on Fault-Tolerant Computing*, pages 83–88, 1995.

[3] N. Cai and R. W. Yeung. Secure network coding. In *International Symposium on Information Theory*, 2002.

[4] B. Chor and E. Kushilevitz. Secret sharing over infinite domains. *J. of Cryptology*, 6(2):87–96, 1993.

[5] J. Feldman, T. Malkin, C. Stein, and R. A. Servedio. On the capacity of secure network coding. In *Proc. 42nd Annual Allerton Conference on Communication, Control, and Computing*, 2004.

[6] M. Franklin, Z. Galil, and M. Yung. Eavesdropping games: a graph-theoretic approach to privacy in distributed systems. *J. of the ACM*, 47(2):225–243, 2000.

[7] J. A. Garay. Reaching (and maintaining) agreement in the presence of mobile faults. In G. Tel, editor, *8th International Workshop on Distributed Algorithms*, volume 857 of *Lecture Notes in Computer Science*, pages 253–264. Springer Verlag, 1994.

[8] J. A. Garay, R. Gennaro, C. S. Jutla, and T. Rabin. Secure distributed storage and retrieval. *Theoretical Computer Science*, 243(1-2):363–389, 2000.

[9] A. Herzberg, S. Jarecki, H. Krawczyk, and M. Yung. Proactive secret sharing, or: how to cope with perpetual leakage. In D. Coppersmith, editor, *Advances in Cryptology – CRYPTO '95*, volume 963 of *Lecture Notes in Computer Science*, pages 339–352. Springer-Verlag, 1995.

[10] K. Jain. Security based on network topology against the wiretapping attack. *IEEE Magazine, Special issue on Topics in Wireless Security*, 2004.

[11] L. M. Kirousis and C. H. Papadimitriou. Searching and pebbling. *Theoretical Computer Science*, 47:205–218, 1986.

[12] A. LaPaugh. Recontamination does not help to search a graph. *J. of the ACM*, 40(2):224–245, 1993.

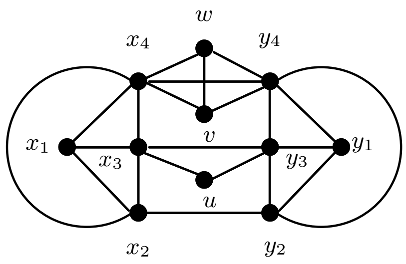


Figure 7: The graph G_4.

[13] R. Ostrovsky and M. Yung. How to withstand mobile virus attacks. In *Proc. of the 10th ACM symp. on Principles of Distributed Computing*, pages 51–59, 1991.

[14] T. D. Parsons. Pursuit evasion in a graph. In Y. Alavi and D. R. Lick, editors, *Theory and Application of Graphs*, pages 426–441. Springer-Verlag, 1976.

[15] R. Prasad, K. Srinathan, and C. Pandu Rangan. On proactive perfectly secure message transmission. Unpublishe manuscript, 2006.

[16] C. E. Shannon. Communication theory of secrecy systems. *Bell System Technical Journal*, 28(4):656–715, 1949.

A An example of the Problem in the Proof of [6]

We next describe an example in which the characterization in [6] is incorrect. Consider the graph G_4, described in Figure 7, from the family of graphs $\{G_n\}$ used in the proof of Theorem 3 in [6]. The graph G_4 consists of a left clique (complete graph) with vertices $\{x_1, x_2, x_3, x_4\}$, a right clique with vertices $\{y_1, y_2, y_3, y_4\}$, a middle clique (M_2) consisting of vertices $\{x_2, y_2\}$, a middle clique (M_3) consisting of vertices $\{x_3, y_3, u\}$, and a middle clique (M_4) consisting of vertices $\{x_4, y_4, v, w\}$.

The following vertex search (suggested by the proof of Theorem 3 in [6]) cuts the directed layered graph, but fails to cut the undirected layered graph:

Round	Guarded set
1	$\{x_1, x_2, x_3, x_4\}$
2	$\{y_2, x_2, x_3, x_4\}$
3	$\{u, y_3, x_3, x_4\}$
4	$\{y_2, x_2, y_3, x_4\}$
5	$\{v, w, y_4, x_4\}$

Round	Guarded set
6	$\{y_2, x_2, y_3, y_4\}$
7	$\{u, y_3, x_3, y_4\}$
8	$\{y_2, x_2, y_3, y_4\}$
9	$\{y_1, y_2, y_3, y_4\}$

One unguarded path in the undirected layered graph goes from y_4 in the first layer, to y_1 in the second layer, to y_4 in the third layer, to y_1 in the fourth layer, to y_3 in the fifth layer, *back* to u in the fourth layer, to x_3 in the fifth layer, to x_1 in the sixth layer, to x_2 in the seventh layer, to x_1 in the eight layer, to x_2 in the ninth layer. With respect to Lemma 4 in [6], note that Alice is unable to simulate the behavior of y_3 in the fifth layer, even though this node is in the set V_s, since the set V_r includes u in the fourth layer.

In fact, the proof of Theorem 3 in [6] implies that the only search that clears the directed layered graph with 4 guards is basically the search described above. Since every search that clears the undirected layered graph clears the directed layered graph, and the above search does not clear the undirected layered graph, every search that cuts the undirected layered graph, uses at least 5 guards.