UC Berkeley Technical Report CSD-96-912

Hidden Cliques as Cryptographic Keys

Ari Juels* **Marcus Peinado†**

August 22, 1996

Abstract

We demonstrate in this paper a very simple method for "hiding" large cliques in random graphs. While the largest clique in a random graph is very likely to be of size about $2 \log_2 n$, it is widely conjectured that no polynomial-time algorithm exists which finds a clique of size $(1+\epsilon)\log_2 n$ with significant probability for any constant $\epsilon > 0$. We show that if this conjecture is true, then when a clique of size at most $(2-\delta)\log_2 n$ for constant $\delta > 0$ is randomly inserted ("hidden") in a random graph, finding a clique of size $(1+\epsilon)\log_2 n$ remains hard. In particular, we show that if there exists a polynomial-time algorithm which finds cliques of size $(1+\epsilon)\log_2 n$ in such graphs with probability $\frac{1}{poly}$, then the same algorithm will find cliques in completely random graphs with probability $\frac{1}{poly}$. Given the conjectured hardness of finding large cliques in random graphs, we therefore show that hidden cliques may be used as cryptographic keys.

*Department of Computer Science, University of California at Berkeley. Supported by a UC Regents Fellowship and NSF Grant CCR-9505448. E-mail: **juels@cs.berkeley.edu**.

†German National Research Center for Information Technology (GMD). Supported in part by DFG grant SFB 408. A portion of this this research was conducted by the author while visiting the International Computer Science Institute, Berkeley, CA. E-mail: **Marcus.Peinado@gmd.de**.

UC Berkeley Technical Report CSD-96-912

1 Introduction

1.1 Background

A *clique* of size k in a graph $G = (V, E)$ is a complete subgraph on k nodes, i.e., a set of k nodes such that every pair is connected by an edge. The problem of finding a clique of maximal size, or even near-maximal size, is believed to be hard for general graphs. Let $\omega(G)$ denote the maximum value of k such that G contains a clique of size k and let $n = |V|$. Unless $P = NP$, it is known that for any $\delta > 0$, no polynomial-time algorithm can find a clique whose size is within a factor of $n^{\frac{1}{3}-\delta}$ of $\omega(G)$ for general graphs [3].[1]

Finding "large" cliques in randomly-generated graphs also appears to be quite hard, and has prompted a great deal of experimental and theoretical interest. Most of the research in this vein focuses on the random graph model $\mathcal{G}_{n,1/2}$, i.e., the uniform distribution over graph instances containing n nodes. A graph G may be drawn from this distribution by inserting each of the $\binom{n}{2}$ possible edges into G independently with probability 1/2. For the overwhelming majority of such graphs, the largest clique is of size $2 \log_2 n - o(\log \log n)$ [2]. Smaller cliques exist in abundance: for $k = c \log_2 n$, where $c \in (0, 2)$, the expected number of cliques of size k is $\Omega(n^{\log n})$.

It is widely conjectured that for any constant $\epsilon > 0$, there does not exist a polynomial-time algorithm capable of finding cliques of size $(1 + \epsilon) \log_2 n$ with significant probability in random graphs. Karp [5] first issued the challenge of finding such an algorithm twenty years ago. Jerrum [4] considerably extended this challenge in calling for a randomized, polynomial-time algorithm capable of finding a clique of size $1.01 \log_2 n$ with high probability over random graphs containing a clique of size $n^{0.49}$. In support of the difficulty of finding such an algorithm, Jerrum demonstrates the existence of an initial state from which the Metropolis algorithm, a fixed-temperature variant of simulated annealing, cannot find a clique of size $(1 + \epsilon) \log_2 n$ for any constant $\epsilon > 0$ in expected polynomial time. He shows, moreover, that this situation holds even when a clique of size as large as $n^{1/2-\delta}$ for constant $\delta > 0$ is randomly inserted ("hidden") in randomly-generated graphs. A number of experimental studies seem to confirm the hardness of the problem of finding large cliques in random graphs. A brief survey of these may be found in [6].

1.2 This paper

Our aim in this paper is to show that it is possible to "hide" relatively large cliques in random graphs in such a way that finding such a clique is as hard as finding a large clique in an unaltered random graph. Let p denote the uniform distribution $\mathcal{G}_{n,1/2}$ over graphs with n nodes. Let p'_k denote the distribution obtained as follows: select a graph $G = (V, E)$ from p, and then form a clique on k nodes selected uniformly at random from V. We refer to a clique formed in this manner as a *hidden* clique. We shall show that when

[1] Håstad shows in [3] that the result holds for a factor of $n^{\frac{1}{2}-\delta}$ for any constant $\delta > 0$ under the assumption that $NP \neq coR$. He has subsequently raised this factor to $n^{1-\delta}$.

$k \leq (2-\delta)\log_2 n$ for any constant $\delta > 0$, finding a large clique in p'_k is as hard as finding one in p. More precisely, we shall show that if there exists an algorithm A which finds a clique of size $(1+\epsilon)\log_2 n$ in p'_k with probability $\frac{1}{q(n)}$, for some polynomial $q(n)$, then the same algorithm can find a clique of size $(1+\epsilon)\log_2 n$ in p with probability $\frac{1}{q'(n)}$ for some polynomial $q'(n)$.

If the conjectured hardness of finding large cliques holds, the result in this paper will imply that hidden cliques may be used as cryptographic keys. In the simplest scenario, a user may generate a random graph G and then plant a clique K of size $\frac{3}{2}\log_2 n$ in G, yielding a graph G'. The hidden clique K may then serve as a password: by revealing K, the user confirms that she was responsible for generating G'.

The remainder of this paper is organized into two sections. In section 2, we describe and prove the main theorem of this paper, stating that large hidden cliques are as hard to find as large cliques in $G_{n,1/2}$. In section 3, we discuss some issues surrounding the application of this result to the realm of cryptography.

2 Main theorem

2.1 Sketch of proof

Let E_k denote the expected number of cliques of size k in a graph drawn from p. Let $C_k(G)$ denote the number of distinct (but possibly overlapping) cliques of size k in a specific graph instance G. Note that when considered appropriately over the distribution p, $C_k(G)$ may be regarded as a random variable with mean E_k. We shall denote this random variable by C_k.

Our proof will begin by demonstrating that when $C_k(G)$ is close to E_k, the probability of graph G in the distribution p'_k will be close to that in p. In other words, when the number of cliques in a graph G is close to the expected number E_k, the process of planting a clique of size $(2-\delta)\log_2 n$ in a random graph will yield G with probability similar to that of the process of simply generating a random graph. We shall then show that the variance of C_k is low. This will imply two things: first, that most graphs G are "good", i.e., for most graphs, $p'_k(G)/p(G)$ is less than a relatively small polynomial; second, that "bad" graphs, i.e., those for which $p'_k(G)/p(G)$ is large, will occupy a small fraction Δ of p'_k. In fact, we will be able to make this fraction Δ arbitrarily small. Therefore, an algorithm A which successfully locates a large clique in a $(\frac{1}{poly})$-fraction of graphs in p'_k must be locating such cliques in a set M of good graphs such that $p'_k(M) = \frac{1}{poly} - \Delta = \frac{1}{poly}$. Since graphs in M are good, $p'_k(M) = \frac{1}{poly}$ will imply $p(M) = \frac{1}{poly}$. Thus, A will successfully locate a large clique in a $(\frac{1}{poly})$-fraction of graphs in p, the uniform distribution over graphs.

2.2 The proof

Lemma 1 $p'_k(G) = \frac{C_k(G)}{E_k} p(G)$.

Proof. Selecting a graph from p'_k may be viewed as the process of selecting a graph G' from p and then planting a clique on a set K of k nodes chosen uniformly at random. In order for the resulting graph to be identical to G, it must be that the nodes K form a clique in G. An appropriate set K will thus be chosen with probability $C_k(G)/\binom{n}{k}$. It must also happen that the edges in G' which lie outside of K correspond exactly to those in G. More precisely, for all edges e not strictly contained in K, we require $e \in G' \iff e \in G$. This will occur with probability $2^{-\binom{n}{2}+\binom{k}{2}}$. Thus,

$$p'_k(G) = \frac{C_k(G)}{\binom{n}{k}} 2^{-\binom{n}{2}+\binom{k}{2}}. \tag{1}$$

The expected number of cliques in p is easily seen to be $\binom{n}{k}/2^{\binom{k}{2}}$. The definition of p implies that $p(G) = 2^{-\binom{n}{2}}$ for any graph instance G. Combining these two facts with eqn. 1 yields the lemma. $\square$

Lemma 1 states that when the number of cliques in a graph instance G is close to its expectation over p, then $p'_k(G) \approx p(G)$. Our goal now is to show that for most graphs G, $p'_k(G)$ is only a polynomial factor larger than $p(G)$. For this we need to show that C_k is concentrated somewhat tightly around its mean, E_k. We shall accomplish this by showing that the variance of C_k is small.

Lemma 2 *Let $k = (2-\delta)\log_2 n$ for some constant $\delta > 0$. Then* $\mathrm{Var}[C_k] = O(n^4 \log n) E_k^2$.

Proof. We employ the method in [2], Chapter XI, and consider pairs of cliques in G. This gives us

$$\mathrm{E}[C_k^2] = \sum_{i=0}^{k} \binom{n}{k} \binom{k}{i} \binom{n-k}{k-i} 2^{-2\binom{k}{2}+\binom{i}{2}}, \tag{2}$$

and thus,

$$\frac{\mathrm{E}[C_k^2]}{\mathrm{E}^2[C_k]} = \sum_{i=0}^{k} \binom{n}{k}^{-1} \binom{k}{i} \binom{n-k}{k-i} 2^{\binom{i}{2}}. \tag{3}$$

Let us denote the i^{th} term in the above sum by f_i. Clearly $f_0 < 1$. By employing the well-known bounds $\binom{n}{k} \le (\frac{ne}{k})^k$ and $\binom{n}{k} \ge (\frac{n}{k})^k$, we obtain for $i > 0$ the inequality

$$f_i \le \left(\frac{k\mathrm{e}}{i}\right)^i \left(\frac{k}{n}\right)^k \left(\frac{(n-k)\mathrm{e}}{k-i}\right)^{k-i} 2^{\binom{i}{2}}. \tag{4}$$

Algebraic manipulation shows that the above is equal to

$$\left(\frac{k^2}{i}\right)^i \left(\frac{(n-k)k}{k-i}\right)^{k-i} \mathrm{e}^k n^{-k} 2^{\binom{i}{2}},$$

which is less than

$$\left(\frac{k}{k-i}\right)^{k-i} \mathrm{e}^k \left(\frac{k^2}{n}\right)^i 2^{\frac{i^2}{2}}.$$

Let us first consider the quantity $(\frac{k}{k-i})^{k-i}$. This is equal to $(1+\frac{i}{k-i})^{k-i} \le e^i$. Since $i < k < (2-\delta)\log_2 n$ for some constant $\delta > 0$, it follows that $(\frac{k}{k-i})^{k-i} = O(n^2)$. Similarly, it is also the case that $e^k = O(n^2)$.

Now let us consider the quantity $\beta = (\frac{k^2}{n})^i 2^{\frac{i^2}{2}}$. Clearly, $\log_2 \beta = -i\log_2 n + i^2/2 + 2i\log_2 k$. Since $k = O(\log n)$, it follows that $\log_2 \beta < 0$ if $i < 2\log_2 n$. Since $i < k \le (2-\delta)\log_2 n$ for some constant $\delta > 0$, it follows that $\beta < 1$ for all values of i. Tying together all of the above, we see that $f_i = O(n^4)$ for all i, and therefore

$$\sum_{i=0}^{k} f_i = \frac{\mathrm{E}[C_k^2]}{\mathrm{E}^2[C_k]} = O(n^4 \log n). \tag{5}$$

The lemma follows. $\square$

From the above lemma, it follows that "bad" graphs, i.e., those graphs G such that $C_k(G) \gg E_k$, constitute a small fraction of p. As we see in the next lemma, when $k = (2-\delta)\log_2 n$ for some constant $\delta > 0$, such graphs also occupy a small fraction of p'_k.

Lemma 3 *Define the set Z of bad graphs to include those graphs G such that $C_k(G) > n^{2h}E_k$ for some constant $h > 0$. In other words, let $Z = [G \mid C_k(G) > n^{2h}E_k]$. Then $p'_k(Z) = O(n^{-h+(4+\epsilon)})$ for any constant $\epsilon > 0$.*

Proof. We shall determine the size of the set $p'_k(Z)$ of bad graphs by considering a set of intervals Z_j whose union contains Z. By computing upper bounds individually on the sets $p'_k(Z_j)$, we shall obtain an upper bound on $p'_k(Z)$.

Let $Z_j = [G \mid n^{jh}E_k < C_k(G) \le n^{(j+1)h}E_k]$. Clearly, $Z \subset \bigcup_{j=2}^{\infty} Z_j$. By Lemma 2, $\mathrm{Var}[C_k] = O(n^4 \log n)E_k^2$. Therefore, by Chebyshev's inequality (as stated in, e.g, [1]), it may be seen that $p(Z_j) < n^{-2jh+(4+\epsilon)}$ for any constant $\epsilon > 0$. By Lemma 1 then,

$$p'_k(Z_j) < n^{(j+1)h}\left(\frac{1}{n^{2jh-(4+\epsilon)}}\right). \tag{6}$$

Since $Z \subset \bigcup_{j=2}^{\infty} Z_j$, it follows that

$$\begin{aligned}
p'_k(Z) &< \sum_{j=2}^{\infty} p'_k(Z_j) \\
&< \sum_{j=2}^{\infty} \frac{n^{(j+1)h}}{n^{2jh-(4+\epsilon)}} \\
&= \sum_{j=2}^{\infty} n^{-jh+h+(4+\epsilon)} \\
&= n^{-h+(4+\epsilon)} \sum_{j=0}^{\infty} n^{-jh} \\
&= n^{-h+(4+\epsilon)} O(1) \\
&= O(n^{-h+(4+\epsilon)}),
\end{aligned}$$

which proves the lemma. $\square$

By making the constant h large enough – in other words, by making the graphs in Z sufficiently "bad" – we may make the set Z arbitrarily small. By making Z small, we ensure that an algorithm A which successfully finds cliques in p'_k does so principally on good graphs. These graphs will constitute a $(\frac{1}{poly})$-fraction of graphs in p, implying that A successfully finds cliques in p with probability $\frac{1}{poly}$.

Theorem 4 *Suppose that $k \leq (2-\delta)\log_2 n$ for some $\delta > 0$. Suppose then that there exists a deterministic, polynomial-time algorithm A which finds a clique in graphs drawn from p'_k with probability $\frac{1}{q(n)}$, for some polynomial $q(n)$. Then there exists a polynomial $q'(n)$ such that A finds a clique in graphs drawn from p with probability $\frac{1}{q'(n)}$.*

Proof.
Suppose $\frac{1}{q(n)} = \Omega(n^{-j})$ for some constant j. Let Z be the set of graphs in p'_k such that $C_k(G) > (n^{2j+10})E_k$. By Lemma 3, $p'_k(Z) = O(n^{-j-1})$. Let Q denote the set of graphs G not in Z on which A finds a clique. Clearly, $p'_k(Q) = \Omega(n^{-j}) - p'_k(Z) = \Omega(n^{-j}) - O(n^{-j-1}) = \Omega(n^{-j})$. Therefore, by Lemma 1, $p(Q) = \Omega(n^{-j})(n^{-2j-10}) = \Omega(n^{-3j-10})$. This proves the theorem. $\square$

Observe that this theorem may be extended in a suitable fashion to randomized algorithms A. In particular, if A is a randomized algorithm which finds cliques in p'_k with probability $\frac{1}{poly}$ in expected polynomial time, then A also finds cliques in p with probability $\frac{1}{poly}$ in expected polynomial time.

Theorem 4 holds also for distributions p_k^c, where a graph from p_k^c is generated by randomly planting any constant number of cliques $K_1, K_2, \ldots, K_c$ of size k in a random graph. In other words, it is possible to hide at least a constant number of large cliques in a random graph.

3 Application of the Result

As mentioned in the introduction to this paper, a graph G containing a hidden clique K of sufficiently large size, say, $\frac{19}{10}\log_2 n$, might be used as a cryptographic key. In particular, the graph G might be published as a public key, while the identity of K might constitute a private key. Assuming the hardness of finding large cliques, deducing K from the public key G, or any other clique of the same size as K, would be infeasible.

In practice, the information required to specify the graph G, and hence the public key, would be rather large with respect to the private key. In order to make it infeasible to find cliques of size at least $\frac{19}{10}\log_2 n$, for instance, the graph G would probably have to contain at least $10,000$ nodes. Hence G would contain about $50,000,000$ edges, and specifying these edges would require almost 6.25 Mbytes of data. The private key K, on the other hand, would consist of only about 20 nodes, and hence might be specified by 280 bits, or about 35 bytes.

As mentioned above, however, our main theorem holds for any constant number of cliques. In other words, it is possible to have a set of multiple private keys $K_1, K_2, \ldots, K_c$ associated with a single public key G. If many (e.g., several hundred) cliques may be securely hidden in practice, this would enable cliques to be used with relative efficiency as cryptographic keys.

The possibility of having multiple private keys associated with a public key is interesting in itself. Cryptographic keys based on hidden cliques have a number of other appealing properties as well. One of these, for instance, is the ability to create private keys "hierarchically". Suppose that G is the random graph into which a clique K is hidden, and $G' = (V, E')$ is the graph resulting from this implantation. A party with knowledge of G is likely to be able to extract K. In particular, the set $E' - E$ will contain half of the edges of K on average; with this information, K can be easily determined with high probability. Consider, therefore, the following protocol. Party P_1 generates a random graph G and randomly inserts into it a clique K_1, yielding graph G_1. Party P_1 then passes G_1 to P_2, who randomly inserts a clique K_2, yielding G_2. This process is continued through user P_c, who then publishes the key G_c. Observe that in a suitably formulated system, with high probability, P_1 can use its knowledge of G_1 to extract the private keys of $P_2, P_3, \ldots P_c$ (although it should be observed that P_1 cannot determine which key belongs to which party). Parties $P_2, P_3, \ldots, P_c$, however, cannot extract the private key of P_1. In general, party P_i can extract the private keys of all users P_j for $j > i$, while the reverse would require the ability to find a large clique, and is therefore likely to be infeasible.

The problem of finding large cliques is quite different from the traditional, numerical problems on which cryptographic systems are generally based, like factoring and the computation of discrete logs. Hence another interesting possibility arises: that hidden cliques might prove resistant to techniques like quantum computing, which threaten someday to render numerical systems insecure.

4 Acknowledgments

Thanks are due principally to Manuel Blum as the inspiration for this paper: the result presented here addresses an open problem posed by him in his undergraduate course at UC Berkeley on randomized algorithms. We wish also to thank Sanjoy Dasgupta, Alistair Sinclair, and David Wagner for their comments on drafts of this paper, as well as Umesh Vazirani, Mor Harchol-Balter, Michael Mitzenmacher, and Satish Rao for discussions of our work.

References

[1] N. Alon, J.H. Spencer, and Paul Erdős. *The Probabilistic Method.* John Wiley & Sons, 1992.

[2] B. Bollobás. *Random Graphs.* Academic Press, 1985.

[3] J. Håstad. Testing of the long code and hardness for clique. In *Proceedings of the 28th Annual ACM Symposium on the Theory of Computing*, pages 11–19, 1996.

[4] M. Jerrum. Large cliques elude the Metropolis process. *Random Structures and Algorithms*, 3(4):347–359, 1992.

[5] R.M. Karp. Probabilistic analysis of some combinatorial search problems. In J.F. Traub, editor, *Algorithms and Complexity: New Directions and Recent Results.* Academic Press, 1976.

[6] M. Peinado. *Approximation Algorithms for Maxclique and Maxcut and their Applications.* PhD thesis, Boston University, 1995.