

Multi-level Crypto Disk: Secondary Storage with Flexible Performance Versus Security Trade-offs

Shiva Chaitanya

Netapp Inc
Shiva.Chaitanya@netapp.com

Bhuvan Urgaonkar

Pennsylvania State University
bhuvan@cse.psu.edu

Anand Sivasubramaniam

Pennsylvania State University
anand@cse.psu.edu

Abstract

Secondary storage devices have become increasingly vulnerable to security attacks as they are now accessed remotely, attached to mobile devices, or used in other previously unanticipated operating environments. Storage vendors have responded to this by offering solutions that encrypt data on the fly—in software or device firmware—before recording. The performance versus security trade-off offered by these secure devices is limited due to their use of only a single level of data encryption. To address these limitations, we propose the Multi-level Crypto Disk (MLCD), a generic storage device with multiple crypto levels for encoding data. Using stochastic modeling, we derive optimal policies to dynamically select crypto levels for data in an MLCD to achieve desired performance versus security trade-offs.

I. Introduction

Storage systems are known to be prone to data loss and compromise that result from improper access, disposal, or theft of computer equipment. Our growing reliance on digital storage implies that such incidents can have serious undesirable repercussions. For example, at Pfizer, an employee without authorized access retrieved the records of up to 34,000 employees, gaining access to information such as names, social security numbers, and bank and credit card information [1]. In another case, laptops containing the mental health histories of over 300,000 people, and the full names and social security numbers of almost 2,000 people, were stolen from the Pennsylvania Public Welfare [2].

The recent emergence of secondary storage devices supporting on-board encryption capabilities holds the promise of improving the ability of organizations to safely store and retrieve sensitive data. These include both traditional magnetic disk drives as well as disks based on newer non-volatile, solid-state memory technologies such as NAND Flash. For example, Seagate [3] and Hitachi [4] now offer 2.5 inch magnetic disk drives with native, hardware-based, “full disk encryption” (FDE). Similarly, Samsung, in collaboration with Wave Systems, now provide a NAND Flash-based self-encrypting disk drive [5].

Inevitably, the security guarantees (Primarily, Data Confidentiality and other properties such as Data Integrity and Authentication) in these secure disks come at the cost of some degradation

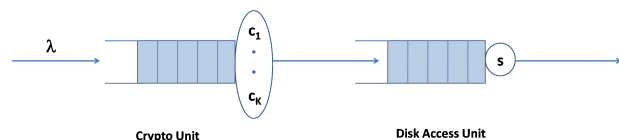


Fig. 1: Queuing model for a MLCD.

in I/O performance. This degradation results from the additional time (compared with a traditional disk) consumed during request processing for the operations within the crypto unit.

II. Multi-level Crypto Disk

In order to provide optimal levels of data security while meeting desired I/O performance targets, we propose to incorporate multiple cryptographic algorithms within the crypto unit of a secure storage device. We call such a device a *multi-level crypto disk (MLCD)*. The crypto unit in an MLCD can choose one out of the several available crypto levels—each offering a different trade-off between computational requirements and security properties—to process a data block before it is written onto the media.

A. Analytic Modeling of an MLCD

We model an MLCD as a network of two queuing servers shown in Figure 1. The first server represents the crypto unit while the second server represents the disk access unit involved in servicing a write request following the crypto unit. We assume the crypto unit to be capable of operating at n different crypto levels chosen from the set $L = \{L_1, \dots, L_n\}$. We denote by the random variable c_i ($1 \leq i \leq n$) the time spent by a request inside the crypto unit when it operates at level L_i . Let \bar{c}_i denote the average of c_i . We assume the levels L_1, \dots, L_n to have monotonically increasing service times, i.e., $0 < \bar{c}_1 < \dots < \bar{c}_n$. Finally, we use the random variable s to denote the time to service a request after its processing by the crypto unit (i.e., the service time of the second queuing server in Figure 1) and denote its average by \bar{s} .

An Optimization Framework to Capture MLCD Performance/Security Trade-offs. : We use the average response time of a request as our performance metric. Our goal is to understand

and characterize the trade-off offered by an MLCD between performance of I/O requests and a notion of data security we choose to call “goodness of security” (GoS). GoS is defined by a revenue function $\$: L \rightarrow \mathbb{R}^+$ such that a write request encrypted at the crypto level L_i generates revenue at the rate $\$(L_i)$. Additionally, we assume that $\forall i, j : i < j \Leftrightarrow \$(L_i) < \$(L_j)$. Intuitively, encrypting a request at a higher crypto level generates higher revenue (which captures the improved GoS for that request) at the cost of requiring more processing time within the crypto unit.

Overall GoS could be defined in many ways. We choose to define overall GoS by the function $\$$ which is the average revenue generated per unit time. We use \bar{R} and R^{max} to denote the achieved average request response time and the specified bound on it, respectively. We cast the following optimization problem whose solution provides the control rule we desire:

Revenue Maximization: Maximize $\$$ s.t. $\bar{R} \leq R^{max}$.

B. Stochastic Model for Tractability

We impose certain restrictions on our MLCD model to make the Revenue Maximization problem tractable. These restrictions concern the nature of the I/O request arrival process as well as the service times associated with the queues in our MLCD model. We assume that I/O requests arrive according to a Poisson process with rate λ . We assume that in the crypto queuing server, the service time for each of the crypto levels, c_i ($1 \leq i \leq n$), is exponentially distributed with mean \bar{c}_i . We denote the corresponding service rates by μ_1, \dots, μ_n , ($\mu_i = \frac{1}{\bar{c}_i}$). Finally, we also assume the service time for the second queuing server s to be exponentially distributed with mean \bar{s} with $\theta = \frac{1}{\bar{s}}$ being the corresponding service rate.

We model such an MLCD using a semi-Markov decision process [6]. Unlike continuous-time markov decision process, a semi-Markov decision process allows decisions to be made at discrete time instants—specifically, only at state transition epochs. That is, a decision is made in a state only at the instant of entry into the state. This decision persists until an entry to the next state. We would like to exploit this characteristic of semi-MDPs to develop a reduction for our Revenue Maximization problem. We achieve this by creating two different classes of states in our semi-MDP that we describe next.

States and Decision Sets: Each state of the semi-MDP belongs to one of two classes. The first class of states captures situations when the system is ready to choose the crypto level for a request that the crypto unit is about to start processing. Therefore, a transition into a state belonging to this class happens when a request: (i) leaves the first queuing server and there is at least one request remaining in it or (ii) a new request arrives into an empty first queuing server. We represent a state belonging to this class using the tuple (q_1, q_2) corresponding to the number of requests present in the two queuing servers. Note that this implies that there is no state belonging to this class where q_2 is zero with the exception of $(1, 0)$. The second class captures all other situations when the system is currently processing a request in the first queuing server at a certain crypto level. We denote this state by the three-tuple (q_1, q_2, L_i) , implying that the current number of requests in the two queuing servers are q_1 and q_2 , respectively and the current crypto level is L_i .

The feasible decision sets for our states are defined as follows. For the first class of states represented by (q_1, q_2) , the decision

set is $L = \{L_1, \dots, L_n\}$, i.e., all crypto levels. This captures the requirement that any decision be allowed when a request departs the first queuing server while still leaving it non-empty or a new request arrives into an empty first queuing server (see (i) and (ii) above). For the second class of states represented by (q_1, q_2, L_i) , the feasible decision set consists of only one member and is L_i . This is because the crypto level may not be changed upon entering these states (which can only happen upon request arrivals to the first queuing server or departures from the second). We incorporate the current crypto level into the state definition for the second class of states. This allows us to remember the decision made at the most recent departure of a request from the first queuing server.

Transitions Between States of Our Semi-MDP: Recall that new decisions are possible only upon entry to the first class of states. Based on this, let us consider transitions from these two classes of states, in turn.

- *Transitions from the first class of states:* When in a state (q_1, q_2) , any crypto level decision can be made. Suppose the level L_i is chosen in this state indicating that the next available request in the crypto unit is to be processed at level L_i . Corresponding to a request arrival into the system, the next state is $(q_1 + 1, q_2, L_i)$ with a rate of transition equal to λ . Similarly, for a request departure from the system, the next state is $(q_1, q_2 - 1, L_i)$ with rate of transition equal to θ . For the event corresponding to a request departure from the crypto unit, we no longer need to remember the previously chosen level L_i . Therefore, the next state is $(q_1 - 1, q_2 + 1)$ with the rate of transition being μ_i .
- *Transitions from the second class of states:* Transitions from the second class of states are easily derived as follows. From a state (q_1, q_2, L_i) , only decision L_i can be made. When a request leaves the crypto unit, the crypto level for the next request (if any) can be chosen and hence the next state must belong to the first class of states. We create a transition to capture this by adding an edge from (q_1, q_2, L_i) to $(q_1 - 1, q_2 + 1)$ with a rate of μ_i . Other transitions occurring due to request arrival or departures to/from the overall system must continue to remember the current crypto level. Therefore, we create (i) a transition from (q_1, q_2, L_i) to $(q_1 + 1, q_2, L_i)$ with a rate λ corresponding to a request arrival to the system and (ii) a transition from (q_1, q_2, L_i) to $(q_1, q_2 - 1, L_i)$ with a rate θ corresponding to a request departure from the system (with appropriate boundary conditions on the values of q_1 and q_2). Figure 2 represents the semi-MDP corresponding to an MLCD possessing two crypto levels L_1 and L_2 .

Reward Functions.: The final component of our reduction is the set of reward functions associated with each (state, decision) pair of our semi-MDP. Average reward-constrained MDPs [7] consider the following problem of maximizing one reward function subject to constraints on other reward functions.

We choose two reward functions, one corresponding to the average response time and the other to the revenue. For a (state, decision) pair $(S, D = L_i)$, the reward function corresponding to revenue is simply $\$(L_i)$. The appropriateness of this reward function is straightforward from our definition of revenue. The other reward function (to correspond to the average response time) is less easy to define. We know from Little’s Law that the average response time for any queuing system at steady state is

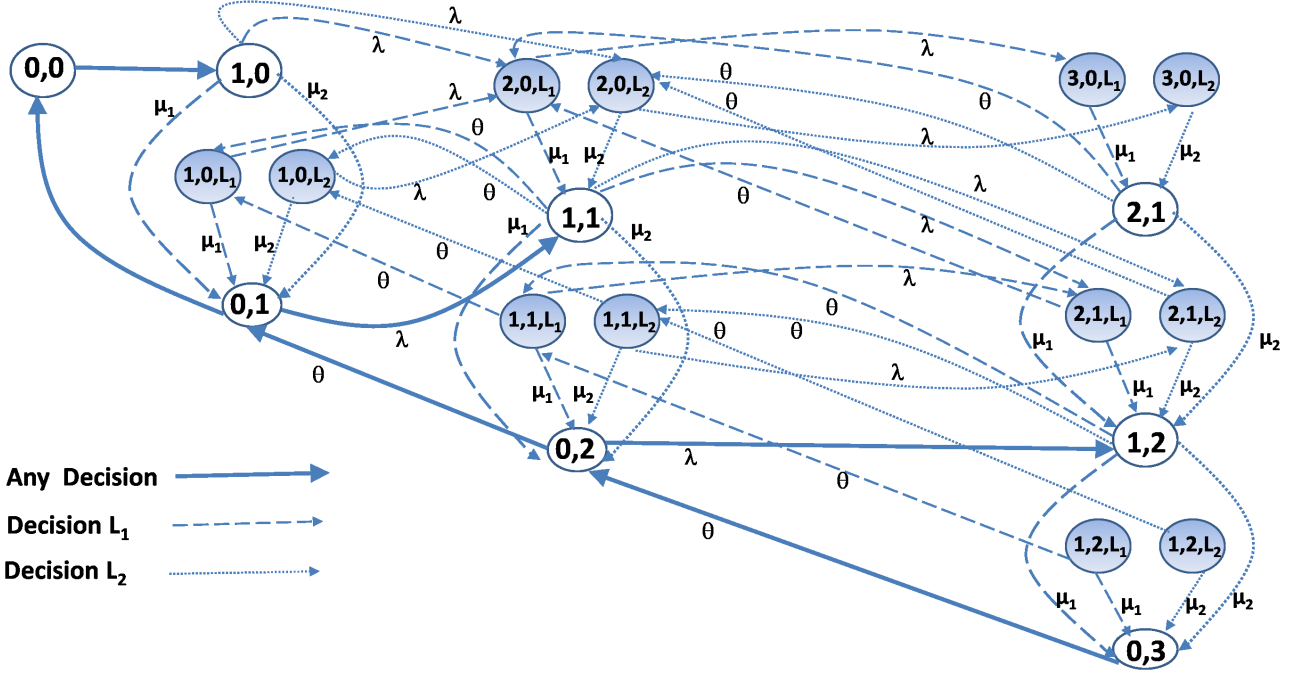


Fig. 2: Semi-Markov Decision Process for an MLCD with two crypto levels. States corresponding to 2 and 3 requests in the overall system and the corresponding transitions are shown.

N/λ , where N is the average number of requests in the system and λ is its throughput. If we define this reward function for each state to be directly proportional to the number of requests in that state, we would achieve the above property. Based on this intuition, for a (state, decision) pair (S, D) , we choose the second reward function to be $\frac{i}{\lambda}$ where i is the number of requests in the system corresponding to the state S . For our semi-MDP, this reward function would then simply be $(\frac{q_1 + q_2}{\lambda})$ in all states (q_1, q_2) and (q_1, q_2, L_i) .

Solution to Revenue Maximization.: It is known that for an average reward-constrained semi-MDP, there exists an optimal policy that is M -randomized stationary where M is the number of reward constraints [7]. A non-randomized stationary policy P always chooses the same decision in a given state S ; the decision made in state S is denoted by $P(S)$. A 1-randomized stationary policy is one that is non-randomized in all states except for one state where the decision is randomized between two actions. Note that such a policy is still stationary, in the sense that the decision policy depends only on the current state. This definition extends to a M -randomized stationary policy. The optimal policy for our stochastic problem is a 1-randomized stationary policy because there is exactly one reward constraint.

Figure 3 shows an example of one such policy to pick the crypto levels within the crypto unit of an MLCD with four crypto levels. As shown, there is a one-to-one mapping from each state (tuple of request numbers in the two queuing servers) to a crypto level, except for one state where two levels can be picked probabilistically.

References

[1] L. Howard, "Pfizer has another breach of security. new london day, aug 2007," in <http://www.theday.com/re.aspx?re=7b6d810e-f7ef-4243-a86c-bfdb9093f983>.

CRYPTO UNIT (Number of Requests)	DISK ACCESS UNIT (Number of Requests)	CRYPTO LEVEL
1	0	L3
1	1	L3
2	0	L2
1	2	L2 with prob 0.2 L1 with prob 0.8
2	1	L1
3	0	L1
...

Fig. 3: Example policy for an MLCD with four crypto levels L1, L2, L3 and L4

[2] J. Murphy, "Computers stolen from welfare office. harrisburg patriot news, sept 11 2007," in <http://www.pennlive.com/midstate/patriotnews/article121468.ece>.

[3] "Seagate momentus 7200 fde hard drives," in http://www.seagate.com/www/en-us/products/laptops/momentus/momentus_7200_fde/.

[4] "Hitachi travelstar 7k320," in <http://www.hitachigst.com/portal/site/en/products/travelstar/7K320/>.

[5] "Full disk encryption comes to solid state drives," in http://www.samsung.com/global/business/semiconductor/newsView.do?news_id=995.

[6] R. A. Howard, "Dynamic probabilistic systems. vol. ii: Semi-markov and decision processes," in *Series in Decision and Control*, 1971.

[7] E. Feinberg, "Optimal control of average reward constrained continuous-time finite markov decision processes," in *Proceedings of the 41st IEEE Conference on Decision and Control*, 2002.