Towards Optimal Risk-Aware Security Compliance of a Large IT System

Daniel Coffman[1], Bhavna Agrawal[2], and Frank Schaffa[2]

[1] Walker Digital LLC, Stamford, CT, USA
DanielMark.Coffman@walkerdigital.com
[2] IBM T. J. Watson Research Center, Yorktown Heights, NY, USA
{bhavna,schaffa}@us.ibm.com

Abstract. A modern information technology (IT) system may consist of thousands of servers, software components and other devices. Operational security of such a system is usually measured by the compliance of the system with a group of security policies. However, there is no generally accepted method of assessing the risk-aware compliance of an IT system with a given set of security policies. The current practice is to state the fraction of non-compliant systems, regardless of the varying levels of risk associated with violations of the policies and their exposure time windows. We propose a new metric that takes into account the risk of non-compliance, along with the number and duration of violations. This metric affords a risk-aware compliance posture in a single number. It is used to determine a course of remediation, returning the system to an acceptable level of risk while minimizing the cost of remediation and observing the physical constraints on the system, and the limited human labor available. This metric may also be used in the course of the normal operation of the IT system, alerting the operators to potential security breaches in a timely manner.

Keywords: Risk-aware compliance, cloud computing, compliance metrics, compliance optimization.

1 Introduction

Modern information technology (IT) systems are large, and disparate. They may consist of thousands of servers, software components, networks, and other devices. They may be located in one or several data centers. An IT system may contain resources owned by a number of different organizations or individuals, but managed by a single entity. Increasingly, the server systems may not even be physical systems themselves, but may be hosted on a number of cloud servers; we designate the server systems as endpoints to distinguish them from the cloud servers. Fig. 1 illustrates an IT system consisting of n_s endpoints, under the control of n_o operators. Additionally, there might be a hierarchy of endpoints belonging to different clusters or clients in a complex delivery center.

The proper operation of an IT system may be interrupted for a number of reasons. Among them are hardware and software failures, resource limitations and malicious

S. Basu et al. (Eds.): ICSOC 2013, LNCS 8274, pp. 639–651, 2013.
© Springer-Verlag Berlin Heidelberg 2013

attacks. The former will be addressed through the adoption of monitoring and best practices, but the latter problems related to a system's security, require special scrutiny. Most often, the managing entity will enumerate the potential threats to the system and will develop and implement a set of policies as a first step of protection from malicious attacks. Such policies may be common to all endpoints or unique to a particular set of endpoints.

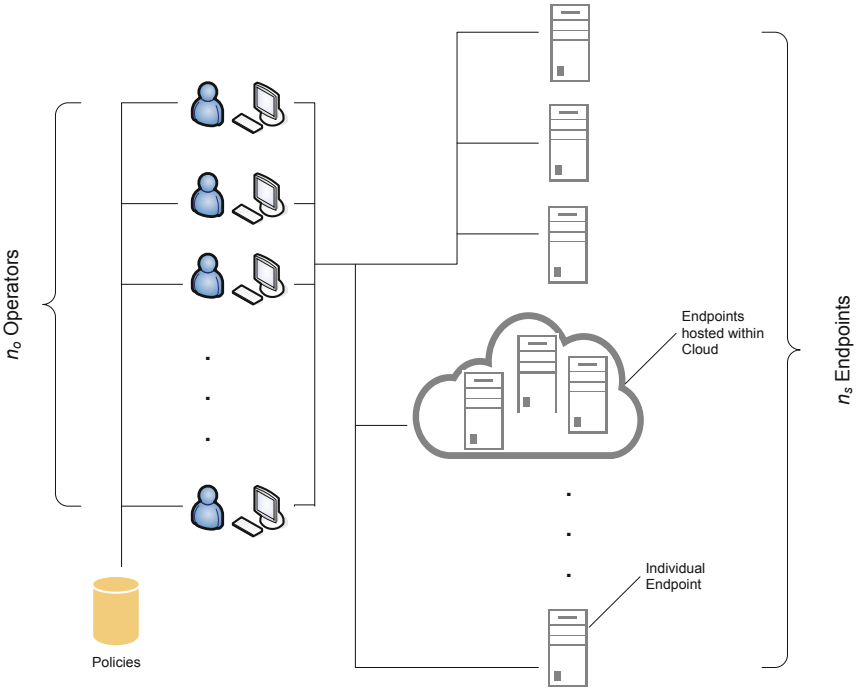

Fig. 1. An IT system comprising n_s endpoints under the control of n_o operators

It is the duty of the operators to ensure that each policy is respected on each relevant endpoint. Evidently, not all endpoints will simultaneously be in compliance with each policy. Frequently, there will be a contractual obligation that obliges the managing entity to maintain the endpoints at a certain level of compliance. However, there is no generally adopted method for measuring this level of compliance as also pointed out by authors in [1] and [2]. There are still more definitions available on the security metrics [3,4] but not many for compliance. Savola [5] defines the compliance metrics as a set of different factors, like number of policy exceptions requested/granted, cost of control, number of security incidence, elapsed time from incident identification to remediation etc., but it is not a single number.

The current practice for measuring compliance is for the operators to report the fraction of the endpoints not in compliance with one or more policies [1,6]. This approach, however, has at least three significant weaknesses. It ignores the fact that some violations of policy may be of a much more serious nature than others. Further it ignores the time-dependence of such a violation: some violations are initially not

terribly serious, but become much more serious as they are left unrepaired. Finally, some policies may yield only a single result on a single endpoint, whereas others may yield many results; this must be accounted for properly.

An example will prove illustrative. Consider the case of two policies: (i) that passwords must expire after 90 days; and (*ii*) that on a UNIX system, only the `root` user may write to the `/sbin` directory. Obviously, violations of policy *ii* are potentially much more serious than violations of policy *i*. However, checking all accounts on an endpoint according to policy *i* will yield Q_i responses where Q_i is the number of user accounts with one response per account. On the other hand, checking an endpoint for compliance with policy *ii* will yield only a single response. Finally, a password being unchanged for a short time after its expiration date will probably cause no harm, but the longer it remains unchanged, the greater the threat it represents that the endpoint's security will be compromised.

Once some violations of the policies have been observed, the system must be brought back into as compliant a state as possible. Again, there is no generally accepted way to accomplish this and only a few accounts of this in the literature. Levi [7] presents a method for generating a prioritized list of vulnerabilities, but he does not take into account the aggregation impact, which is when a large number of slightly lower risk vulnerabilities might pose a higher risk than a single high risk vulnerability. Additionally, he does not account for the impact of time for which these vulnerabilities are exposed as explained in the example above. Taraz [8] presents a method of computing the vulnerability score of a single device, but not for a large group of devices.

In the subsequent sections of this paper, we present a compliance metric that addresses these weaknesses. We develop a methodology using this metric for restoring the IT system to maximal compliance within given constraint of resources and cost. In addition, we show how this metric is used in the daily operations of the IT system.

2 Risk-Aware Compliance Metric

When calculating the rate of compliance of an IT system, the current practice ignores the different risks associated with different types of violations. However, such violations may be of very different characters in the risks they pose to the IT system. Evidently, some violations may be much more serious if they are left uncorrected for a substantial length of time. We propose a metric which takes these factors into account.

Consider a single policy k and a single endpoint l. We define $P(N_{kl}, Q_{kl}, R_k)$ as the probability that the endpoint is safe according to policy k. P is given by the binomial distribution:

$$P(N_{kl}, Q_{kl}, R_k) = \sum_{n=0}^{N_{kl}} \binom{Q_{kl}}{n} R_k^n (1-R_k)^{Q_{kl}-n} \tag{1}$$

Where

N_{kl} = number of responses indicating compliance with policy k on endpoint l
Q_{kl} = Total number of responses from checking policy k on endpoint l

R_k = Risk factor associated with policy k (also a probability of compromise for a single violation of policy k)

Here, $\binom{Q_{kl}}{n}$ are the usual binomial coefficients. The risk factors, R_k, are assigned values between 0 (lowest risk) & 1 (highest risk) based on the risk associated with the policy. For the paper, we do ad hoc assignment, but in practice they are assessed by the area specialists familiar with the various risks and policies. As an example, if the policy requires "password never expires box should be unchecked", there might be 10 user id's which are compliant, and 5 that are not, then the value of N will 10, and Q will be 15 for this endpoint and this policy. Note that for $Q=1,\ P(0,1,R)=1-R$, that is the probability of being safe (not being at risk) after the detection of a single violation.

Once we determine the probability of being safe from one policy violation on one endpoint, we can combine these to determine the compliance metric for a larger system of endpoints, governed by a number of policies. We define the risk-aware compliance metric, Λ, as the product of these probabilities for all endpoints, and all policies:

$$\Lambda = \prod_{k=1}^{n_p} \prod_{l=1}^{n_s} P(N_{kl}, Q_{kl}, R_k) \tag{2}$$

where n_p is the total number of policies being checked on each endpoint, and n_s is the total number of endpoints. This metric evaluates to 1 when all the endpoints are compliant with all the policies, and tends to zero with a large number of violations. Additionally, we note that Λ falls very rapidly with the number of detected violations (=Q-N, non-compliant messages) from its maximum value of 1, particularly for high risk violations. This is illustrated in Fig. 2, where this compliance metric is plotted against the number detected violations for policies with different risk factors.

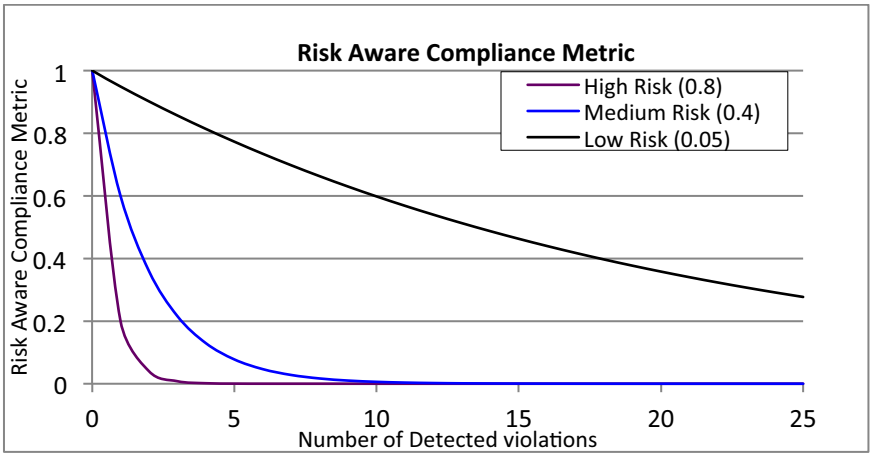

Fig. 2. Risk-aware Compliance Metric as a function of the number of detected violations for three different risk factors

The metric may be extended to incorporate the duration of a detected violation, that is, the time elapsed since the violating condition was first observed. We do this by modifying the risk factors associated with the policies to include this duration

$$R_k(t_i) = R_k + (1 - R_k)(1 - e^{-t_i/\tau_k}) \tag{3}$$

where t_i is the time for which the detected violation has remained unrepaired, and τ_k is the criticality time constant for this policy. Once again, the area specialists will be able to specify the criticality time constants based on their experience and knowledge. As we would expect, the risk factor increases from its original value to a maximum of 1 as the elapsed time increases, as depicted in Fig. 3a. Fig. 3b shows the corresponding change of Λ, which slowly goes to zero from its original value as time goes by. The results are plotted for a single detected violation for clarity of presentation, however, behavior is similar for multiple violations of multiple policies.

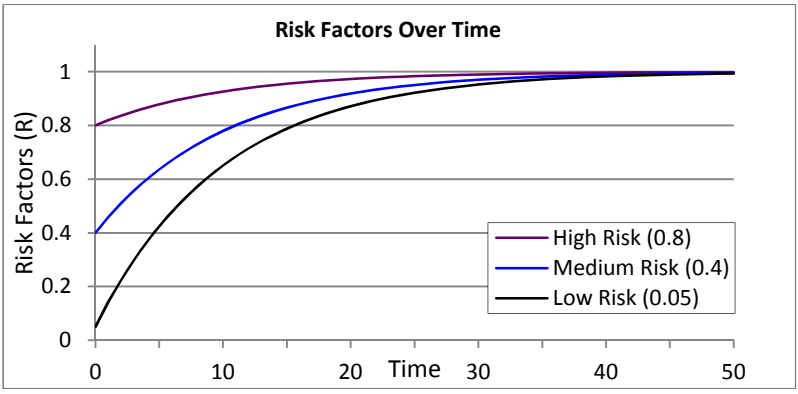

Fig. 3a. Risk factors as a function of time, for a time constant τ = 10 in arbitrary units

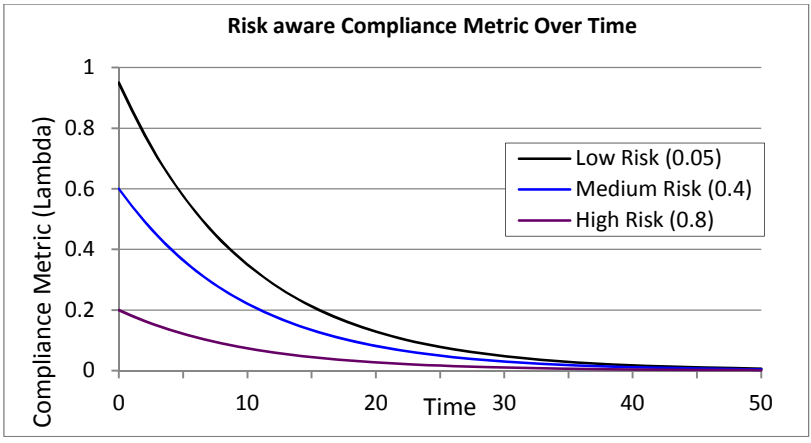

Fig. 3b. Compliance Metric as a function of time, for a single detected violation and three different risk factors

The risk-aware compliance metric is used in several different ways. It functions first as a "tripwire"; the metric is calculated periodically and a message is sent to the operators when the value of the metric falls below some predefined value. This typically indicates that some high risk violation has just been detected. Further, with the incorporation of the duration of observed violations, the value of the metric will degrade even if no new violations are reported. Finally, by observing the pattern of the measured values of the metric over time, it is possible to detect problems in the configuration of the IT system: e.g. if the metric exhibits considerable scatter and discontinuous behavior rather than varying smoothly over time, it is indicative of systematic problems.

3 Optimal Risk-Aware Compliance

Once an IT system becomes non-compliant, the observed violations will need to be remediated. In this section we describe an "optimal" way to bring system back to maximal compliance given system and human resource constraints. The system itself contains a finite set of resources such as network bandwidth. The endpoints also are constrained by say their available memory or CPU cycles. Each remediation will consume some portion of these resources. Similarly, each remediation will be accompanied by a certain *cost,* system and human. Finally, the remediation will be performed by human operators, each requiring a certain amount of time for each task and therefore a total time to complete all of their assigned tasks. The time required for the remediation is the longest of these total times. An optimal course of remediation will be accomplished in allowed time, at minimum cost, respecting all of the system resource constraints, and bringing the system back to maximum compliance.

We propose that the best remediation solution will maximize the risk-aware compliance metric, Λ, described in the last section while simultaneously minimizing the costs of such remediation and observing the constraints noted above. Let Δ be the set of remediations to be performed, and if Λ can be written as a function of Δ, then we can define the objective function as

$$\chi = -\alpha \ln(\Lambda(\Delta)) + C(\Delta) \tag{4}$$

where C is the total cost of remediation and α is an empirical, non-negative scale factor. Obviously *maximizing* $\Lambda(\Delta)$ is equivalent to *minimizing* $-\ln(\Lambda(\Delta))$. Hence, we seek to minimize χ subject to the constraints. The value of α is used to adjust the desired balance of risk and cost. We further define $\overline{N}$ as the initial number of responses indicating compliance and N is the number of final compliance messages. Hence, $\Delta = N - \overline{N}$.

The following subsections describe the cost and constraints for this objective function.

3.1 Cost

We use a linear approximation for defining cost, $C(\Delta)$,

$$C(\Delta) = \sum_{k=1}^{n_p} C_k \Delta_k \tag{5}$$

where n_p = number of policies, C_k is the cost to repair one violation of policy k and Δ_k is the total number remediations performed according to policy k, which can be further defined as:

$$\Delta_k = \sum_{j=1}^{n_o} \sum_{l=1}^{n_s} \Delta_{jkl} \tag{6}$$

where n_o is the number of operators, n_s is the number of endpoints, and Δ_{jkl} is the number of remediations performed by operator j according to policy k on endpoint l.

While we use a linear approximation of cost, this is not a strict requirement, and can be generalized without impacting the formulation and solution of the problem

The optimal system performance then consists in finding that set $\Delta = \{\Delta_{jkl}\}$ which maximizes $\Lambda(\Delta)$ while simultaneously minimizing $C(\Delta)$ subject to the constraints.

3.2 Constraints

For each of the system resource, *e.g.* CPU, memory, disk space, etc., there is a constraint on the maximum amount available. The performance of all remediations on a particular endpoint can never consume more than this amount. Furthermore, assuming that there is a maximum time allowed for all remediation work to be completed (e.g. total amount of time across all operators), the time required to perform the remediation must not exceed this maximum. We defined these two constraints below.

Resource Constraints

On endpoint l, there will be an amount of a resource of type m, say G_{ml}. The amount must be greater than the amount required by all of the desired remediations. That is

$$G_{ml} \geq \sum_{k=1}^{n_p} F_{km} \Delta_{kl} \tag{7}$$

$$\Delta_{kl} = \sum_{j=1}^{n_o} \Delta_{jkl} \tag{8}$$

where here, F_{km} is the amount of a resource of type m to required for one remediation according to policy k.

Time Constraints

Operator j will require a time t_j to complete his or her work. With the assumption that the individual tasks are completed sequentially, without interruption or overlap, this time is given by

$$t_j = \sum_{l=1}^{n_s} \sum_{k=1}^{n_p} T_k \Delta_{jkl} \tag{9}$$

where T_k is the time required to work on policy k.

The amount of time to complete all the remediations is obviously given by the longest such time. If the maximum allowable time for all remediations is T then we require

$$T \geq \max_j (t_j) \tag{10}$$

3.3 Optimization Function

Using the definition of cost and constraint functions defined above, the objective (minimization) function in Eq (4) can be re-written as (employing the standard technique of combining constraints with objective function using Lagrange multipliers):

$$\chi = -\alpha \ln(\Lambda(\Delta)) + \sum_{k=1}^{n_p} C_k \Delta_k + \sum_{m=1}^{n_R} L_m \sum_{l=1}^{n_s} (G_{ml} - \sum_{k=1}^{n_p} F_{km} \Delta_{kl}) + \sum_{j=1}^{n_o} \mu_j (T - \sum_{l=1}^{n_s} \sum_{k=1}^{n_p} T_k \Delta_{jkl}) \tag{11}$$

Where n_R is the number of resource types, and L_m and μ_j are Lagrange multipliers. For computational tractability, we approximate the first term. First, by definition (dropping subscripts for simplicity of presentation)

$$\ln(\Lambda) = \ln(\prod P(N,Q,R)) = \sum \ln(P(N,Q,R)) \tag{12}$$

Now consider

$$\ln(P(N,Q,R) = \ln(\sum_{n=0}^{N} \binom{Q}{n} R^n (1-R)^{Q-n}). \tag{13}$$

In order to make the objective function more tractable, we approximate P by $\hat{P}$ (a quadratic function) where

$$-\ln(\hat{P}(N,Q,R)) = C(Q-N)^2. \tag{14}$$

We fix the value of C by requiring

$$-\ln(\hat{P}(0,Q,R)) = -\ln(P(0,Q,R)). \tag{15}$$

Now

$$-\ln(\hat{P}(N,Q,R)) = \frac{-\ln(1-R)}{Q}(Q-N)^2 \tag{16}$$

The good agreement between the exact and approximate values is illustrated in Fig. 4. We also note that even with larger values of Q, the difference between the exact and approximate values is larger, but the agreement still holds.

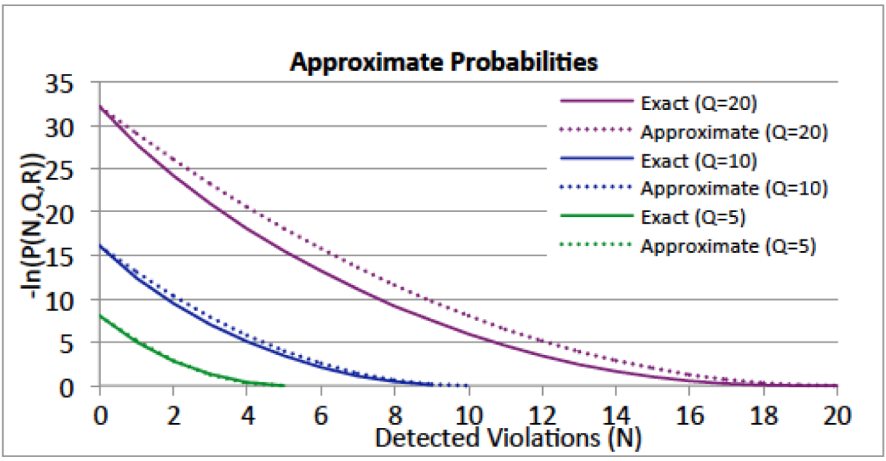

Fig. 4. Exact and Approximate probabilities for a range of Q values

With this approximation, the objective function is quadratic in Δ and may be minimized as an integer programming problem using any of the several optimizer packages including CPLEX [9] and Gurobi [10].

4 Discussion

The optimization problem described above is an integer programming (IP) problem with a quadratic objective function. It is well-known that such a problem is difficult to solve exactly. However, for rather small sets of policies, and endpoints with only a few operators, the problem is computationally tractable.

We present results first using an exact IP optimization for a system with two operators ($n_o = 2$), three policies ($n_p = 3$) and four endpoints ($n_s = 4$). Further, violations of the policies are of high, medium and low risk, respectively. The costs are taken to be 10, 5 and 5 for remediations of these policies, with each remediation taking a time of 8, 5 and 2, respectively; these values are obviously arbitrary, but can easily be mapped to a real values. This system will require the determination of 24 values for its complete optimization. These values are given in Fig. 5a for an optimization with roughly equal weights given to the risk-aware metric and the cost function and no constraint on the total time.

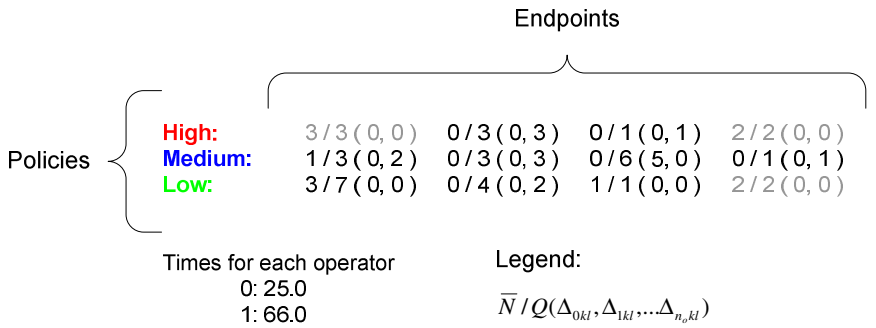

Fig. 5a. Optimal solution without time constraint. The first cell in row 2, 1/3(0,2), implies that for Medium risk policy on endpoint 1, there were originally 1 out of 3 messages indicating compliance, i.e. there were 2 deviations. For the optimal solution, operator 1 should remediate 0 and operator 2 should remediate 2 deviations. Note that the cells in gray have no violations.

The solution obtained through the optimization procedure above is in fact a course of action. It represents the steps to be taken to render the IT system maximally compliant at minimum cost within the allotted time. The result in the second cell of the first row, for example means that operator 2 should perform 3 remediations of High risk policy on endpoint 2. It is interesting to note that without the time constraint, the remediations assigned to the operators require quite different times. Imposing a constraint that the total time be less than or equal to 45 yields a somewhat different solution, as illustrated in Fig. 5b. Now, the effort required of both operators is almost exactly the same, whereas one fewer low risk violation can now be remediated.

This simple model may be explored further to reveal the dependence of the risk-aware metric on the allotted time. Naturally, as less time is available to complete the work, less can be done to bring the system into compliance and the metric will achieve a lower value at the optimum point. This is shown in Fig. 6.

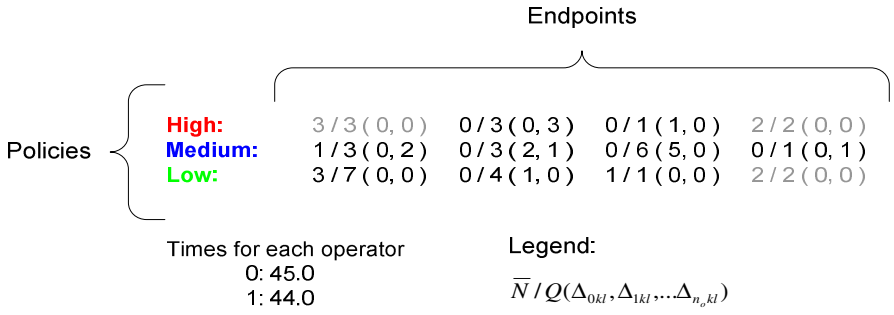

Fig. 5b. Optimal solution with total time required to be less than 45

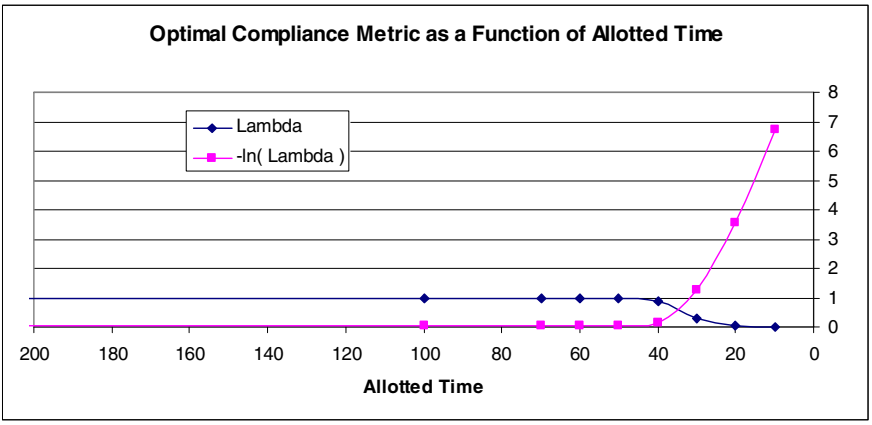

Fig. 6. Value of Risk-aware Compliance Metric at Optimum Point as a function of the time allotted for remediation. Also shown is the negative logarithm of this quantity. Note that if all violations were remediated by a single operator, this would require a time of 108, on this scale.

Also of interest is the appropriate value of α, the relative weight of the risk-aware metric and the cost function in the objective function. Again, we can examine the simple model here for some guidance. Let $\alpha = \max(C)/\max(-\ln(\Lambda)) * w$; clearly w, the *weight* factor, should be of order unity. We optimize our model for a number of values of this weight factor, with results presented in Fig. 7. At values greater than one, the risk-aware metric dominates the objective function, while for values less than one, the cost function is dominant; In this latter regime, the metric's value plummets as it is no longer cost-effective to remediate even high risk violations. We assert that value of about 4 offers a good balance of cost and risk for this simple model.

The optimal solution pinpoints the resource or resources that constrain the performance of the system overall. As noted above, if there is inadequate time available, the

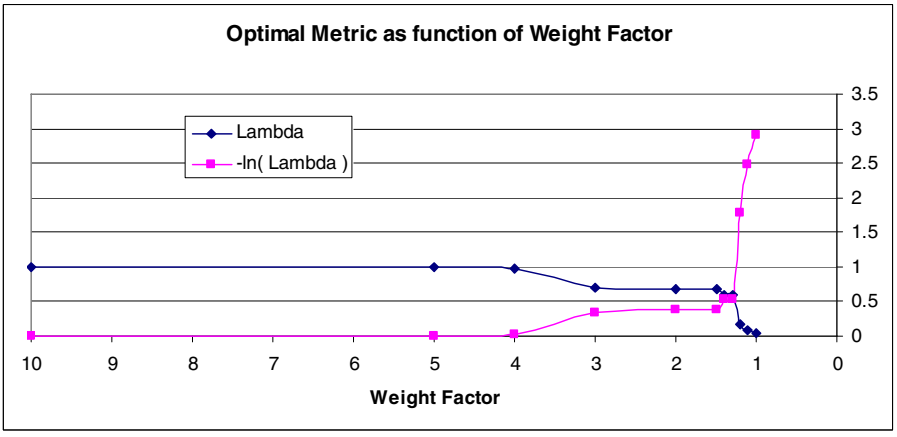

Fig. 7. Value of the Risk-aware Compliance Metric at the optimum point, as a function of the weight factor, $w = \alpha \max(-\ln(\Lambda))/\max(C)$

value of the risk-aware metric will be greatly reduced; thus, the optimization procedure will enumerate the benefit to be obtained through the hiring of more personnel. Similarly if, for example, one endpoint would benefit from additional RAM, that fact will be reflected in the solution.

4.1 Extension to Large Systems

The IP techniques used so far are unfortunately not suitable for problems with very large numbers of variables. Solving the simple model above requires only a few seconds on a modern computer, but even a modest extension to 10 endpoints and 4 operators requires tens of minutes for its solution. Fortunately, for this work, a provably optimal solution is desirable but not necessary. Therefore, it is reasonable to relax the constraint on the variables that they be integers and employ the machinery of a linear programming (LP) optimizer. *Ex post facto*, we can apportion the results among the operators if necessary.

Consider a model with 5 operators, 5 policies for high risk violations, 10 for medium risk and 6 for low risk, and altogether 1000 endpoints. Using the same objective function and constraints, this model may be optimized in just a few seconds with LP. In this first investigation, we relax the time constraints. The results are shown in Fig. 8, in the same format used for the smaller model, solved through IP. We note that the results are sensible --- the high risk violations are remediated in preference to the lower risk violations --- and so on.

The difficulty with using LP techniques comes exactly when imposing the constraints. When the time constraint comes into play, the solution will sometimes favor equal division of the work among the operators, say each being assigned 20% of a task. Thus, we cannot adopt the LP solution without further work in parceling out the work. However, this is not a worrisome state of affairs since the number of

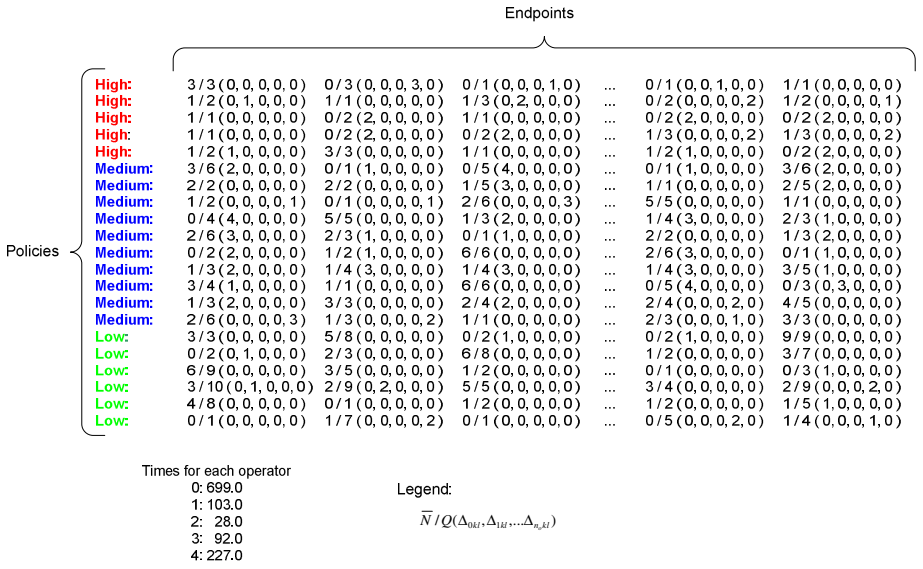

Fig. 8. Extract of optimal results for a large model, solved using LP techniques

remediations required is normally quite large, in the tens of thousands, so that the exact distribution of tasks is immaterial. The LP solution will, however, provide reliable guidance on the exact mixture of tasks, that is remediations of high, medium and low risk violations, that will lead to an optimally compliant system.

5 Further Work

There are several facets of this work which will be explored further. As mentioned above, this work approximates the total cost of remediation with a simple linear function of the number of remediations performed. We wish to examine how this may be relaxed without compromising the numerical properties desired for the optimization.

As mentioned above, we will examine techniques for apportioning the results from LP optimization and adopt one which is numerically tractable and theoretically defensible. Further, we will employ this technique on real data collected from data centers to aid in the improvement of the compliance of the systems they support.

References

1. Jansen, W.: Directions in security metrics research. National Institute of Standards and Technology, NISTIR 7564 (2010)
2. Julisch, K.: Security compliance: the next frontier in security research. In: Proceedings of the 2008 Workshop on New Security Paradigms, pp. 71–74. ACM (2009)
3. First.org. A Complete Guide to the Common Vulnerability Scoring System Version 2.0 - CVSS, http://www.first.org/cvss/cvss-guide
4. Pironti, J.P.: Developing Metrics for Effective Information Security Governance. INTEROP, New York (September 2008), http://www.interop.com/newyork/2008/presentations/conference/rc10-pironti.pdf
5. Savola, R.: Towards a security metrics taxonomy for the information and communication technology industry. In: International Confernce on Software Engineering Advances, ICSEA, Cap Estrel, France (August 2007)
6. Herrmann, D.S.: Complete guide to security and privacy metrics: measuring regulatory compliance, operational resilience, and ROI. CRC Press (2007)
7. Levi, E.: Device, Method and Program Product for Prioritizing Security Flaw Mitigation Tasks in a Business Service. U.S. Patent Application 12/361,279, Filed (January 28, 2009)
8. Taraz, R.: Method and apparatus for rating a compliance level of a computer connecting to a network. U.S. Patent Application 11/289,740, Filed (November 29, 2005)
9. Cplex, IBM ILOG. 12.5 User's Manual (2010), ftp://public.dhe.ibm.com/software/websphere/ilog/docs/optimization/cplex/ps_usrmancplex.pdf
10. Optimization, Gurobi. Gurobi optimizer reference manual (2012), http://www.gurobi.com