A Metadata Calculus for Secure Information Sharing

Mudhakar Srivatsa
IBM T.J. Watson Research Center
msrivats@us.ibm.com

Dakshi Agrawal
IBM T.J. Watson Research Center
agrawal@us.ibm.com

Steffen Reidt
University of London
sreidt@rhul.ac.uk

ABSTRACT

In both commercial and defense sectors a compelling need is emerging for rapid, yet secure, dissemination of information to the concerned actors. Traditional approaches to information sharing that rely on security labels (e.g., Multi-Level Security (MLS)) suffer from at least two major drawbacks. First, static security labels do not account for tactical information whose value decays over time. Second, MLS-like approaches have often ignored information transform semantics when deducing security labels (e.g., output security label = max over all input security labels). While MLS-like label deduction appears to be conservative, we argue that this approach can result in both underestimation and overestimation of security labels. We contend that overestimation may adversely throttle information flows, while underestimation incites information misuse and leakage.

In this paper we present a novel calculus approach to securely share tactical information. We model security metadata as a vector half-space (as against a lattice in a MLS-like approach) that supports three operators: $\gamma$, $+$ and $\cdot$. The value operator $\gamma$ maps a metadata vector into a time sensitive scalar value. The operators $+$ and $\cdot$ support arithmetic on the metadata vector space that are homomorphic with the semantics of information transforms. We show that it is unfortunately impossible to achieve strong homomorphism without incurring exponential metadata expansion. We use B-splines (a class of compact parametric curves) to develop concrete realizations of our metadata calculus that satisfy weak homomorphism without suffering from metadata expansion and quantify the tightness of values estimates in the proposed approach.

Categories and Subject Descriptors
C.2.0 [General]: Security and protection

General Terms
Security, Management

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first Page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

CCS’09, November 9–13, 2009, Chicago, Illinois, USA.
Copyright 2009 ACM 978-1-60558-352-5/09/11 ...$10.00.

Keywords
Information Flow, Access Control, Risk Management

1. INTRODUCTION

In both commercial and defense sectors a compelling need is emerging for rapid, yet secure, dissemination of information to the concerned actors. For example, in a commercial setting, the ability of multiple partners to come together, share sensitive business information and coordinate activities to rapidly respond to business opportunities is becoming a key driver for success. Similarly, in a military setting, traditional wars between armies of nation-states are being replaced by highly dynamic missions where teams of soldiers, strategists, logisticians, and support staff, fight against elusive enemies that easily blend into the civilian population [18]. Securely disseminating mission critical tactical intelligence to the pertinent people in a timely manner will be a critical factor in a mission’s success.

While secure and timely information dissemination is increasingly becoming a necessity, it is important for the sender to assess the value of information being shared with the recipient. However, there are two important challenges in quantifying the value of information [14]. First, overestimation severely constrains information flow, thereby either preventing or delaying the delivery of information to the right people at the right time. Second, underestimation may incite information misuse and leakage, thereby exposing sensitive information to the wrong people.

We contend that traditional approaches to information sharing fail to strike a good balance between these two conflicting forces for the following reasons.

First, traditional approach to information sharing use fairly static security labels to tag information, and thus do not attempt to capture dynamic attributes of information such as time sensitivity; for example, the value of tactical information (e.g., consumer demand, insurgent locations) decays over time, while that of strategic information (e.g., business secrets, nuclear technology) remains sensitive over extended durations of time. The value of a 'piece of information' (henceforth, called an object) is computed as a function of its security labels (henceforth, called metadata). For example, Multi-Level Security (MLS) uses security labels such as unclassified < confidential < secret < top-secret; Decentralized Label Management (DLM) labels each object with allow (A) and deny (D) lists and regulates information flows based on these labels.

Second, we contend that traditional approaches to information sharing use an improper approach to deduce the
metadata for an output object $x$ derived from input objects $y_1, y_2, \ldots, y_n$ (see Figure 1). For example, MLS computes the security label for an output object as a max over all the security labels of the input objects. Such approaches do not account for information transforms that explicitly down-grade an object or upgrade a collection of objects. For example, a set of numeric objects may be statistically downgraded using their mean; an image may be downgraded by lowering its resolution, smoothness factor, etc. On the other hand, let us consider an aggregation operation that reconstructs a secret $x$ (output) from a collection of its secret shares $\{y_i\}$ (input) using a standard $k$-out-of-$n$ secret sharing scheme. A MLS-like approach would deduce that the security label of output $x$ is equal to that of $y_i$, when in reality the individual secret shares $(y_i)$ have almost no value and the secret $(x)$ may be highly valuable. Hence, using MLS-like approach, we are forced to either overestimate the values of $y_i$ or underestimate the value of $x$. These examples clearly allude to the lack of information transform semantics which impedes a MLS-like approach from distinguishing between two fundamentally different operations, such as, computing a mean as opposed to aggregating secret shares.

In this paper we present a novel metadata calculus for securely sharing tactical information. We capture an object’s metadata using a vector half-space $M$ (as against a lattice in MLS-like approaches) and define a metadata calculus that supports the following primitives: the value operator $\Gamma$ that maps an object’s metadata to a scalar time varying value, and arithmetic operators + and $\cdot$ support metadata deduction (that is aware of information transform semantics) on the vector space $M$. We use an information theoretic approach to empirically deduce the value of an output object as a function of the value of input objects and the transformation function $g$. The key idea here is to use the notion of information loss [8] to quantify what fraction of information in an input object $y_i$ that is carried over to the output object $x = g(y_1, \ldots, y_n)$, and the notion of information gain [8] to quantify the boost in overall information content when two or more input objects are fused with one another. We argue that our empirical formula satisfies several intuitive properties (e.g., non-negativity, monotonic decay with time, etc) using information transform functions ranging from statistical summaries (e.g., min, max, average), database operations (e.g., join, select, project), and cryptographic functions (e.g., encryption, decryption, signature).

Based on the empirical value function, we postulate homomorphic properties that need to be satisfied by the + and $\cdot$ operators on the metadata vector space. These homomorphic properties allow us to derive the metadata vector of an output from the metadata vector(s) of input object(s) based on the transform $g$. Unfortunately, the non-negativity constraint on the value function restricts our metadata vector space to a half-space [6]. We recognize that it is impossible to achieve strong homomorphism without incurring metadata expansion\(^1\); in particular, we show that satisfying strong homomorphism may result in exponential metadata expansion. We propose a weak homomorphic property that relaxes the tightness of our value estimates. We use B-splines (a class of compact parametric curves) to develop concrete realizations of our metadata calculus that satisfy weak homomorphism without suffering from metadata expansion and quantify the tightness of values estimates in the proposed approach.

The rest of this paper is organized as follows. Section 2 describes related work on risk based secure information flows. Section 3 describes our metadata model and presents an information theoretic approach to estimating the value of an output object obtained by either upgrading or down-grading one or more input objects. Section 4 describes a calculus for succinctly computing the metadata for an output object emphasizing on strong and weak homomorphic properties and metadata expansion. We discuss the limitations of our proposal and open problems in Section 5 and finally conclude the paper in Section 6.

2. RELATED WORK

2.1 Secure Information Flows

There has been significant research on decentralized information labels and assured information sharing within and across multiple organizations [11, 12, 13, 25, 26, 27] in recent years. However, these works primarily focus on the problem of specifying and manipulating the sharing, propagation and downgrading constraints on data. These works also assume appropriate security controls that manipulate, bind and respect these labels are already in place, for example, via a secure distributed runtime language, or some other form of a secure distributed trusted computing base. Clearly, in practice, for the settings described in the introduction, one partner cannot be sure of either the existence, or the proper usage of, a secure runtime environment of another partner. Recently, new approaches based on risk estimation and economic mechanisms have been proposed for enabling the sharing of information in dynamic environments [5, 14]. These approaches are based on the idea that the sender dynamically computes an estimate of the risk of information disclosure when providing information to a receiver based on the secrecy of the information to be divulged and the sender’s estimate on the trustworthiness of the recipient. The sender then charges the recipient for this estimated risk. The recipient, in turn, can decide which type of information is most useful to him and pay only to access those pieces of information. Entities would either be given a line of risk credit, or adopt a market-based mechanism to purchase risk using a pseudo-currency. Under the assumption that the line of risk credit or the risk available for purchase in the market is limited, an entity will be encouraged to be frugal with their amassed risk credits and consequently, reluctant to spend them unnecessarily. Since all information flows are charged against expected losses due to unauthorized disclosure and

\[ MLS \text{ uses constant size metadata; however, DLM may suffer from metadata expansion.} \]
the amount of risk available is limited, an argument is made that the total information disclosure risk incurred by an organization is controlled.

While as a concept, using risk estimation, charging for risk of information flows, and limited risk credits are promising ideas for enabling information sharing in dynamic environments, the existing work in this area [5, 14] has gaps in how this concept can be realized to enable secure information flows in dynamic tactical environments. In [5, 14, 22, 23], while risk is estimated based on the object metadata [24], the actual formulas or examples use static credentials (e.g., the security clearance or category set) of the shared object and the recipient, rather than a dynamic value of the shared object. Further, they do not provide solutions to either deduce the metadata or estimate the value of new or derived objects. Indeed, the value of most tactical information tends to decrease with time and evolves as the object is downgraded and fused with other objects. In this paper we present a novel metadata calculus that can be used to succinctly represent security metadata and estimate the time varying value of tactical information in a dynamic setting.

2.2 Security Calculus

Abadi and Gordon [1] have proposed a calculus for formally analyzing cryptographic protocols. Their calculus extends the popular pi-calculus [19] that was originally designed for describing process oriented systems and protocols. Abadi and Gordon introduced cryptographic primitives as first class entities into the pi-calculus allowing them to describe and analyze cryptographic protocols. Type calculus (e.g., λ-calculus [17]) have long been used for representing abstract data types and reasoning about object mutations. Other authors have proposed a calculus for composing security complex policies from simpler ones [3]. This is accomplished by parameterizing policies with values or other policies and supporting operators for conjunction and disjunction of policies. While some of these calculi are quite mature and expressive, a metadata calculus geared towards secure information sharing (e.g., MLS) is highly rigid and inaccurate (underestimation and overestimation). In this paper we propose a novel metadata calculus that automatically allows us to deduce security metadata for derived objects in a dynamic setting.

3. METADATA MODEL

In this section we introduce our metadata model to facilitate secure and timely information dissemination. For the sake of simplicity we include only one dynamic attribute, namely time, in our metadata model. We represent the metadata for an object \( a \) as a vector \( \vec{x} \) in a vector space \( \mathcal{M} \). \( \mathcal{M} \) supports a unary operator \( \Gamma \) that maps a metadata vector \( \vec{x} \) to a time varying value function. \( \mathcal{M} \) also supports two binary operators + and \( \cdot \); the + operator denotes vector addition (say, \( \vec{x} + \vec{y} \)) and the \( \cdot \) operator denotes scalar multiplication (say, \( a \vec{x} \)). In the rest of this section, we will study the unary value function in great detail. We note that the + and \( \cdot \) operators are artifacts of the value operator \( \Gamma \) that support homomorphic operations over the metadata vector space.

3.1 Value Operator \( \Gamma \)

The metadata vector space \( \mathcal{M} \) supports an unary operator that maps \( \vec{x} \in \mathcal{M} \) to value function \( \Gamma: \mathcal{M} \rightarrow (\mathcal{F} \rightarrow \mathcal{F}) \), where \( \mathcal{F} \) denotes the set of all integers or real numbers. For example, \( \vec{x} = (10, 2) \) and \( \Gamma(t) = \max(10 - 2t, 0) \) (linear decay function) or \( \Gamma(t) = 10^{-2t} \) (exponential decay function). We note that the absolute value of an object at any time instant \( t \) is by itself meaningless; what is of interest is the relative value of two objects at the time \( t \) or the relative value of an object at two different time instances \( t_1 \) and \( t_2 \). We assume that the value operator \( \Gamma \) satisfies the following properties:

\[
0 \leq \Gamma(t) < \infty, \forall t
\]

\[
x \subseteq y \Rightarrow \Gamma(t) \leq \Gamma(y)(t), \forall t
\]

\[
\frac{\partial \Gamma(t)}{\partial t} \leq 0, \forall t
\]  

(1)

The first constraint restricts the value of any object to non-negative numbers in \( \mathcal{F} \). We note that there may be exceptions. For example, sources of disinformation (misguiding information) may be associated with a negative value \([0, 20]\)^2. Indeed sources of disinformation have been used a psychological tool that instills uncertainty in military and business settings. In this paper, we do not consider pieces of information that are intended to misguide the recipient. In the absence of disinformation the value operator is monotonic, that is, we assume that adding additional information never decreases value. In particular, if an object \( x \) is completely contained in object \( y \), then \( \Gamma(t) \leq \Gamma(y)(t) \), for all \( t \). Finally, the third constraint simply states that the value of tactical information decays with time. Note that this definition subsumes non-tactical information by setting the decay rate \( \frac{\partial \Gamma(t)}{\partial t} \) to zero.

3.2 Value Arithmetic

Having characterized the properties of the value operator, we examine an information theoretic approach to estimate the value of an output object given the value of input objects and the transformation function. Let us suppose that \( x = g(y_1, y_2, \ldots, y_n) \). We assume that the value of input objects \( \Gamma(y_i)(t) \) and the transformation function \( g \) is known. The key idea here is as follows. Let us suppose that recipient’s prior knowledge about \( y_i \) is captured by a probability distribution function. Assuming that the recipient knows the function \( g \) and the output object \( x \), the recipient can compute a posterior distribution for \( y_i \). In the absence of disinformation, the posterior distribution is guaranteed to be statistically closer to the true distribution of \( y_i \) than the prior. Now, the information gain between the posterior and prior distribution of \( y_i \) denotes information about \( y_i \) that is indirectly learnt by the recipient when it receives \( x \). Intuitively, the value of \( x \) should be positively correlated with the product of the value of \( y_i \) and the information gain on \( y_i \) given \( x \) [2].

\[
\Gamma(x) = \sum_{i=1}^{n} \Gamma(y_i)(t) \cdot f_X(x, y_i|x_i, B) / f_X(x|y_i)
\]  

(2)

Based on this intuitive idea, we propose a candidate empirical formula to compute the value of the object \( x \) as shown in Equation 2, where \( f_X \) denotes the probability distribution function for a random variable \( X \), \( \overline{Y} = \{Y_1, \ldots, Y_{1-i}, Y_{i+1}, \ldots, Y_n\} \) and \( \overline{X} = \{y_1, \ldots, y_{1-i}, y_{i+1}, \ldots, y_n\} \). We use B
to denote background information known to the recipient of object \( x \) (detailed examples follow in Section 3.3).

Value computation uses the notion of self-information expressed as \( I(y_i|x) = D(\delta_{y_i} || f_{y_i|x}(y_i|x)) = -\log f_{y_i|x}(y_i|x) \), where \( D(X \parallel Y) \) denotes KL-divergence between probability distributions \( X \) and \( Y \) and \( \delta_{y_i} \) denotes the Dirac delta function whose value is one when \( Y_i = y_i \) and zero otherwise [15]. Intuitively self-information \( f(y_i|x) \) denotes the number of additional bits that need to be learnt in order to exactly reconstruct \( y_i \) given \( x \). Hence, \( 2^{-I(y_i|x)} = f_{y_i|x}(y_i|x) \) denotes the fraction of information about \( y_i \) that may be inferred from \( x \).

The probability distribution function \( f_{y_i|x} \) depends on the transformation function \( g \). For a simple function \( x = y_1 + 1 \), then the probability distribution function \( f \) is as follows: \( f(y_1 = x - 1|x) = 1 \) for all \( x \); \( f(y_1|x) = 0 \) for all \( y_1 \neq x - 1 \) and for all \( x \). In several cases, the transformation function \( g \) may be quite complex making it a challenging problem to determine the probability distribution function \( f \). For example, \( g \) may be a join operation over two database tables \( y_1 \) and \( y_2 \); \( x = y_1 \times y_2 \) and the prior distributions of \( y_1 \) and \( y_2 \) may be unknown. It gets more interesting when we examine cryptographic functions: \( x = H(y_1) \), where \( H \) is a computationally irreversible hash function. In this case, \( f(y_1 = H^{-1}(x)|x) = 1 \); and zero for all other \( y_1 \); however, computing \( H^{-1} \) may be computationally infeasible. Hence, although \( H^{-1} \) exists and information theoretically, \( \Gamma \bar{x}(t) = \Gamma \bar{y_1}(t) \), one may choose to override this definition using the computational assumption and set \( \Gamma \bar{x}(t) = 0 \), namely, the recipient cannot infer (almost) anything about \( y_1 \) given \( x \).

Figure 2: Value Computation for Arithmetic Functions

Figure 3: Value Computation for Database Functions: Assume, we have two database fields \( Y_1 \) and \( Y_2 \). The values \( y_1 \) in \( Y_1 \) and \( y_2 \) in \( Y_2 \) are independent and chosen using a uniform distribution between \( 0, 2^k - 1 \)

3.3 Value Arithmetic: Examples

In this section we argue that the empirical Equations 2 and 3 satisfy the intuitive notions of object downgrade and upgrade transforms. In the rest of this section, we demonstrate the applicability of Equation 2 to a wide range of functions \( g(\cdot) \) ranging from arithmetic functions, database operations and cryptographic functions. Figure 2, 3 and 4 show value arithmetic for some sample functions \( g \) from various application domains. Recall that \( B \) denotes the background information known to a recipient of object \( x \).

Arithmetic Functions and Statistical Summaries: Figure 2 shows value arithmetic for simple arithmetic and statistical functions. In the case of bijective functions (functions that have a unique inverse: such as \( x = g(y_1) = y_1 + 1 \)), we note that given \( x \) and the function \( g \), one can completely recover all information about \( y_1 \). Hence, the value of \( x \) equals the value of \( y_1 \) for all time instances \( t \). On the other hand, arithmetic functions such as \( x = y_1 \) lose information on \( y_1 \); in particular, given \( x \) one can identify two possible values for \( y_1 \) (namely, \( \pm \sqrt{x} \)). In the absence of any background information on \( y_1 \), this results in an entropy loss of one bit; equivalently the value of \( x \) is half the value of \( y_1 \). However, if the recipient were to know that \( y_1 \geq 0 \), then there is no loss of information.

Functions such as sum and average may exhibit different information loss characteristics. For example, let us suppose \( x = y_1 + y_2 \). If the recipient knows that \( 0 \leq y_1, y_2 \leq 5 \), then given \( x = 0 \) (or 10), it can obtain all information about \( y_1 \) and \( y_2 \) respectively; hence, \( \Gamma \bar{x}(t) = \Gamma \bar{y_1}(t) + \Gamma \bar{y_2}(t) \)
when \( x = 0 \) and \( y_1 = y_2 = 0 \) or when \( x = 10 \) and \( y_1 = y_2 = 5 \). On the other hand, if the recipient knows that \( y_1 \) and \( y_2 \) are uniformly and randomly distributed between 0 and 5, then given \( x = 5 \) does not reveal any additional information to the recipient; hence \( \Gamma x(t) = 0 \) when \( x = 5 \). In general, the value of the output object \( x \) as defined by Equation 2 depends on the value of the input objects \( y_1 \) and the background information available at the recipient. These examples highlight the ability of our metadata calculus in handling information downgrade transforms.

**Database Operations:** Aggregation over two or more objects may boost the value of the output over and beyond the input objects. We clarify this concept using two types of database operations: union and join. Union operates on two sets of the same type; for example, let us consider sets of type color \( y_1 = \{\text{red}\} \) and \( y_2 = \{\text{blue}\} \). Consequently, union of two sets does not result in any additional information than the input sets. On the other hand, join operates on two sets of different types; for example, let us consider a set of type (x-coord, id) \( y_1 = \{(10, id_1)\} \) and a set of type (y-coord, id) \( y_2 = \{(5, id_2)\} \). A join on \( y_1 \) and \( y_2 \) reveals the \((x, y)\) coordinates of the entity \((id_1)\) and thus has significantly more information than the sum of the value of the input sets.

To cite another example, let us consider a 128-bit cryptographic key \( K = L \parallel R \) of value \( V \), where \( L \) and \( R \) denote the left and the right 64-bits of a 128-bit key \( K \). Let us consider transformation functions \( lsplit \) and \( rsplit \) that take as input \( K \) and return \( L \) and \( R \) respectively. It is easy to see that using Equation 2, the value of \( L \) and \( R \) is \( 2^{-64} \cdot V \), assuming the key \( K \) is 128 bits long and it is randomly chosen. Now, let us consider a joint of sets of type \((L, \text{kid})\) and \((R, \text{kid})\), where \( \text{kid} \) denotes key identifier: \( y_1 = \{(L_1, \text{kid}_1)\} \) and \( y_2 = \{(R_1, \text{kid}_1)\} \). It is easy to see that the value of output object \( x \) must be significantly higher than the sum of values \( y_1 \) and \( y_2 \). In this simple example, the sum of the values of \( y_1 \) and \( y_2 \) is \( 2^{-64} \cdot V + 2^{-64} \cdot V = 2^{-63} \cdot V \); indeed this is a gross underestimation of a key that is valued at \( V \). Equation 2 captures the notion of information gain due to object aggregation using the factor \( f_{\chi \Gamma y}(x|\overline{y}) \).

In this case, \( f_{\chi \Gamma y}(x|\overline{y}) = f_{\chi \Gamma y}(K|R) = 2^{-64} \), similarly, \( f_{\chi \Gamma y}(x|\overline{y}) = f_{\chi \Gamma y}(K|L) = 2^{-64} \). Hence, on joining \( y_1 = \{(L_1, \text{kid}_1)\} \) and \( y_2 = \{(R_1, \text{kid}_1)\} \), Equation 2 amplifies the values of \( y_1 \) and \( y_2 \) by a factor \( f_{\chi \Gamma y}(x|\overline{y}) = 2^{64} \) when deriving the value of \( x \). These examples highlight the ability of our metadata calculus in handling information upgrade (aggregation) transforms.

**Cryptographic Functions and Computational Hardness:** For cryptographic functions, we relax the notion of information theoretic security and incorporate the notion of computational hardness. For most cryptographic functions, we override the information theoretic definitions assuming computational hardness holds. For example, using an ideal encryption function \( x = E_K(y_1) \), the value of \( x \) is zero if the recipient does not know key \( K \); otherwise, the value of \( x \) is equal to the value of \( y_1 \), since the recipient can recover all information about \( y_1 \) using the corresponding decryption function \( D \) and the key \( K \).

We note that exact value arithmetic on concrete implementations of cryptographic primitives such as MD5, SHA1, DES, AES, DSA, etc may be a challenging problem in itself. In this paper, we point out other interesting corner cases: \( x = H_K(y_1) \), where \( K \) is known to the recipient. Assuming that \( H \) is an ideal PRF, the computational cost of inferring \( y_1 \) given \( x \) and \( K \) depends on the size of the domain of \( y_1 \). For instance, if \( y_1 \in \{0, 1\} \), then one can trivially determine \( y_1 \) given \( x \) and \( K \) using at most one computation of \( H \); on the other hand, if \( y_1 \) is 128 bits long and randomly chosen, then it may be computationally infeasible to determine \( y_1 \) given \( x \) and \( K \).

In such cases, our metadata calculus permits users to specify the value function (say, based on the entropy of the domain of \( y_1 \)); for example, \( \Gamma x(t) \) may be computed as shown below. We use \( E(y) \) to denote the entropy of the domain of \( y_1 \).

\[
\Gamma x(t) = \begin{cases} \Gamma y_1(t) & \text{if } y_1(t) \leq 32 \\ \Gamma y_1(t) \cdot 2^{(E(y_1) - 32)} & 32 < E(y_1) \leq 64 \\ 0 & E(y_1) > 64 \end{cases}
\]

A good example of aggregation in cryptographic operations is secret sharing. Let us consider a function split that takes as input \( x \) and integers \( k \) and \( n \) and output \( n \) secret shares \( y_1, y_2, \ldots, y_n \) using a \( k \)-out-of-\( n \) secret sharing scheme. Then \( f(y_1|x) = 1 \), \( f(x|\varphi) = 0 \) if \( |\varphi| < k \); 1 if \( |\varphi| \geq k \), where \( \varphi \subseteq \{y_1, y_2, \ldots, y_n\} \). Plugging these distribution functions, one can meaningfully perform value arithmetic for function that fuses a set of secret shares of the type \((\text{share}, id)\), where \( id \) denotes the identity of the original object.

Finally, we note that it is trivial to encode value arithmetic for integrity check functions. However, we use a slightly modified version of the integrity check function as follows: \( x = \text{verify}(y_1, y_2) \) wherein the function \( \text{verify} \) returns \( y_1 \) if \( y_1 \) satisfies integrity check against a symmetric or public key

<table>
<thead>
<tr>
<th>( g )</th>
<th>( B )</th>
<th>( \Gamma x )</th>
<th>note</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x = K )</td>
<td>-</td>
<td>0</td>
<td>Key ( K ) has zero value; but knowledge of ( K ) affects the value of other objects</td>
</tr>
<tr>
<td>( x = E_K(y_1) )</td>
<td>-</td>
<td>0</td>
<td>Ideal encryption</td>
</tr>
<tr>
<td>( x = E_K(y_1) )</td>
<td>K</td>
<td>( \gamma y_1(t) )</td>
<td>Ideal encryption</td>
</tr>
<tr>
<td>( x = D_K(E_K(y_1)) )</td>
<td>K' # K</td>
<td>0</td>
<td>Ideal decryption</td>
</tr>
<tr>
<td>( x = H(y_1) )</td>
<td>-</td>
<td>0</td>
<td>Ideal Pseudo-Random Function</td>
</tr>
<tr>
<td>( x = H_K(y_1) )</td>
<td>K</td>
<td>?</td>
<td>Ideal Keyed Pseudo-Random Function</td>
</tr>
<tr>
<td>( x = \text{Sig}_K(y_1) )</td>
<td>-</td>
<td>0</td>
<td>Ideal signature</td>
</tr>
</tbody>
</table>

---

\( ^{\text{Under standard notation, verify outputs true/false; however, our metadata calculus associates value to an object and not Boolean literals true/false.}} \)
signature $y_2$ and random otherwise. Hence, $\Gamma x(t) = \Gamma y_1(t)$ if the integrity check succeeds and zero otherwise.

4. METADATA CALCULUS

We have so far described a metadata model that allows value estimation over dynamic attributes (e.g., time) while respecting the semantics of information transforms. In this section we will develop a metadata calculus for secure information sharing. The goal of this calculus is to support operators on the metadata vector space that are homomorphic to the value arithmetic proposed in Section 3. We describe a metadata calculus using two binary operators on the metadata vector space $M$: vector addition $+$, $\mathbb{M} \times \mathbb{M} \rightarrow \mathbb{M}$ and scalar multiplication $\cdot: F \times M \rightarrow M$. These binary operators satisfy the following homomorphic properties:

- **Commutative:** For any $\vec{y}_1, \vec{y}_2 \in \mathbb{M}$, $\vec{y}_1 + \vec{y}_2 = \vec{y}_2 + \vec{y}_1$.
- **Associative:** For any $\vec{y}_1, \vec{y}_2, \vec{y}_3 \in \mathbb{M}$, $\vec{y}_1 + (\vec{y}_2 + \vec{y}_3) = (\vec{y}_1 + \vec{y}_2) + \vec{y}_3$.
- **Distributive over $+$ in $\mathbb{M}$:** For any $a \in \mathbb{F}$, $\vec{y}_1, \vec{y}_2 \in \mathbb{M}$, $(a + b) \cdot \vec{y}_1 = a \cdot \vec{y}_1 + b \cdot \vec{y}_1$.
- **Distributive over $\cdot$ in $\mathbb{M}$:** For any $a, b \in \mathbb{F}$, $\vec{y}_1 \in \mathbb{M}$, $a \cdot (b \cdot \vec{y}_1) = (ab) \cdot \vec{y}_1$.
- **Scalar 1 in $\mathbb{F}$:** For any $\vec{y}_1 \in \mathbb{M}$, $1 \cdot \vec{y}_1 = \vec{y}_1$.

Based on the properties described above, it is easy to see that when $x$ is computed as $g(y_1, y_2, \cdots, y_n)$ then the metadata $\vec{x}$ can be computed as shown in Equations 4 and 5 for exact reconstruction and approximate reconstruction respectively. We note that $\sum$ in Equations 4 and 5 denotes vector addition $+$ over $n$ metadata vectors. Assuming the homomorphic properties of the $+$ and $\cdot$ operator, it is easy to see the Equations 4 and 5 are homomorphic with Equations 2 and 3 respectively.

$$\vec{x} = \sum_{i=1}^{n} \frac{f_{y_i}(x|y_i, B)}{f_{y_i}(x|B)} \cdot \vec{y}_i$$

$$\vec{x} = \sum_{i=1}^{n} 2^{- \sum_{i=1}^{n} \left \{ \Gamma (y_i | x, B) - \Gamma (x | B) \right \}} \cdot \vec{y}_i$$

4.1 Realizing the Metadata Calculus

So far we have identified the properties that need to be satisfied by the metadata vector space in order to build a meaningful metadata calculus for secure information sharing. In this section, we describe a concrete instantiation of our metadata calculus. We first remark that the non-negativity constraint on value $\Gamma x(t) \geq 0$, restricts the metadata vector space to a half-space $^4[6]$. We recognize that achieving homomorphism on both the $+$ and $\cdot$ operator on a half-space is infeasible without metadata expansion. Metadata expansion refers to the problem of expansion in the metadata size of an output object when compared to the metadata size of the input objects. In order to overcome this limitation we first define the notion of weak homomorphism as follows.

$$\vec{x} = a \cdot \vec{y}_1 \Leftrightarrow \Gamma \vec{x}(t) = a \cdot \Gamma \vec{y}_1(t), \forall t$$

$$\vec{x} = \vec{y}_1 + \vec{y}_2 \Leftrightarrow \Gamma \vec{x}(t) \geq \Gamma \vec{y}_1(t) + \Gamma \vec{y}_2(t), \forall t$$

It follows from Equation 6 that satisfying weak homomorphism results in overestimation of value $^5$. In this section we will present optimal constructions (optimal in the tightness of value estimates) that satisfy weak homomorphism, while using a constant size metadata of two units. The optimality of these constructions further acts as a testament that it is impossible to satisfy strong homomorphism without suffering from metadata expansion. However, these constructions do not allow us flexibly tradeoff metadata size with tightness. In the subsequent sections we propose using a class of compact parametric curves (B-splines) for approximating value functions while satisfying weak homomorphism and facilitating flexible tradeoff between metadata size and tightness.

Our sample construction uses a metadata space $M = \mathbb{Z} \times \mathbb{Z}$, where $\mathbb{Z}$ denotes the set of all integers. Hence, a metadata vector $\vec{x}$ can be represented as $\vec{x} = (c_0, c_1)$, where $c_0, c_1 \in \mathbb{Z}$. We first describe our construction assuming a linear value function $\Gamma \vec{x}(t) = \max(c_0 - c_1 \cdot t, 0)$. Given $\vec{y}_1 = (c_0, c_1)$ and $\vec{y}_2 = (c_0', c_1')$, the $+$ and $\cdot$ operators are defined as follows:

$$\Gamma \vec{x}(t) = \max(c_0 - c_1 \cdot t, 0)$$

$$\vec{y}_1 + \vec{y}_2 = \left( c_0 + c_0', \frac{c_0 + c_0'}{\max \left( \frac{c_1 + c_1'}{\frac{c_1}{c_1'}}, \frac{c_1}{c_1'} \right) } \right)$$

(7)

Figure 5 illustrates the $+$ operator on $\vec{y}_1 = (10, 2)$ and $\vec{y}_2 = (20, 10)$. It is easy to see that $\Gamma \vec{y}_1(t) + \Gamma \vec{y}_2(t)$ is:

$$\Gamma \vec{y}_1(t) + \Gamma \vec{y}_2(t) = \begin{cases} 30 - 12t & \text{if } t \leq 2 \\ 10 - 2t & \text{if } 2 < t \leq 5 \end{cases}$$

However, the above equation cannot be represented by a straight line and thus cannot be mapped into a metadata vector in $M$. Hence, we choose the least conservative straight line $(30 - 6t)$ such that $\Gamma \vec{y}_1(t) + \Gamma \vec{y}_2(t) \leq \max(30 - 6t, 0)$ for all $t$. In this case, one can show that the $+$ and $\cdot$ operators as defined above are optimal. We say that the output metadata $\vec{x}$ is optimal if there exists no $\vec{x}' \neq \vec{x}$ such that:

$$\Gamma \vec{y}_1(t) + \Gamma \vec{y}_2(t) \leq \Gamma \vec{x}'(t) \leq \Gamma \vec{x}(t), \forall t$$

Denoting $\vec{x}' = (c_0', c_1') = \vec{y}_1 + \vec{y}_2$, it is easy to see that by setting $t = 0$, $c_0' \geq c_0 + c_0'$; similarly, by setting $t = \max \left( \frac{c_1}{c_1'}, \frac{c_1'}{c_1} \right)$.

$^4$A plane splits the vector space into two halves of which only one half is a feasible region. In our case, the plane is defined by the constraint $\Gamma \vec{x}'(t) \geq 0$.

$^5$Unlike MLS our approach never underestimates the value of an object.
it is easy to see that $c_1^2 \leq \frac{c_1 + c_2}{\max\left(\frac{4}{7}, \frac{8}{7}\right)}$. Hence, it follows that $\Gamma(x)(t) \geq \Gamma(t)$ for all $t$.

Similarly, one can define operators $+$ and $\cdot$ assuming an exponentially decaying value function $\Gamma(x)(t) = c_0 + c_1 e^{-c_1 t}$. Given $\vec{y}_i = (c_1, c_1^1)$ and $\vec{y}_2 = (c_2, c_1^2)$, the $+$ and $\cdot$ operators are defined as follows (see Figure 6):

$$\Gamma(x)(t) = c_0 + c_1 e^{-c_1 t}$$

$$a \cdot \vec{y}_i = (a \cdot c_0, c_1)$$

$$\vec{y}_i + \vec{y}_2 = (c_0 + c_2, \min(c_1^1, c_1^2))$$

In this case, one can again show that the $+$ and $\cdot$ operators as defined above are optimal. Denoting $\vec{x}' = (c_0, c_1^1) = \vec{y}_i + \vec{y}_2$, it is easy to see that by setting $t = 0$, $c_0^3 \geq c_0^1 + c_2^0$. By setting $t = \infty$ and requiring that $\lim_{t\to\infty} c_0^3 e^{-c_1 t} \geq 1$, it is easy to see that $c_1^2 \leq \min(c_1^1, c_1^2)$). Hence, it follows that $\Gamma(x)(t) \geq \Gamma(t)$ for all $t$. Indeed, the fact that the value estimates obtained by these constructions are weakly homomorphic and optimal in their tightness further testifies our claim that it is impossible to satisfy strong homomorphism without incurring metadata expansion.

Now, we revisit the metadata expansion problem for strong homomorphism. We show that metadata expansion rate is in the worst case exponential in the number of information transforms for both polynomial and exponential decay functions. For simplicity, let us start by considering a linear decay function. In the example shown in Figure 5 it is easy to see that one can accurately capture the metadata of the output object $x$ using two straight lines (one from $(0, 30)$ to $(2, 6)$ and the second from $(2, 6)$ to $(5, 0)$). In general if the value of input objects $y_1$ and $y_2$ may be represented by $m_1$ and $m_2$ piece-wise linear functions respectively, then the value of $\vec{x}' = \vec{y}_1 + \vec{y}_2$ may in the worst case consist of $m_1 + m_2$ piece-wise linear functions. Hence, it follows that $|\vec{x}'| \geq 2\min(|\vec{y}_1|, |\vec{y}_2|)$, where $|\vec{x}'|$ denotes the size of metadata vector $\vec{x}'$. Extending this argument, it is easy to see that for $\vec{x} = \vec{y}_1 + \vec{y}_2 + \cdots + \vec{y}_n$, $|\vec{x}| \geq 2^n - 1 \min\left\{\sum_{i=0}^{n} (c_0^i e^{-c_1 n})\right\}$ (exponential in $n$). In general for any polynomial decay function of degree $k$, one can show that the worst case metadata expansion rate is $(k + 1)^{n - 1}$.

The same argument also applies to the exponential decay function. Using an exponential decay function, one can at best capture the sum of two metadata vectors $(20, 10)$ and $(10, 2)$ using a metadata vector of size four $(20, 10, 10, 2)$. In general one can define the value of a metadata vector $(c_0, c_1, \cdots, c_{2k-1})$ as $\sum_{i=0}^{k-1} c_{2i} e^{-2c_{2i+1} t}$. It is easy to see that for $\vec{x}' = \vec{y}_1 + \vec{y}_2$, $|\vec{x}'| \geq 2^n - 1 \min\left\{|\vec{y}_1|, |\vec{y}_2|\right\}$, thereby resulting in exponential metadata expansion rates.

Computing optimal metadata expansion rates of various decay functions in order to satisfy strong homomorphism may be an independent problem of interest. Unfortunately, exponential metadata expansion rates make it extremely unwieldy to manage metadata. In the rest of this paper, we will focus on constructing metadata spaces that do not suffer from metadata expansion (and thus only satisfy weak homomorphism). We argue that the exponential metadata expansion rate stems from the lack of a compact basis\(^6\) for value decay functions. In this paper we propose to use basis-splines (B-splines for short), a class of compact parametric curves, for tightly approximating value functions. The use of B-splines allows to compactly define the $+$ and $\cdot$ operators without incurring metadata expansion. We also show that the strong convex hull property of B-splines guarantees weak homomorphism; further, we show that one can vary a tunable parameter (number of control points in a B-spline) to flexibly tradeoff the size of the metadata vector and tightness of value estimates.

### 4.2 B-Splines and Weak Homomorphism

A spline is a special function defined piecewise by polynomials. B-spline is a spline function that has minimal support with respect to the degree of polynomial and smoothness such that every spline function of a given degree and smoothness can be expressed as a linear combination of B-splines of the same degree and smoothness. Splines have been extensively studied in the field of computer graphics and are commonly used for curve-fitting. In this paper, we will use a clamped uniform cubic B-spline [16] (henceforth, simply referred to as spline) that can be compactly represented as $\mathcal{V}$ that spans (or generates) $\mathcal{V}$.\(^6\)
a collection of parametric functions defined as follows:

\[
S_i(\tau) = \begin{bmatrix} \tau^3 & \tau^2 & \tau \end{bmatrix} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix} \begin{bmatrix} p_{i-1} \\ p_i \\ p_{i+1} \\ p_{i+2} \end{bmatrix}
\]

for \( \tau \in [0, 1] \)

where \( p_i \) denotes the \( i^{th} \) control point in the spline. Figure 8 shows a sample spline over 11 control points (labeled 0, \( \cdots \), 10). A spline attempts to follow its control points while satisfying \( C^2 \) continuity. A cubic B-spline on \( n \) control points is defined using \( n - 3 \) parametric curves, where the \( i^{th} \) \((0 \leq i \leq n - 4)\) segment refers to \( \tau \in [\frac{i}{n-3}, \frac{i+1}{n-3}] \).

**Definition 1** (\( C^n \)-Continuity). A function \( f \) is \( C^n \) continuous if its first \( n-1 \) derivatives (including the \( n^{th} \) derivative, namely, the function \( f \) itself) exist and are continuous.

**Definition 2** (Slow-Decreasing Functions). A function \( f \) is said to be slow-decreasing if \( f \) is strictly decreasing function and its first derivative \( f' \) is non-decreasing.

**Lemma 1** (Properties of Slow-Decreasing). If functions \( f_1(t) \) and \( f_2(t) \) are slow-decreasing then: (i) \( f_1(t) + f_2(t) \) is slow decreasing and (ii) for any scalar \( a > 0, a*f_1(t) \) is slow decreasing.

In this section, we will describe techniques to develop B-spline approximations to value functions that are guaranteed to satisfy weak homomorphism. First we show that assuming linear or exponential decay the value function is impossible.

A cubic B-spline is defined on a minimum of four control points; on fewer control points a cubic B-spline is just a collection at most three piecewise linear functions that connect adjacent control points. Second, we show how to construct a convex hull that dominates slow-decreasing functions. Third, we show how to construct a spline that dominates the convex hull, and thus dominates the value function guaranteeing weak homomorphism.

First, we show that assuming linear or exponential decay the value function \( f \) is slow-decreasing. We have already shown that assuming linear decay the value function is composed of piecewise linear functions (see Figure 5). We claim that the derivate of the value function is monotonically non-decreasing but discontinuous at certain time instances. For example, in Figure 5 the derivative of the value function ('strong' in Figure 5) is \(-2 \leq t < 2\) and \(-2 < t \leq 5\). For an exponential function, its second derivate is always positive implying that its first derivative is non-decreasing. From Lemma 1 it follows that all value functions derived using a slow-decreasing value operator \( \Gamma \) and operators + and \(-\) are slow-decreasing.

**Lemma 2** (Convex Hull & Slow-Decreasing [16]). For any slow decreasing function \( f(t) \) and for any \( t_1 < t_2 < \cdots < t_r \), the points \( \{(t_i, f(t_i))\} \) form a convex hull that dominates the function \( f \).

Second, it follows from Lemma 2 that for any \( t_1 < t_2 < \cdots < t_r \), the points \( \{(t_i, f(t_i))\} \) form a convex hull that dominates a slow-decreasing value function \( f \). Figure 9 shows a convex hull obtained by setting \( t \) to the points of discontinuity in the derivate of a linear decaying value function; hence, the convex hull matches the value function itself. Figure 10 shows a convex hull obtained by selecting random control points on an exponentially decaying value function.

**Lemma 3** (Strong Convex Hull Property [16]). A B-spline curve is guaranteed to be contained in the convex hull of its control poly-line.

Third, having constructively shown the existence of a convex hull that dominates a slow-decreasing value function, let us pick one such convex hull \( C \). Let \( S \) denote a spline whose control points are the extreme points of the convex hull. By the strong convex hull property of B-splines, the spline is guaranteed to be contained within the convex hull, that is, the spline dominates the convex hull (see Figures 9 and 10), which in turn dominates the slow-decreasing value function.

We now show that one can define exact + and - operators on spline functions. Let \( S(p_1, \ldots, p_k) \) denote a spline whose control points are \( \{p_1, \ldots, p_k\} \). Then, the + and - operators are defined as follows:

\[
a \cdot S(p_1, \ldots, p_k) = S(a \cdot p_1, \ldots, a \cdot p_k)
\]

**Figure 7:** Metadata Expansion Vs Tightness of Value Estimates

**Figure 8:** Cubic B-Splines

**Figure 9:** A convex hull obtained by selecting random control points on an exponentially decaying value function.

**Figure 10:** A convex hull obtained by setting \( t \) to the points of discontinuity in the derivate of a linear decaying value function; hence, the convex hull matches the value function itself.

**Figure 11:** A convex hull obtained by selecting random control points on an exponentially decaying value function.
We remark that strong homomorphism of spline functions does not translate into strong homomorphism on the metadata vector space $M$. Recall that splines dominate the value function, hence strong homomorphism on splines may at best translate into weak homomorphism on $M$.

We formally show how to establish weak homomorphism on $M$ using spline approximations to value functions. We define the metadata of an object $x$ as $\vec{x} = \{p_1, \ldots, p_k\}$ an ordered set of control points in a spline $S(p_1, \ldots, p_k)$ that dominates the value function of $x$. Given $\vec{y} = \{p_1^1, \ldots, p_k^1\}$ and $\vec{y}^2 = \{p_1^2, \ldots, p_k^2\}$ then the $+$ and $\cdot$ operators on the metadata vector space as defined as follows:

$$\vec{a} \cdot \vec{y} = \{a \cdot p_1, \ldots, a \cdot p_k\}$$

$$\vec{y} + \vec{y}^2 = \{p_1^1 + p_1^2, \ldots, p_k^1 + p_k^2\}$$

**Theorem 1.** Using the $+$ and $\cdot$ operators as defined in Equation 11 guarantees weak homomorphism on the metadata vector space $M$ whose value operator $\Gamma$ is slow-decreasing.

**Proof.** Let $\vec{y}_1 = \{p_1^1, \ldots, p_k^1\}$ and $\vec{y}_2 = \{p_1^2, \ldots, p_k^2\}$. Let $\vec{x} = \vec{y}_1 + \vec{y}^2 = \{p_1^1 + p_1^2, \ldots, p_k^1 + p_k^2\}$. We have already shown that the splines $S(\vec{p}_1, \ldots, \vec{p}_k)$ and $S(\vec{p}_1^1, \ldots, \vec{p}_k^2)$ dominate the value function $\Gamma \vec{y}_1(t)$ and $\Gamma \vec{y}^2(t)$ respectively. From Lemma 4 it follows that $S(\vec{p}_1^1 + \vec{p}_1^2, \ldots, \vec{p}_k^1 + \vec{p}_k^2) = S(\vec{p}_1^1, \ldots, \vec{p}_k^2) + S(\vec{p}_1^1, \ldots, \vec{p}_k^2)$. Hence, setting $\Gamma \vec{x}(t) = S(\vec{p}_1^1 + \vec{p}_1^2, \ldots, \vec{p}_k^1 + \vec{p}_k^2)$ guarantees that $\Gamma \vec{x}(t) \geq \Gamma \vec{y}_1(t) + \Gamma \vec{y}^2(t)$. Similarly, let $\vec{x} = a \cdot \vec{y}_1 = \{a \cdot p_1, \ldots, a \cdot p_k\}$. From Lemma 4 it follows that $S(a \cdot \vec{p}_1, \ldots, a \cdot \vec{p}_k) = a \cdot S(\vec{p}_1, \ldots, \vec{p}_k)$. Hence, setting $\Gamma \vec{x}(t) = S(a \cdot \vec{p}_1, \ldots, a \cdot \vec{p}_k)$ and $\Gamma \vec{y}_1(t) = S(\vec{p}_1, \ldots, \vec{p}_k)$ guarantees that $\Gamma \vec{x}(t) = a \cdot \Gamma \vec{y}_1(t)$.

We have so far shown that one can use a B-spline approximation value functions (assuming that the value operator $\Gamma$ is slow-decreasing) to guarantee weak homomorphism. One of the key advantages of using the B-spline approximation is that one can flexibly trade off tightness of value estimates with the number of control points (size of the metadata) in the spline. In the rest of this section, we will discuss solutions to vary the number of control points in a spline.

It is easy to see that one can increase the number of control points by choosing new control points on a linear interpolation of any two existing control points; however, note that this neither changes the spline function or the tightness of value estimates. To decrease the number of control points, we first argue that removing one or more control points does not compromise weak homomorphism in the metadata vector space. Then, we describe solutions to reduce the number of control points in a spline using the minimum curvature heuristics.

**Theorem 2.** Removing any control point from a spline $S$ does not violate weak homomorphism on the metadata vector space $M$ whose value function $\Gamma$ is slow-decreasing.

**Proof.** Let $\mathcal{C}_P$ denote the convex hull formed by an ordered set of control points $P$ that contains the spline $S_P$. On removing any control point $p \in P$, we claim that the set of remaining control points form a convex hull $\mathcal{C}_{P-P}$ that is contained within $\mathcal{C}_P$ and thus dominates $\mathcal{C}_P$. From the strong convex hull property, a spline $S_{P-P}$ on the set of control points $P - \{p\}$ dominates the convex hull $\mathcal{C}_{P-P}$, which in turn dominates $\mathcal{C}_P$, thereby guaranteeing weak homomorphism.

We remark that the spline $S_{P-P}$ may not strictly dominate the spline $S_P$; however, the value estimates obtained from spline $S_{P-P}$ will in expectation be tighter than that obtained from spline $S_{P-P}$ (see Figure 11).

**Definition 3 (Curvature).** Given a spline $S$ with control points $\{p_i\}$, the curvature of a control point $p_i$ is defined as $|\pi - \theta|$ where $\theta$ denotes the angle between a straight line connecting control points $p_{i-1}$ and $p_i$ and a straight line connecting control points $p_i$ and $p_{i+1}$.

The minimum curvature heuristics removes the control point that has the least curvature. The key idea is that if the curvature of a control point $p_i$ is close to zero, then the control points $p_{i-1}$, $p_i$ and $p_{i+1}$ are nearly collinear. Hence, removing the control point $p_i$ results in minimum deformation to the underlying convex hull. One can apply the minimal curvature heuristics multiple times in order to eliminate two or more control points from a spline without violating weak homomorphism on $M$. 

![Figure 9: Strong Convex Hull Property: Linear Decay Function](image1)

![Figure 10: Strong Convex Hull Property: Exponential Decay Function](image2)
Figure 11: Reducing the Number of Control Points: Hull2 is contained in Hull1; Spline1 is contained in Hull1; Spline2 is contained in Hull2 which in turn is contained in Hull1 (⇒ weak homomorphism); Spline2 may not strictly dominate Spline1

4.3 Tightness of Value Estimates

We have so far described sample realizations of our metadata calculus that satisfy weak homomorphism for linear and exponential value decay functions. Recall that by definition, weak homomorphism makes a conservative estimate on the output object’s value. In this section, we present solutions to capture the tightness of the value estimates obtained using our metadata calculus. We show that in the worst case, weak homomorphism without metadata expansion may result in arbitrary loss of tightness.

Let \( A\vec{x} \) denote the area under the curve \( \Gamma\vec{x}(t) \). \( A\vec{x} \) summarizes the value of an object \( x \) over its lifetime.

\[
A\vec{x} = \int_{t=0}^{\infty} \Gamma\vec{x}(t)dt
\]

Let \( \vec{x} = \vec{y}_1 + \vec{y}_2 \). We define the tightness \( \eta \) of our value estimates as follows:

\[
\eta = \frac{A\vec{x}}{A\vec{y}_1 + A\vec{y}_2} - 1
\]

We note that if \( \Gamma(t) = \Gamma\vec{y}_1(t) + \Gamma\vec{y}_2(t) \) for all \( t \), then \( \eta = 0 \). However, we show that even for the optimal + operator (on metadata of size two units) as defined in Equations 7 and 8 for the linear and exponential decay functions (respectively), the worst case tightness \( \eta \) is \( \infty \). Using a linear decay function, tightness of the optimal + operator as defined in Equation 7 is given by:

\[
\eta = \frac{\min(c^1_0, c^2_0) \cdot (c^1_1 + c^2_1)}{c^1_0 \cdot c^1_1 + c^2_0 \cdot c^2_1} - 1
\]

One can see that \( \eta = 0 \) (the value estimation is tight) if \( c^1_0 = c^2_0 \). On the other hand, in the worst case \( \eta = \infty \) when \( c^1_2, c^2_0, c^2_0 \) are arbitrarily large while \( c^1_1 \) is arbitrarily small.

Similarly, using an exponential decay function, tightness of the optimal + operator (on metadata of size two units) as defined in Equation 8 is given by:

\[
\eta = \frac{c^1_1 + c^2_1}{\min(c^1_0, c^2_0)} - 1
\]

One can see that \( \eta = 0 \) (the value estimation is tight) if \( c^1_0 = c^2_0 \). On the other hand, in the worst case \( \eta = \infty \) when \( c^1_2, c^2_0, c^2_0 \) are arbitrarily large while \( c^1_1 \) is arbitrarily small.

5. DISCUSSION AND LIMITATIONS

Background Information and Uncertainty: In this paper we have highlighted the role of background information in estimate the value of information\(^9\). However, in reality it may be impossible to obtain accurate and up to date background information. Hence, the key challenge here is to develop meaningful value estimates under uncertainty on background information either due to delayed updates or possibly incorrect (false) updates.

Disinformation: Our proposed approach does not handle disinformation, namely, information that is intended to misguide the recipient. This limitation follows from the fact that information theory does not offer constructs for disinformation. Accounting for disinformation may require an entirely new metadata calculus; say based on subjective logic [10, 7] to reason over conflicting evidences.

\(^9\)MLS does not account of background and consequently makes poor value estimates
Bootstrapping: In this paper we have not addressed the problem of initializing metadata for a newly generated object. Other authors have proposed natural language processing based techniques to automatically categorize human generated documents [4, 21].

6. CONCLUSION

In both business and military applications, getting the right information, to the right people, at the right time, remains a challenging problem. One of the key research challenges in addressing this problem is to develop mechanisms that support tight value estimation for shared information. In this paper we have argued that the inability of MLS to accommodate dynamic attributes and information transform semantics result in either gross underestimation or overestimation of value.

In this paper we have presented a novel calculus approach to securely share tactical information. We modeled tactical security metadata as a vector half-space (as against a lattice in a MLS-like approach) that supports three operators: $\Gamma$, $+$ and $\cdot$. We have shown techniques to incorporate dynamic metadata attributes (e.g., a time sensitive value function $\Gamma$) and information transform semantics using the $+$ and $\cdot$ operators. We have shown that it is unfortunately impossible to achieve strong homomorphism without incurring exponential metadata expansion. We then used B-splines to develop concrete realizations of our metadata calculus that satisfy weak homomorphism without suffering from metadata expansion. Our initial investigation has shown that by increasing metadata size by a factor of 10 we could improve tightness by a factor of 200 and 57 for linear and exponential decay functions respectively.

Acknowledgements

Research was sponsored by the U.S. Army Research Laboratory and the U.K. Ministry of Defence and was accomplished under Agreement Number W911NF-06-3-0001. The views and conclusions contained in this document are those of the author(s) and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Research Laboratory, the U.S. Government, the U.K. Ministry of Defence or the U.K. Government. The U.S. and U.K. Governments are authorised to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

7. REFERENCES


