Determining an Economic Value of High Assurance for Commodity Software Security

Virgil Gligor Adrian Perrig David Basin*

Abstract. Security measures that attempt to prevent breaches of *commodity* software have not used high assurance methods and tools. Instead, rational defenders have risked incurring losses caused by breaches because the cost of recovery from a breach multiplied by the probability of that breach was lower than the cost of prevention by high assurance, e.g., by formal methods. This practice may change soon since breach-recovery costs have increased substantially while formal methods costs have decreased dramatically over the past decade.
We introduce the notion of *selective high assurance* and show that it is economically justified, as producers can easily recoup its cost even in very small commodity markets, and necessary for rational defenders to decrease their breach recovery costs below a chosen limit. However, these decreases depend on defenders' risk aversion, which is difficult to assess since risk preferences cannot be anticipated. A challenge is to determine a *lower bound* on the economic value of selective high assurance *independent* of the defenders' risk preferences; i.e., a value that depends only on the commodity software itself and the attacks it withstands. We propose an approach to determine such a value and illustrate it for SCION, a networking software system with provable security properties.

1 Introduction

Early observations regarding the use of high assurance methods in commodity software security suggest that little, if any, such software has benefited from the use of high assurance. High assurance includes formal methods for the specification and proof of security properties. This has been generally understood to be required to meet or exceed the evaluation assurance level EAL 7 of the Common Criteria [1]. After four decades of research, a variety of formal methods and tools have been used experimentally, but very little *commodity* software has included provable security properties. In fact, only a few *experimental* software systems whose code sizes are less than 50K SLoC have benefited from formal proofs of security at the source code level, e.g., microkernels, micro-hypervisors, I/O kernels, cryptographic libraries and protocols, and a few applications [4,5,6,7]. This raises the question of whether high assurance has *any* economic value for commodity software security.

Why Not High Assurance? Over the past two decades, three reasons have been given for not using high-assurance methods for commodity software.

* Authors' addresses: Virgil Gligor, ECE Dept. and CyLab, Carnegie Mellon University, Adrian Perrig and David Basin, Computer Science Dept., ETH Zurich.

The first is that their opportunity cost is very high. That is, rapid innovation in the commodity software market (undoubtedly fueled by the near-zero cost of entry, no liability, and hardly any regulation) eschews the use of costlier high-assurance methods in favor of developing functions to meet market demand [8].

The second reason is that many security properties that need to be proven for large, complex commodity software are either unknown or difficult to prove and hence the widespread use of high assurance becomes impractical. For example, increasing software productivity includes adding new functions to and reusing software components without causing backward incompatibility by removing or modifying existing code. This suggests that in commodity software "only giants survive" [9]. Software growth in size and complexity inevitably reaches the point where no one can identify all key security properties of the final product. It becomes difficult to reason why even simple properties hold. Hence, high assurance cannot be achieved, and even simple properties become expensive to prove.

The third reason is that, by analogy to cryptography, the pervasive use of high-assurance methods that prevent security breaches is equivalent to reaching near perfection, which is always impractical in commodity software. Instead, a *rational defender* weighs the cost of preventing breaches by high assurance against the cost of loss caused by breaches, whereby this loss, in expectation, equals the cost of recovery from a breach multiplied by the probability of the breach [10]. Until now the cost balance has always tilted away from high assurance.

Overview. We introduce the notion of *selective high assurance* and show that producers can easily recoup its cost even in very small commodity software markets. We then show that selective high assurance is necessary for rational defenders to decrease the expected recovery costs from security breaches significantly.

If a rational defender would always reject high-assurance methods for breach prevention in favor of low assurance and attack deterrence [10,11], then s/he would attempt to minimize the losses caused by undeterred attacks and unavoidable breaches; i.e., s/he would try to decrease both the cost of recovery from a breach and the breach probability. In practice, a rational defender must assume that the breach probability is 1 – as suggested by the NSA [20] and reinforced by recent industry evidence [19] – and focus on minimizing the breach-recovery cost. We argue that minimization of this cost liability requires cybersecurity insurance. However, we show that, at the expected CAGR[1] of 18.2% [21], the insurance market will fail to cover many defenders' cost liability for the foreseeable future and argue that insurance gaps will exist forever. Thus, a rational defender's only recourse is to decrease the breach probability from 1 to a sufficiently low value such that the expected loss would not exceed the cost of insurance, had insurance been possible. Then we argue that neither low assurance nor attack deterrence can lower this probability to desired levels in practice, and hence rational defenders need *selective high assurance* to do so.

Finally, we argue that the extent to which selective high assurance is necessary depends on the defenders' risk aversion. To avoid this dependency, we

[1] CAGR stands for compound annual growth rate.

describe a way to find a lower bound on the value of selective high assurance that depends only on the selected software components and the attacks they withstand.

The numerical figures cited below, taken from industry reports, refer to surveys and other empirical methods used to infer defenders' beliefs and preferences.

2 Why Revisit High Assurance?

None of the reasons for avoiding it implies that high assurance is *always* unjustifiable for commodity software security. In fact, there are at least two basic reasons for revisiting the case for it. First, the cost of high assurance has decreased dramatically during the past decade, while simultaneously the cost of recovery from security breaches has increased substantially. The balance between the cost of high assurance, which demonstrably prevents security breaches, and the cost of recovery from breaches when high assurance is not used, is thus beginning to tilt towards the use of high assurance.

Cost trends. Recent industry evidence shows that the cost of security breaches is nearly 1% of the global GDP, representing a three-fold increase over a decade ago[2]. The average cost of recovery from a single breach is about $4.24M globally [14]. Although this figure drops to $3.28M for systems employing mature zero-trust architectures, it is generally not lower than $2.9M for systems that use the most advanced AI and automation tools for early breach detection and recovery. At the same time, the cost of using formal specification and verification methods and tools has decreased dramatically. A decade ago, this cost was about $362/SLoC[3] for well-known micro-kernel development, i.e., seL4 [4]. Use of comparable formal methods has incurred a 33% lower per-SLoC cost (e.g., about $225/SLoC), even after increasing labor costs to $300K/person-year; see an approximately 50% smaller and less known micro-kernel for I/O separation [7]. An even lower cost of about $128/SLoC was incurred for the Ironclad project [5]. EverCrypt [6] achieves the lowest cost of any major system to date at under $40/SLoC – one ninth of the SLoC cost of seL4. Although one could question cost comparability – given the different systems' complexity, how they are designed, and the designers' skill variability – the trend is unmistakable: the cost of code-level formal verification is decreasing dramatically.

Selective high assurance. The second reason to revisit high assurance is that it can be used *selectively* to prevent breaches of relatively small, security-critical isolated software components (i.e., software "wimps") rather than for all other components of a large commodity software system, i.e., a software "giant" [16]. High assurance is practical for selected commodity software components of non-trivial size, say 70K SLoC, early in the development cycle. How many security breaches could be demonstrably prevented by the selected components? The answer depends on their size and complexity, desired security

[2] Recent cost estimates range between 0.8% [2] to slightly over 1% [13] of global GDP.
[3] SLoC stands for Source Lines of Code.

properties and the attacks they counter, as well as whether the producer can recoup its cost on the market.

An Example. Let the publicly unknown value t be the maximum number of exploitable attack targets for a commodity software system, e.g., the number of unremediated CVEs of a "giant" can be very large [24]. Let b, $b \ll t$, be the number of attacks that can be countered by formally verifying selected code of the "giant." A software producer needs to select, isolate, and formally verify at most b "wimps" to deny these attacks at a *one-time cost* C_b(verification). Let C_b(recovery) be a defender's minimum *recurrent annual cost* of recovery from b breaches of the "giant" when all its code is unverified. Since the "giant" is a *commodity* system, its market is comprised by m enterprises which use the software n years, where m is of the order of tens of thousands worldwide and n is of the order of ten years. Hence, the recovery from b breaches would have a *market cost* of the order of $\mathrm{C}_b(\text{recovery})\cdot mn$. Below we show that a producer can recoup the cost of selective high assurance even in very small markets.

Recall that when the probability of a breach is 1 [20,19], a rational defender is wiling to pay for breach prevention (e.g., by formal methods) if $\mathrm{C}_b(\text{verification}) \leq \mathrm{C}_b(\text{recovery})\cdot 1$ [10]. A defender can always determine C_b(verification) by relying on independent estimates by security companies, evaluation laboratories, and technical literature. A producer's selection of b "wimps" and their code sizes can also satisfy this condition for *most* defenders since it can easily obtain the lowest (average) breach cost from annual surveys of breach costs[4]. For example, a typical US enterprise sustains 3 distinct breaches in 42 attacks per year [19] at a cost of $\mathrm{C}_3(\text{recovery}) = \$8.7\text{M}$ (3×\$2.9M),[5] using the lowest (average) breach cost of \$2.9M [14]. Isolating and formally verifying 3 "wimps" of at most 72.5K SLoC each would cost $\mathrm{C}_3(\text{verification}) \leq \$8.7\text{M}$ (3×72.5K SLoC×\$40/SLoC).

Assume that all b attacks would target, and hence be countered by, the b formally verified penetration resistant "wimps." In this extreme scenario, most rational defenders would find $\mathrm{C}_b(\text{verification}) \leq \mathrm{C}_b(\text{recovery})$ and spend C_b(verification) upfront because there is no cost after the first year, regardless of what the other $m-1$ defenders do. Not spending C_b(verification) upfront is far riskier, since remediating vulnerabilities by low assurance and limited deterrence after breaches (see Section 3.2) cannot rule out future breaches and recovery-cost recurrence.

Now consider the other extreme scenario whereby an adversary targets and causes b breaches per year by attacking the decreased target space of $t-b$ vulnerabilities of the "giant." That is, these vulnerabilities were *not* removed by the formal verification of the b "wimps." Since the producer has lowered the breach probability by $\epsilon \ll 1$ at the cost C_b(verification), the market cost for recovery becomes $(1\text{-}\epsilon)\cdot\mathrm{C}_b(\text{recovery})\cdot mn$. The producer can recoup C_b(verification) when

$$(1\text{-}\epsilon)\cdot\mathrm{C}_b(\text{recovery})\cdot mn + \mathrm{C}_b(\text{verification}) \leq \mathrm{C}_b(\text{recovery})\cdot mn$$

is satisfied, or equivalently when $\epsilon > [\mathrm{C}_b(\text{verification})/\mathrm{C}_b(\text{recovery})]/(mn)$. Since $\mathrm{C}_b(\text{verification}) \leq \mathrm{C}_b(\text{recovery})$, the smallest ϵ that satisfies the above con-

[4] It is possible to select $\mathrm{C}_b(\text{verification}) > \mathrm{C}_b(\text{recovery})$, e.g., using an average per-breach cost, and still satisfy the required condition for *some* defenders.

[5] A decade ago, the *average* recovery cost of a US company was already \$8.9M [15].

dition is $\epsilon > 1/(mn)$.[6] For any $t \gg b$, setting $\epsilon = b/(t-b)$ and using $\epsilon > 1/(mn)$ shows that a producer can recoup C_b(verification) whenever $t < b(mn+1)$. As shown below, this holds for values of t derived from industry surveys yielding *very small* lower bounds $m_0 < m$ and $n_0 < n$ for commodity market sizes. A producer could always recoup its cost by a commodity price increase of C_b(verification)$/m$ $<$ C_b(verification)$/m_0$, where $m > m_0$ is the anticipated number of defenders.

A producer's cost of using selective high assurance is easily recouped for all cases between the above two extreme scenarios.

Estimating (t, m_0, n_0). How can we estimate t for a specific commodity software system with b formally verified "wimps" as well as m_0 and n_0 from $t < b(m_0n_0+1)$? Although all relevant CVEs for many commodity software systems are published (e.g., https://www.cvedetails.com/product-list.php), industry surveys cannot report t for any specific system, since that would reveal how many vulnerabilities are left unremediated and encourage attacks. Instead, surveys report only the total number of responders, R, and the total number of unremediated vulnerabilities, V, covering an unknown number of commodity software systems for a possibly unknown number of unknown organizations. Also unreported in any survey is the average number of vulnerable software systems in each organization, s, and the average number of individuals of each organization reporting unremediated vulnerabilities independently, r, though typically $r = 1$. Then, $V = t \times s \times R/r$, and therefore $t = \lceil rV/sR \rceil$. Since most organizations have many more commodity software systems than survey responders $r/s \leq 1$, and hence $t \leq \lceil V/R \rceil$. From this and $t < b(m_0n_0+1)$ we can derive m_0 for a given $n_0 > 1$ years of software use.

For example, in a recent survey [24], 47% of 634 respondents reported that their organizations had vulnerabilities in their software systems that were *not* remediated over the past 12 months. On average, 1.1M individual vulnerabilities were in this backlog and an average of only 46% were remediated. Hence, R $=$ 47%×634, $V =$ (1-46%)×1.1M, and t $\leq \lceil (1-46\%) \times 1.1M/(47\% \times 634) \rceil =$ 1994. When $t = 1994$, the values of $m > m_0$ and $n > n_0$ that allow a producer to recoup C_b(verification) are *very* small. That is, for $b = 3$, $m_0 = 332$, or less than 5.3% of the more than 6,300 companies registered on the US stock exchanges, and n_0 as small as $n_0 = 2$ satisfy the relation $1994 < 3(m_0n_0{+}1)$. For more typical values of n_0, such as $n_0 = 7$, m_0 decreases substantially, i.e., $m_0 = 95$.

The same survey [24,25] shows that 66% of 634 responders reported that their organizations have backlogs of over 100,000 vulnerabilities, and 54% of these responders reported that less than 50% of the vulnerabilities in their backlogs were remediated. That is, each of 192 ((1-54%)×66%×634) organizations has *at least* 50,000 unremediated vulnerabilities. Since s is unreported for any of these organizations, we assume that each has $s = 200$ commodity software applications – a figure reported in another survey [26] of 30,000 applications in 190 companies. Thus, $t \geq 250$ (50,000/200) and for $b = 3$, the relation $250 < 3(m_0n_0{+}1)$ yields values of m_0 and n_0 that are much smaller than above, i.e., $m_0 = 42$ for $n_0 = 2$, and $m_0 = 12$ for $n_0 = 7$.

[6] If C_b(verification)$<$$\mathrm{C}_b$(recovery), $\epsilon > 1/(mn) >$ C_b(verification)/C_b(recovery)·(mn).

3 The Need for Selective High Assurance

In the previous section, we showed that a producer's selective use of formal methods is economically justified; i.e., the cost of selective high assurance can be easily recouped. In this section, we show that selective high assurance is necessary for rational defenders to decrease the probability of software breaches and thus reduce the estimated cost of recovery below a chosen limit.

The *rational defenders* we consider are enterprises with the following common characteristics: they use only low assurance for breach prevention and attack deterrence; they can afford to use advanced AI methods and automated tools for early breach detection and low recovery cost; and they can afford all available cybersecurity insurance needed to reduce financial losses to a minimum. A rational defender computes the expected recovery cost as

$$\textit{cost(defender)} = \textit{recovery\_cost(breach)} \times \textit{probability(breach)}$$

and attempts to minimize either *recovery_ cost(breach)* or *probability(breach)*, or both. Then rational defenders balance *cost(defender)* against the breach-prevention cost before deciding whether prevention is cost effective [10].

3.1 Minimizing breach-recovery cost

Recent industry evidence [14] shows the lowest (average) *recovery_ cost(breach)* = \$2.9M is currently achieved when an enterprise uses advanced AI methods and automated tools that integrate the results of its many security administrative tools, e.g., on the average from 60 to more than 75 security tools [26]. However, to decrease the *recovery_ cost(breach)* further and minimize *cost(defender)*, any rational defender would certainly purchase cybersecurity insurance whenever possible. Why? Insurance providers spread recovery liability over many defenders with market-clearing premiums, thus decreasing a defender's cost to a *minimum*.

In 2021, the highest cyber-insurance premiums for IT-intensive industries (e.g., financial services, healthcare, payment processing, pharmaceuticals, gaming, and e-commerce) were under \$2,500 per \$1M liability with a small deductible, i.e., \$1K [22]. Assume that an insurable policy allows an optimistic scaling factor[7] of 2.9 to cover the cost of a typical US enterprise experiencing an average of three breaches per year. A company acting as a rational defender would require a recurrent yearly premium of about \$21,750 ($3\times$\$2.5K$\times$2.9). Although these premiums increase for companies with higher breach-recovery costs, even a ten-fold increase would be clearly affordable for the over 6,300 companies listed on the US stock exchanges and the top 25% of the more than 50,000 companies listed on all other stock exchanges worldwide.

Insurance gaps. In 2021, the cybersecurity insurance premium market was about \$9.5B worldwide [21], of which roughly \$3.2B was in the US [23]. Only about 1,103 (\$3.2B/\$2.9M) breaches could be covered in the US and about 2,173

[7] This scaling accounts for the lowest *recovery_ cost(breach)* = \$2.9M, which assumes that advanced AI methods and tools detect and recover from breaches. This is lower than the recovery cost per breach of \$3.28M in mature zero-trust architectures [14].

($6.3B/$2.9M) in the rest of the world. Assuming the yearly average of three breaches per US enterprise holds worldwide, a large gap appears between companies that could be insured and those which could not; namely, 5,932 US companies (6,300-1,103/3) and 11,775 in the top 25% worldwide ex-US (25%×50,000 – 2,173/3) could not be insured. Note that insurance gaps persist even if we assume that each company sustains a single breach per year.

Remarkably, insurance gaps are expected to persist for the next decade, given that the cybersecurity insurance market will reach an estimated $61.2B during the next ten years at an expected CAGR of 18.2% [21], and assuming the 2021 ratio between the US market and the rest of the world remains about 1:2 ($3.2B/$6.3B). That is, the US market would reach $20.4B in ten years, which will cover fewer than 2,344 companies ($20.4B/$2.9M×3), accounting for only 37% of the 6,300 companies currently listed on the US stock exchanges. Similarly, the insurance available worldwide ex-US in ten years could only cover 4,689 companies ($40.8/$2.9M×3) accounting for less than 40% of 11,775 companies. Insurance gaps will persist worldwide ex-US for about nine years if we assume that each company sustains a single breach per year. Note that gap estimates exclude many other qualified companies, including large governments (i.e., city, state, federal) and non-profit organizations (e.g., hospital chains), which are not accounted for in these estimates.

Why do insurance gaps persist? Note that rational defenders create significant insurance demand to minimize breach-recovery costs. Why would insurance providers not fully meet this demand and thereby eliminate insurance gaps in the future? Insurance providers deny coverage for recurrent breaches that would otherwise cause substantial provider losses. Uninsurable breaches include those caused by failure to patch known vulnerabilities (e.g., published CVEs), "insider" attacks, and "acts of war." For instance, many enterprises have very large backlogs of unpatched known vulnerabilities [24], which prevent cybersecurity insurance. Breach damage caused by outside attackers who penetrate insider accounts (e.g., by malware exfiltration of credentials) and undetectably masquerade as insiders cannot be insured. Although definitions of "acts of war" can be disputed in foreign-state-sponsored attacks[8], most cyber attacks against a country's infrastructure perpetrated by a foreign country are generally attributable and indisputable. This means that companies that control critical cyber infrastructures (e.g., energy generation and distribution) *cannot* recover damages using insurance when foreign-state-sponsored attackers commit "acts of war" by breaching these infrastructures.

Note that not all losses caused by security breaches can be covered even when breach remediation costs are insurable. For example, losses caused by intellectual property theft and third-party liabilities cannot be bounded and hence cannot be covered by cybersecurity insurance.

[8] The 2017 *NotPetya* malware attack, which was attributed to Russia's military intelligence agency in the conflict with Ukraine, was found *not* to be an "act of war" when deployed against the Merck pharmaceutical company, causing a $1.4B liability for Merck's insurers [27].

The uncomfortable insurance gap may persist longer than anticipated, as cybersecurity insurance costs are increasing, and liability of frequent state-sponsored attacks may be harder to cover, if at all, in the near future [28].

3.2 Minimizing breach probability

Persistent insurance gaps show that minimum recovery costs cannot be reached by most rational defenders. Many large companies listed on worldwide stock exchanges, which certainly could afford cybersecurity insurance, appear to be unable get it for the foreseeable future. As argued above, current low-assurance methods for commodity software are unable to rule out, for instance, recurrent insider-masquerading attacks, penetrations enabled by unpatched software vulnerabilities, and acts of war.

What alternatives for lowering recovery costs would a rational defender have, since incurring a recurrent annual loss of $C_3(\text{recovery}) = \$8.7M$ for three breaches becomes unacceptable in future years? Given the *cost(defender)* equation above, the only other way to minimize these losses is to decrease their probability; e.g., to make breaches rare events, even if they are recurrent and uninsurable. How much should this probability be decreased? A reasonable limit would be to reach the cost of a hypothetically insured US company, whose cost would not exceed the insurance premium. Using the *cost(defender)* equation, the condition $\mathit{cost(defender)} \leq \mathit{insurance\_cost}$ yields a probability upper limit

$$\mathit{upper\_limit} = \mathit{insurance\_cost}/\mathit{recovery\_cost} \geq \mathit{probability(breach)}.$$

For a US enterprise, the probability of three breaches per year would decrease from 1 to an upper limit of about 0.0025 ($\$21{,}750/3\times\$2.9M$). Would a rational defender find that such a decrease is possible by low-assurance and deterrence methods? The answer is negative, even if this *upper limit* increases ten fold.

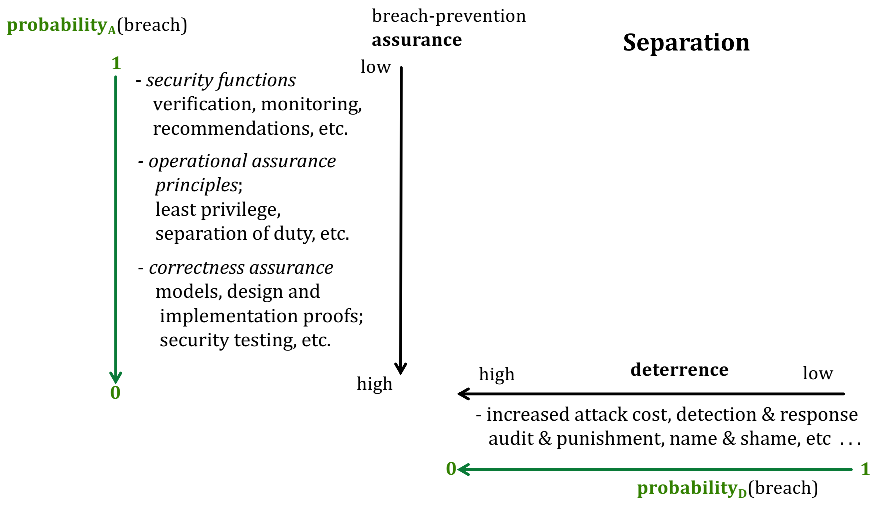
probability_A(breach)
1
- security functions verification, monitoring, recommendations, etc.
- operational assurance principles; least privilege, separation of duty, etc.
- correctness assurance models, design and implementation proofs; security testing, etc.
0
breach-prevention assurance
low
high
Separation
high
deterrence
low
- increased attack cost, detection & response audit & punishment, name & shame, etc ...
0
1
probability_D(breach)

Fig. 1. Separation of Assurance and Deterrence

Insufficiency of deterrence and low assurance. As shown in Figure 1, assurance and deterrence are fundamentally different[9] methods for defending against security breaches [11,18]. The former aims to prevent attacks by implementing security functions, following security principles, and gaining confidence by using models, specifications and correctness proofs, and testing. The latter includes techniques for increased attack cost, detection and response, audit and punishment, and naming and shaming attackers. This difference is reflected in different probability spaces corresponding to increased assurance and increased deterrence, which map differently to the [0,1] range as illustrated in Figure 2. Both mappings are monotonic, but not necessarily strictly so: intuitively, higher assurance and higher deterrence lead – in different ways – to lower breach probabilities. The desired *upper limit* shown above can bound these probabilities.

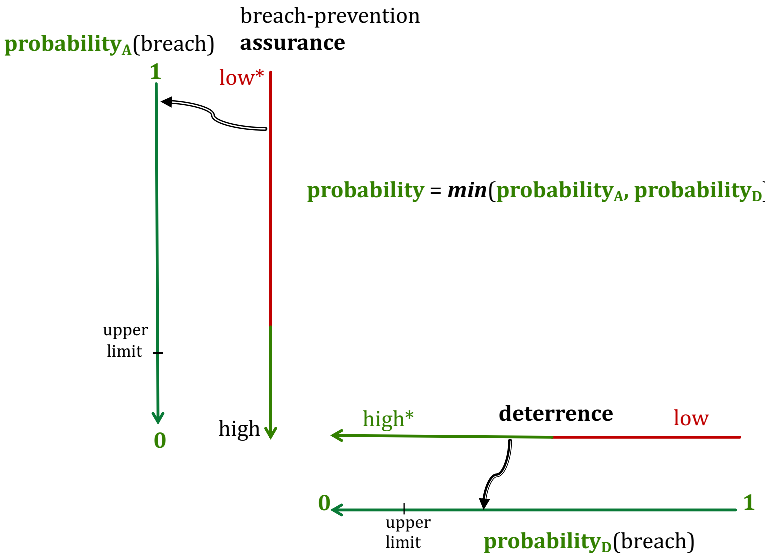

Fig. 2. Different Probabilities for Assurance and Deterrence

Deterrence requires punishment and punishment requires attack attribution anywhere on the internet [29]. Attribution on the internet is expensive, few national security agencies can afford it, and hence it is not scalable even when they can. Furthermore, many attacks originate – sometimes under false flag – from countries that do not extradite attackers, thereby rendering attribution irrelevant. Without attribution, attacker punishment becomes impossible, which rules out deterrence. This implies that, in practice, the probability of a breach cannot be always be lowered sufficiently by deterrence such that it would not exceed the desired *upper limit*, i.e., 0.0025 in our example.

[9] This difference reflects the *behavioral-economics* [17] separation between increased beliefs of trustworthiness (e.g., obtained by assurance of security properties) and decreased betrayal aversion (e..g, obtained by attack-deterrence measures) [18].

Low-assurance methods that intend to decrease breach probability are often limited to informal penetration analyses, which typically develop breach hypotheses and confirm or deny them by testing software from the OS kernel up to the web application interfaces. However, these analyses have never offered more than little or no assurance of penetration resistance [31]. Hence, they cannot guarantee decreases of breach probability such that it would not exceed a desired *upper limit* of, say, 0.0025.

Unfortunately, although deterrence and assurance are separable, the mappings to their respective probability spaces are *not* independent. In the absence of inexpensive recovery measures (as in the case here), deterrence methods are used to compensate for low assurance, and high assurance deters certain attacks; see the use of cryptography. Furthermore, deterrence implementation requires assurance and hence low assurance cannot always support effective deterrence. These dependencies imply that the probabilities induced by low assurance and deterrence cannot be multiplied to obtain a low breach probability. This means that to enforce the *upper limit* on the breach probability would require that the *minimum* of the deterrence *and* low-assurance breach probabilities must not exceed this limit. Typically this is highly improbable, and hence irrational to expect, even if the *upper limit* increases ten fold; e.g., if current insurance costs increase by a factor of ten and recovery costs remain constant. For example, to fall below the upper limit of 0.0025 shown above, the current empirically determined breach probability of 0.0714 (3 breaches/42 attacks [19]) would have to decrease by a factor of over 1:28 (0.0714/0.0025) by either deterrence or low-assurance measures, or both, which is highly improbable. A ten-fold increase in the upper limit would have a probability decrease factor of 1:2.85 (0.0714/0.025), which would still exceed the capabilities of such measures.

Market demand for selective high assurance. The only alternative left to rational defenders aiming to decrease breach probability to some desired upper limit to lower the expected recovery cost is to demand strong evidence (i.e., proofs) of formal penetration resistance for isolated critical components of commodity software products from their producers; see Figure 3. Note that, in principle, formal proofs of penetration resistance reduce – and often eliminate – the need for deterrence, since an adversary's attacks are unlikely to succeed. This implies that the probability of a breach is lower than that provided by deterrence; i.e., $\text{probability}_D > \textit{upper limit} \geq \text{probability}_A$ in Figure 3.

Recall that a commodity software system can have a market of at least m companies worldwide, where m is of the order of tens of thousands, and a lifetime of n years, where n is of the order of ten years. Thus, the demand to decrease breach probability for critical components of a single product by a non-negligible fraction of these companies could be significant, since the *market's recovery-cost savings* could reach billions of dollars thereby exceeding the one-time cost of formal proofs by orders of magnitude.

Formal proofs of penetration resistance are practical now, and hence market demand for high assurance can be satisfied by the software industry in practice. Static source code analysis for proving penetration-resistance properties was

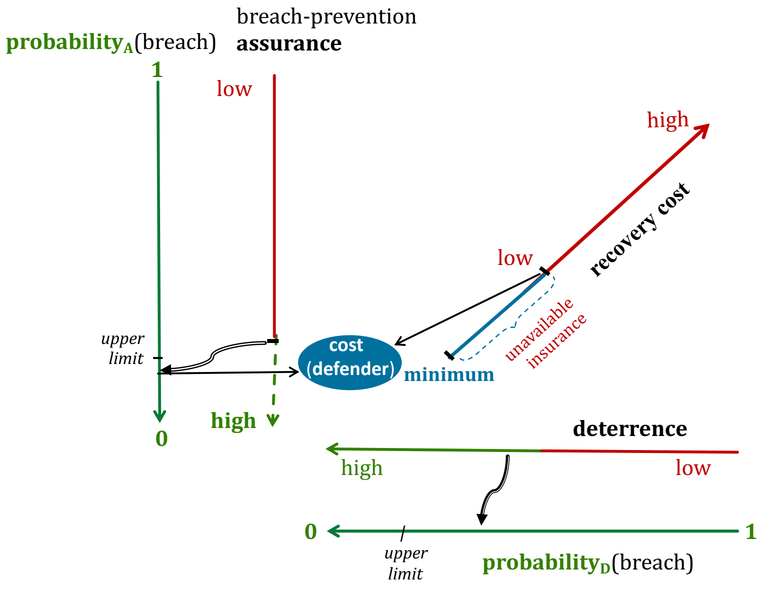

Fig. 3. Decreasing Breach Probability and Defender Cost by High Assurance

introduced over three decades ago[10] and undoubtedly stronger high-assurance methods and tools have appeared since then, e.g., for binary code and analysis of additional properties. For example, there have been substantial advances in formal property specifications, scaling model checkers, and increasing theorem-prover performance, e.g., by combining them with SMT solvers. The economics of security have been shifting for the past half a dozen years to the point that selective high assurance has become practical for commodity software [32,33].

4 A Challenge

The extent to which selective high assurance can guarantee penetration resistance of critical, isolated software ultimately depends on the defenders' risk aversion. Intuitively, to decrease breach probability to a desired upper limit, some defenders would undoubtedly require more extensive use of high assurance methods and tools than others, according to their risk aversion. However, our challenge is to determine an economic value of high assurance for penetration-resistance properties of critical software components that is *independent* of defenders' risk aversion; i.e., a value that depends only on the commodity software components and attacks they withstand. For example, this value would depend only on the software component's size, complexity, exposed interfaces, and attack surfaces. This is important because commodity software development cannot usually anticipate the context of its use and defenders' preferences. Clearly, if we determine

[10] The earliest high-assurance method and automated tool for analyzing penetration-resistance properties were used on C language programs of the Trusted Xenix (https://en.wikipedia.org/wiki/Xenix) kernel and system processes [30,31].

such a value of *selective high assurance* for penetration resistance, then we obtain a lower-bound economic value for all high assurance methods for the security of that software component.

5 Illustrating a High-Assurance Value

How can one express the value of *selective high assurance* in business-understood terms? Our approach is to illustrate a lower-bound on its economic value by applying it to a relatively large software system that has formal proofs for its security properties, for example to the SCION protocols and services [34]. This will yield realistic results since SCION has well-defined security properties, substantial size and complexity, internet-facing interfaces, and known attack surfaces. Hence, the lower-bound value of formal methods applied to its components can be convincing to potentially skeptical business audiences.

The approach we take is to select from the more than two hundred thousands of security vulnerabilities reported in the CVE[11], and CWE (e.g., MITRE's, NIST's) databases those that refer to networking software employed by the ordinary Internet. The selection process requires the use of automated tools that were used in past projects that analyzed CVE data. Then we show which attacks exploiting those vulnerabilities are countered by SCION protocols and services after employing formal design and source-code verification. Finally, we determine an industry-sanctioned average cost of recovering from breaches – or ranges thereof – caused by those attacks in the ordinary Internet, and thus obtain a *lower-bound* value of the formal methods that enables SCION to counter those breaches.

Our approach suggests the following three tasks:

1. Automate vulnerability-directed scanning of the CVE and CWE databases. The selected vulnerabilities must refer to communication services employed by the ordinary Internet, e.g., BGP, control plane, data plane, DNS, NTP, AS, ISP, vulnerabilities. A selection policy and mechanism are defined that enable directed scanning of over two hundred thousand entries of the CVE and CWE databases and to pick Internet-relevant ones for analysis.

2. Select SCION-countered vulnerability exploits. The Internet relevant CVE/CWE reported vulnerabilities are examined to determine which exploits are countered by the formal-method-based SCION design and implementation.

3. Determine the average cost of Internet recovery from the selected breaches. The average cost of recovering from breaches caused by the SCION-countered exploits selected above in the ordinary Internet can now be determined using industry-illustrated average costs, e.g., [14]. Alternatively, lower-bound recovery costs can be used for a stronger argument. In turn, this yields a value of the *opportunity loss* caused by not using the formal-method-based SCION design and source code. This is a demonstrable formal-methods value expressed in business-understood terms.

[11] The US vulnerability database (see https://nvd.nist.gov/general/nvd-dashboard and https://cve.mitre.org/cve/identifiers/) currently contains over 200000 CVEs.

6 Conclusion

We showed that it is possible to determine a useful economic lower bound for *selective high assurance* and outlined an approach to calculate this. The remarkable resemblance of Figures 1 and 3 and with those illustrating *trust establishment* [35] should be unsurprising. The notion of selective high assurance is an example of how trust establishment can yield a flexible cost allocation among security functions and assurances, residual risk reduction (e.g., when insurance is available), and some adversary deterrence.

Recent industry reports indicate a shift from "presumed breach" and recovery [35] to a "prevention-first" mindset [36]. We have argued that a shift to selective high assurance is a necessary – but not the only – step in that direction. Industry evidence shows that separate cloud services are now supporting selective formal methods/automated reasoning for increasingly many applications [33]. In time, selective high assurance could expand to more and more software components and critical systems. In the limit, with the help of further automation, selective high assurance could become a common and unremarkable discipline for all commodity software development.

Acknowledgment

Mads Dam, Kevin Foltz, Rick Kuhn, Bryan Parno, and Frank Stajano provided helpful comments on earlier drafts of this paper. We gratefully acknowledge support for this project from the Werner Siemens Stiftung (WSS) Centre for Cyber Trust at ETH Zurich and CyLab at Carnegie Mellon University.

References

1. Common Criteria. *Evaluation Assurance Levels (EALs).* *https://en.wikipedia.org/wiki/Evaluation_Assurance_Level.*

2. Finances Online. 119 Impressive Cybersecurity Statistics: 2021/2022 Data & Market Analysis, Cybermarket Statistics. *https://financesonline.com/cybersecurity-statistics/.*

3. Zhanna Malekos Smith and Eugenia Lostri and James A. Lewis. The Hidden Costs of Cybercrime. McAfee Report for Center for Strategic and International Studies, Dec. 2020. *https://www.mcafee.com/enterprise/en-us/assets/reports/rp-hidden-costs-of-cybercrime.pdf.*

4. Gerwin Klein, June Andronik, Kevin Elphinstone, Toby Murray, Thomas Sewell, Rafal Kolanski, Gernot Heiser. Comprehensive formal verification of an OS microkernel. In *ACM Transactions on Computer Systems, vol. 32*, no. 1, pp. 1-70, Feb. 2014.

5. Chris Hawblitzel, Jon Howell, Jacob R. Lorch, Arjun Narayan, Bryan Parno, Danfeng Zhang, Brian Zill. Ironclad Apps: End-to-End Security via Automated Full-System Verification. In Proc. of USENIX OSDI, pp. 165 – 181, Oct. 2014.

6. Jonathan Protzenko, *et al.* *EverCrypt*: A Fast, Verified, Cross-Platform Cryptographic Provider. In Proc. of the IEEE Symposium on Security and Privacy, (May 2020).

7. Miao Yu, Virgil Gligor and Limin Jia. An I/O separation model for formal verification of kernel implementations. In Proc. of the IEEE Symposium on Security and Privacy (May 2021).

8. Virgil D. Gligor. On the security limitations of virtualization and how to overcome them (also see transcript of discussion). In Proc. of the 2010 Security Protocols Workshop, Cambridge, UK, no. LNCS 7061, Springer, 2014.

9. Butler W. Lampson. Software components: only the giants survive. In *Computer Systems: Theory, Technology, and Applications*, Chapter 20, K. Spark-Jones and A. Herbert (eds), Springer Verlag (9):137–146, 2004.

10. Butler W. Lampson. Computer Security in the Real World. In Proc. of 16th Annual Computer Security Applications Conference. Proc. of AC-SAC, (Dec. 2000) (also in IEEE Computer, vol. 37, pp. 37-46, June 2004). *https://www.acsac.org/2000/papers/lampson.pdf.*

11. Butler W. Lampson. Usable security: How to get it. *Comm. of ACM, vol. 52*, no. 11, pp. 25-27, (Nov. 2009).

12. Finances Online. 119 Impressive Cybersecurity Statistics: 2021/2022 Data & Market Analysis, Cybermarket Statistics.
https://financesonline.com/cybersecurity-statistics/.

13. Zhanna Malekos Smith and Eugenia Lostri and James A. Lewis. The Hidden Costs of Cybercrime. McAfee Report for Center for Strategic and International Studies, Dec. 2020.
https://www.mcafee.com/enterprise/en-us/assets/reports/rp-hidden-costs-of-cybercrime.pdf.

14. IBM Corporation and Ponemon Institute. *Cost of a data breach report 2021-2022. https://www.ibm.com/security/data-breach.*

15. HP Enterprise Security and Ponemon Institute. *2012 Cost of Cyber Crime Study: United States.*
https://www.ponemon.org/local/upload/file/2012_ US_ Cost_ of_ Cyber_ Crime _ Study_ FINAL6%20.pdf.

16. Virgil D. Gligor. Dancing with the adversary: A tale of wimps and giants (article and transcript of discussion). In Proc. of the 2014 Security Protocols Workshop, Cambridge, UK (2014), no. LNCS 8809, Springer.

17. Ernst Fehr. The Economics and Biology of Trust. In *Journal of the European Economics Association*, vol. 7, April-May, 2009.

18. V. D. Gligor and J. M. Wing. Towards a Theory of Trust in Networks of Humans and Computers (also see transcript of discussion). In Proc. of Security Protocols Workshop (SPW XIX), Cambridge University, LNCS 7114, Springer, pp. 223–242, 2011.

19. VentureBeat Staff. Report: US businesses experience-42-cyberattacks-per-year. Sept. 20, 2022.
https://venturebeat.com/security/report-u-s-businesses-experience-42-cyberattacks-per-year/.

20. National Security Agency. Embracing a Zero Trust Security Model, 2021. *https://media.defense.gov/2021/Feb/25/2002588479/1/1/0CSI_EMBRACING_ZT_SECURITY_MODEL_UOO115131-21.PDF.*

21. Future Market Insights. Cybersecurity Insurance Market Snapshot (2022 – 2032). *https://www.futuremarketinsights.com/reports/cybersecurity-insurance-market.*

22. Adrian Mak. Cyber Insurance Cost by Industry. AdvisorSmith, Nov. 2021. *https://advisorsmith.com/business-insurance/cyber-liability-insurance/cost-by-industry/.*

23. NAIC Staff. Report on the Cyber Insurance Market, Memorandum, Oct. 18, 2022. *https://content.naic.org/sites/default/files/cmte-c-cyber-supplement-report-2022-for-data-year-2021.pdf.*

24. Rezilion and Ponemon Institute. The State of Vulnerability Management in DevSecOps, Sept. 2022.
https://www.rezilion.com/wp-content/uploads/2022/09/Ponemon-Rezilion-Report-Final.pdf.

25. Tim Keary. Vulnerability management: Most orgs have a backlog of 100K vulnerabilities. In *VentureBeat*, Sept. 2022.
https://venturebeat.com/security/vulnerability-management-most-orgs-have-a-backlog-of-100k-vulnerabilities.

26. Roberto Torres. Enterprise App Sprawl with most apps outside IT control. In CIO Dive, Sept 2021.
https://www.ciodive.com/news/app-sprawl-saas-data-shadow-it-productiv/606872/.

27. Andrea Vittorio. Merck's $1.4 Billion Insurance Win Splits Cyber From "Act of War." in *Bloomberg Law*, Jan. 19, 2022.
https://news.bloomberglaw.com/privacy-and-data-security/mercks-1-4-billion-insurance-win-splits-cyber-from-act-of-war

28. Shmulik Yehezkel. The cost of cybersecurity insurance is soaring–and state-backed attacks will be harder to cover. It's time for companies to take threats more

seriously. in *Fortune*, Feb. 15, 2023.
https://fortune.com/2023/02/15/cost-cybersecurity-insurance-soaring-state-backed-attacks-cover-shmulik-yehezkel/

29. Robert Joyce. Disrupting Nation State Hackers. Invited Keynote at USENIX Enigma Conference, Jan. 2016.
https://www.youtube.com/watch?v=bDJb8WOJYdA.

30. Sarbari Gupta and Virgil D. Gligor. Towards a Theory of Penetration-Resistant Computer Systems. In *Journal of Computer Security, vol. 1*, no. 2, pp. 133-158, (April 1992) (also in Proc. of 4th IEEE Computer Security Foundations Workshop, Franconia, New Hampshire, pp. 62–78, (June 1991)).
https://content.iospress.com/articles/journal-of-computer-security/jcs1-2-02.

31. Sarbari Gupta and Virgil D. Gligor. Experience with a Penetration Analysis Method and Tool. In Proc. of 15th National Computer security Conference, Baltimore, MD, pp. 165-183 (October 1992).
https://csrc.nist.rip/publications/history/nissc/1992-15th-NCSC-proceedings-vol-1.pdf.

32. Byron Cook. Formal reasoning about the security of Amazon Web Services. In Computer Aided Verification: 30th International Conference, CAV 2018, Held as Part of the Federated Logic Conference, FloC 2018, Oxford, UK, July 14-17, 2018, Proceedings, Part I 30. Springer International Publishing, 2018.
https://link.springer.com/chapter/10.1007/978-3-319-96145-3_3.

33. John Backes *et al.* One-Click Formal Methods. In *IEEE Software*, Vol. 36, No. 6, Nov.-Dec. 2019, pp 61–65.
https://doi.org/10.1109/MS.2019.2930609.

34. Laurent Chuat, Markus Legner, David Basin, David Hausheer, Samuel Hitz, Peter Mueller, and Adrian Perrig. *The Complete Guide to SCION – From Design Principles to Formal Verification.* In ISC Series, Springer 2022.

35. Virgil D. Gligor. Zero Trust in Zero Trust? CMU CyLab Technical Report 22-002 December 17, 2022.
https://www.cylab.cmu.edu/_files/pdfs/tech_reports/CMUCyLab22002.pdf

36. Tony Bradley. Shifting Cybersecurity to a Prevention-First Mindset. In *Forbes,* March 26, 2023. *https://www.forbes.com/sites/tonybradley/2023/03/26/shifting-cybersecurity-to-a-prevention-first-mindset/?sh=209bbc4359cc.*