Architectural Security Regulation

Adam Hastings, Ryan Piersma, Simha Sethumadhavan

***Abstract*—Across the world, governments are instituting regulations with the goal of improving the state of computer security. In this paper, we propose how security regulation can be formulated and implemented at the architectural level. Our proposal, called FAIRSHARE, requires architects to spend a predetermined fraction of system resources (e.g., execution cycles) towards security but leaves the decision of how and where to spend this budget up to the architects of these systems. We discuss how this can elevate security and outline the key architectural support necessary to implement such a solution. Our work is the first work at the intersection of architecture and regulation.**

I. Introduction

There is presently a large push across governments to improve computer security through regulations and mandates [1], [2], [3], [4], [5], [6], [7]. These are aimed at incentivizing better security practices with the goal of improving overall security. Just as security is a full system property (where each layer of the system has to be secure for a system to be secure), regulations must consider each level of the system stack to be effective. In this paper we ask the question, what should architectural level regulation look like?

Our stance on architectural regulation is threefold. First, it should not be framed as punishment: Holding vendors accountable for security flaws that cross the hardware-software interface (like memory or speculation unsafety) is challenging because it is not easy to pinpoint root causes to software, hardware, or programming error. Second, architectural regulation should not delve into the specifics of which security classes or techniques should be implemented: While such detailed specification works for standardized cryptographic algorithms, issues like memory and speculation safety are too broad and too dependent on system implementation to be meaningfully presented as enforceable regulation[1]. Furthermore, a heavy-handed, fine-grained regulatory approach (e.g., specifying memory safety in terms of memory tag lengths) could compel the implementation of specific defenses at specific strength levels; However, this approach would hinder manufacturers' ability to innovate and would not allow for swift adaptation to the ever-changing strategies of attackers. Third, we want users to also do their part for security, and thus want to incentivize users to keep architectural security mechanisms on during deployment.

[1]To be clear we are not saying that these techniques defy taxonomization. Our stance is that a taxonomy alone is not sufficient for regulation. As an example: While it might be possible for a regulator to say that hardware must support memory safety and even come up with a fairly stable taxonomy of different kinds of memory safety based on current knowledge of attacks/defenses, in practice, it matters a lot to what degree each memory safety class is actually supported, which is hard to quantify. For instance, while ARM processors support memory safety via MTE, due to speculative execution vulnerabilities, this defense can be completely negated. Should this be counted as hardware support for memory safety by a regulator?

FAIRSHARE (Fair Architecture-Inclusive Regulation for Secure Hardware via Allocated REsources) is one of many potential solutions for security regulation at the architectural level, which encourages a wide range of solutions and at the same time creates a level playing field. The basic idea is simple: *Require computer architects to allocate a fixed percentage of a product's resources towards security*. This security budget applies to traditional budgeted resources like product development costs, which represent non-recurring costs, *and also includes system resources like execution cycles or energy*, which account for recurring expenses.

Our proposal naturally brings up the question: how does one determine what percentage of resources should be allocated towards security? Rather than rely on social processes to determine this number, we demonstrate a quantitative approach: We design a model of a security game played between groups of Attackers and Defenders, using real-world data where possible, and then run simulations over the games' parameter space to study how the resource set-aside for security affects game dynamics and outcomes. Our data show that, for our simulations, a security budget of 20–40% provides the most protection at the least cost.

Also, for any regulatory proposal to be useful it needs to be enforceable. For our proposal this means that we need a way to measure how many resources are being spent on security in the field. We demonstrate the feasibility of doing this measurement using neural network regression models and show that the runtime overhead of security defenses can be captured with precision and at low cost.

II. Our Solution: FAIRSHARE

FAIRSHARE is a four-step process of interactions and responsibilities between regulator, manufacturer, and user, shown in Figure 1 and described below.

❶ The regulator (or the government) requires that products for a certain sector (e.g., healthcare or critical infrastructure) dedicate at least a fixed percentage of resources towards security. For the computer architect this translates to a requirement that at least a fixed fraction of execution cycles and/or energy consumption be dedicated to security. In Section III, we describe how a regulator or government can go about determining the resources to be spent towards security.

❷ The computer architect chooses how to spend the security budgets as they see fit. The architect selects the defenses that they see as providing the most security benefit for their product and deployment, and publishes their chosen allocation of the security budget as part of the product's description (similar to security labels [3]). The architect then also has to come up with techniques to measure the on-device security resource allocation to confirm that the mandate is being met

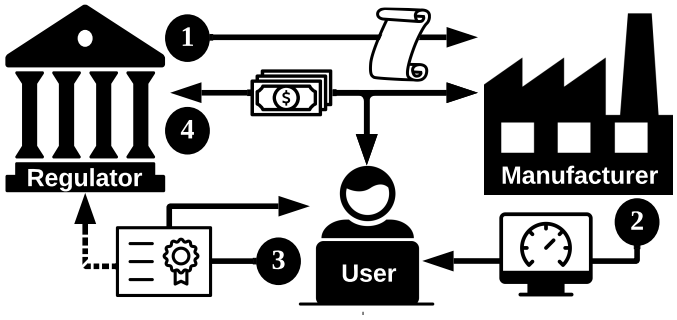


Fig. 1: The FAIRSHARE mechanism outlined in four steps.

during real product usage (a practical implementation of this is provided in Section IV).

3 The processor measures the on-device system behavior at the end user to ensure the system is allocating a sufficient amount of resources towards security. This is periodically recorded in audit logs generated by the device. The user can examine the logs to verify that the mandate is being met locally on their device. The user may submit the logs to the regulator.

4 The regulator monitors and manages the FAIRSHARE system as a whole to ensure that all parties are fulfilling their roles and responsibilities. The regulator may encourage users to submit audit logs in exchange for incentives like tax credits or deductions (this also disincentivizes users from turning defenses off). Manufacturers found to be in violation of the mandate may be fined to encourage compliance.

Benefits: FAIRSHARE produces a virtuous cycle of architecture-based security improvements, for three reasons:

1) FAIRSHARE removes the obstacles for adopting cross-cutting architecture-based defenses. Under FAIRSHARE, architects are given the (performance) budget needed to defend systems against longstanding and critical weaknesses like memory safety without the fear of losing marketplace competitiveness for allocating performance for security.

2) FAIRSHARE puts security decision-making in the hands of the domain experts. An issue with many regulations is that those who write the rules (i.e. regulators, policymakers, and even subject matter experts) are not involved in day-to-day engineering and may not be able to keep up with the current security attack landscape. For architecture security regulation, the risk is that the regulators may not fully understand the burdens of their demands or are uninformed as what which parts of a systems are most in need of security investments. FAIRSHARE avoids this problem by letting the manufacturers themselves decide where to spend the security budget.

3) FAIRSHARE fairly distributes the burden of security. One reason why insecurity persists is that it is not clear who should have to pay to fix it [8]. FAIRSHARE presents a fair solution to this dilemma by making *all* players in the game of security pay for at least *part* of the burden: regulators by enforcing regulations, manufacturers by building defenses, and users by enduring runtime overheads.

III. How Should Security Budgets be Set?

FAIRSHARE asks manufacturers to allocate a fraction of resources towards security. In this section, we propose and demonstrate a methodology for determining what this fraction might be. Our novel finding is that a higher security budget is not necessarily better, and that there is an optimum when security efficiency is considered.

Parameter	Definition
`MANDATE`	% of defender assets that are spent on security
`ATTACKERS`	Number of Attackers (as a % of number of Defenders)
`PAYOFF`	% of a defender's assets that can be stolen in an attack
`EFFICIENCY`	% of `MANDATE` by which the cost to attack increases
`INEQUALITY`	Amount of Attackers' collective wealth (as a % of Defenders' wealth)
`WAGER`	% of a defender's assets that count toward `D.atk_cost`

TABLE I: Simulations are initialized with six parameters, each of which can take a value between 0.1 and 1 at 0.1 intervals, except for `MANDATE`, which can also take the value 0.

We use an agent-based simulation with agents split between two groups, Attackers and Defenders. Agents are initialized with a fixed number of tokens that represent their wealth/resources. Attackers try to gain tokens by stealing from Defenders while Defenders try to prevent losses by making investments in security. The system is inefficient when the level of defense spending (i.e. `MANDATE`) does not minimize Defenders' overall losses.

In an effort to mirror reality, we make the following assumptions:

- Agents' initial tokens are drawn from a lognormal distribution (approximating the distribution of global wealth [9]).
- There are more Defenders than Attackers.
- At initialization, Defenders are wealthier than Attackers.
- Defenders exhibit a range of "security posture" (i.e. some are more vulnerable to attack than others).
- Attacks on wealthier defenders yield a higher payoff for the attacker but are also more expensive to accomplish.
- Attack success is probabilistic in nature but partially depends how much a Defender invests in security.
- A Defender can prevent attacks (or at least make attacks less likely) by making security investments.
- Attackers know Defenders' wealth, but Defenders do not know Attackers' wealth.
- Attackers only attack if expected earnings are > 0.

We formalize these heuristics into a set of six discretized game parameters in Table I. Since we do not know which configuration of parameters best approximates the real world, we simulate all possible 1.1×10^6 parameter combinations.

To model the probabilistic nature of attacks, each defender `D` is initialized with a probability `D.p_atk_success`, which represents "security posture" or the probability that a given

attack will be successful[2]. Defenders are also initialized with `D.attack_cost` = `D.tokens` × `WAGER` to represent how much an attacker must spend to attempt an attack.

Gameplay is iterated over a series of rounds. During each round, each attacker is paired to "fight" with a randomly selected defender by following Algorithm 1. Rounds are iterated until either all defenders lose all their tokens, or the game converges to a stable equilibrium[3].

Algorithm 1 Fight between an Attacker `A` and Defender `D`

```
loot ← D.tokens × PAYOFF
expected_earnings ← loot × D.p_atk_success
if expected_earnings > D.atk_cost then
    if D.atk_cost < A.tokens then
        r ← random.uniform(0,1)
        if r < D.p_atk_success then  // A wins
            D.tokens ← D.tokens − loot
            earnings ← loot − D.atk_cost
            A.tokens ← A.tokens + earnings
        else // D wins
            A.tokens ← A.tokens − D.atk_cost
```

A. Simulation Results and Findings

To narrow the large parameter space, we prune the games where Defenders do not suffer losses (since these parameter settings are not representative of the real world).

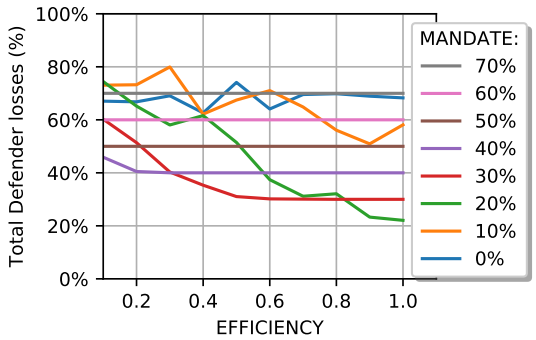
We calculate the expected value of each parameter (`MANDATE`=0.2, `ATTACKERS`=0.5, `PAYOFF`=0.8, `EFFICIENCY`=0.5, `INEQUALITY`=0.5, and `WAGER`=0.3) to give us a "baseline" parameter setting; the distance from 0.5 gives a relative measure of how strongly biased a parameter is towards favoring Attackers or Defenders.

Using this baseline configuration, we observe how a given mandate affects total Defender losses (= mandate cost + losses to Attackers) across a range of parameter values. This approach reveals—for each value of each parameter—which level of `MANDATE` minimizes Defenders' collective total losses.

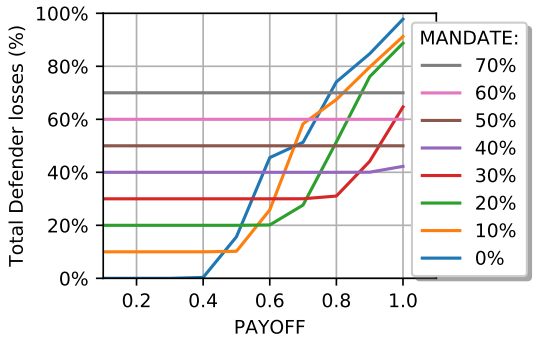
To illustrate, see Figure 2a, which sweeps across values of the `EFFICIENCY` parameter for various levels of `MANDATE`. If `EFFICIENCY` < 0.3 (meaning that security spending does little to raise security posture) then losses are minimized when `MANDATE`=40%. However, if $0.3 \leq$ `EFFICIENCY` ≤ 0.8, then losses are minimized when `MANDATE`=30%. When `EFFICIENCY` > 0.8, only a 20% security mandate is needed to minimize Defender losses. For all values of `EFFICIENCY`, `MANDATE` $< 20\%$ fails to sufficiently protect against losses while `MANDATE` $> 50\%$ is too costly to justify the additional protection.

[2]This probability is drawn from a normal distribution with mean and standard deviation inferred from an industry report that finds, on average, 39% of ransomware attacks are successful with a standard deviation of 6.2% [10].

[3]In our games, convergence is defined as when the sum total of all Defenders' tokens changes by less than ϵ for at least δ rounds of the game. We choose $\epsilon = 100$ (which is very small relative to Defenders' initial sum tokens) and $\delta = 50$ (which is a significant fraction of most games' number of rounds). No games failed to reach a stable equilibrium.



(a) Defender losses across a sweep of `EFFICIENCY` for various `MANDATE` levels. The optimal `MANDATE` is 20%, 30%, or 40%, depending on the value of `EFFICIENCY`.



(b) Defenders losses across a sweep of the `PAYOFF` for various `MANDATE` levels. The optimal `MANDATE` ranges from 0% to 40% depending on the value of `PAYOFF`.

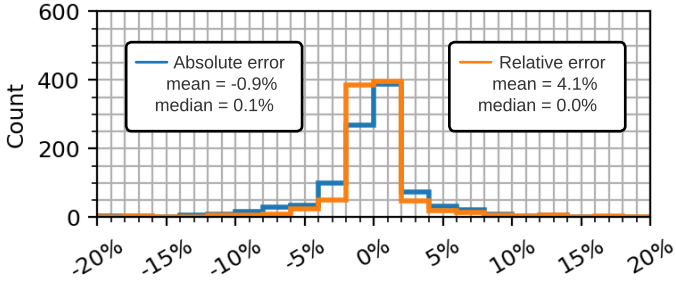
Fig. 2: Dynamics of the simulation are revealed by sweeping parameters across their full range of values while simultaneously holding all other parameters at their "baseline" value.

We repeat for the `PAYOFF` parameter in Figure 2b. When `PAYOFF` < 0.5, Defenders suffer no losses at all (presumably because there is not sufficient incentive for Attackers to attack). As `PAYOFF` increases, losses are minimized under a 10%, 20%, 30%, and finally a 40% `MANDATE` (when `PAYOFF` ≥ 0.9). Above 40%, the cost of the mandate itself outweighs the protections it provides. While not shown here, we add that for the `WAGER` parameter, Defender losses are also minimized when $0.0 <$ `MANDATE` < 0.4. We omit plots for the `ATTACKERS` and `INEQUALITY` parameters, since we find that these parameters do not have an effect on Defenders' collective losses. Hence losses are collectively minimized across all parameters when $0.2 \geq$ `MANDATE` ≥ 0.4.

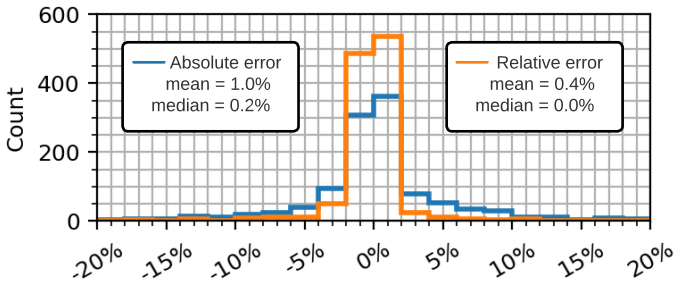
The benefit of this approach is that we can observe what security budget levels are most effective across the full range of parameter values without requiring us speculate on what these values might actually be in the real world. For example, although there is undoubtedly some relationship between security spending and resulting security posture, the real-world function between the two is largely unknown due to a lack of publicly available data. Our approach allows us to build models from reasonable heuristics and make observations about the dynamics of security in the absence of strong quantitative real-world measurements.

IV. Implementing FAIRSHARE

FAIRSHARE requires a mechanism that can measure resources allocated for security. This requires measurements to be taken **on-device** and **continuously** (since pre-computed overheads are only valid for the test devices and benchmarks used and are highly sensitive to changes in system configuration and usage behavior).



(a) Binary hardening overhead estimation errors



(b) NoFAT overhead estimation errors

Fig. 3: Our DNN-based regression model estimates real-time security overhead with high precision and accuracy.

To achieve this, we train a predictive neural network model that leverages system data collected via specially chosen hardware performance counters ("HPCs") that track key indicators of runtime performance. First, we create training data by running the SPEC2017 benchmark suite with and without defenses applied and collect HPC data at each system call boundary (using the tools DynamoRIO and easyperf). Next, we perform sequence alignment on execution traces of the "baseline" and "secured" variants (using collected syscall numbers) to find periods of equivalent execution and determine the defenses' performance overhead *at each system call*.

Using this as our training data, we train a model using PyTorch that inputs 20 HPC events and predicts current security overhead as a scalar output[4]. We use a 90/5/5 split for the training, testing, and validation sets, respectively. To avoid overfitting, we choose the model that minimizes the validation set's loss. We also reduce model size by using half-precision (16-bit) weights to highlight the feasibility of building the model in hardware and achieve a final model size of 12KB.

We use the above process for two defenses: compiler-based binary hardening flags[5], and NoFAT [11]. We evaluate the models by comparing predicted overhead to observed overhead using both absolute and relative error. Results are plotted in Figure 3. For both defenses, we find that our models estimate performance overheads with very high accuracy and precision.

[4]We use four layers with Leaky ReLU activation functions. We used the Adam optimizer with a learning rate of 10^{-3}, an epsilon of 10^{-4}, and Mean Squared Error (MSE) as our loss function.

[5]Specifically, `-fPIE -pie -D_FORTIFY_SOURCE=2-Wl, -z now, -z relro, -fstack-protector-all -fsanitize=safe-stack`

V. Related Work

We are not aware of any other work at the intersection of regulation, systems design and security. However, we do find that some prior works aim to predict computer performance from provisioned hardware resources and configurations in an effort to design better and more efficient systems [12], [13]. Other work estimates benchmark program runtimes from program semantics and machine characterizations combined [14] or predicts energy overheads in addition to performance [15], [16], [17]. To our knowledge, our approach is the only attempt to estimate runtime predictions of performance and the only to make real-time predictions of security overheads.

VI. Conclusion

FAIRSHARE is a combined technical-regulatory solution to the shortcomings of existing regulatory techniques when applied to architecture. FAIRSHARE asks manufacturers to dedicate some fraction of resources (both organizational and system-based) towards security but without prescriptively telling manufacturers how to do so. We provided simulations to help decide what an appropriate fraction might be, and then demonstrated how measuring security resource allocation in situ and on device might be achieved.

References

[1] Executive Office of the President, "Executive Order 14028," May 2021.
[2] The White House, "National cybersecurity strategy," Mar 2023.
[3] The White House, "Biden-harris administration announces cybersecurity labeling program for smart devices to protect american consumers," July 2023.
[4] E. U. European Commision, "Proposal for a regulation of the european parliament and of the council on horizontal cybersecurity requirements for products with digital elements and amending regulation (eu) 2019/1020," tech. rep., European Commission, 2022.
[5] M. Department for Digital, Culture and U. K. Sport, "New smart devices cyber security laws one step closer," *Press release*, 2022.
[6] TRAFICOM, "Statement of compliance for the cybersecurity label," tech. rep., National Cyber Security Centre, Finland, 2019.
[7] C. C. Centre, "Cybersecurity certification guide," tech. rep., Cyber Security Agency of Singapore, 2021.
[8] A. Hastings and S. Sethumadhavan, "Wac: A new doctrine for hardware security," in *Proceedings of the 4th ACM Workshop on Attacks and Solutions in Hardware Security*, pp. 127–136, 2020.
[9] Credit Suisse Research Institute, "Global wealth report 2021," June 2021.
[10] Sophos, "The state of ransomware 2021." Accessed: 2021-11-2.
[11] M. T. I. Ziad, M. A. Arroyo, E. Manzhosov, R. Piersma, and S. Sethumadhavan, "No-fat: Architectural support for low overhead memory safety checks," in *2021 ACM/IEEE 48th Annual International Symposium on Computer Architecture (ISCA)*, pp. 916–929, IEEE, 2021.
[12] J. W. Boyse and D. R. Warn, "A straightforward model for computer performance prediction," *ACM Computing Surveys (CSUR)*, vol. 7, no. 2, pp. 73–93, 1975.
[13] P. Ein-Dor and J. Feldmesser, "Attributes of the performance of central processing units: A relative performance prediction model," *Commun. ACM*, vol. 30, p. 308317, apr 1987.
[14] R. H. Saavedra and A. J. Smith, "Analysis of benchmark characteristics and benchmark performance prediction," *ACM Trans. Comput. Syst.*, vol. 14, p. 344384, nov 1996.
[15] S. Penolazzi, L. Bolognino, and A. Hemani, "Energy and performance model of a sparc leon3 processor," in *2009 12th Euromicro Conference on Digital System Design, Architectures, Methods and Tools*, pp. 651–656, 2009.
[16] S. Penolazzi, I. Sander, and A. Hemani, "Inferring energy and performance cost of rtos in priority-driven scheduling," in *International Symposium on Industrial Embedded System (SIES)*, pp. 1–8, 2010.
[17] S. Penolazzi, I. Sander, and A. Hemani, "Predicting energy and performance overhead of real-time operating systems," in *2010 Design, Automation & Test in Europe Conference & Exhibition (DATE 2010)*, pp. 15–20, IEEE, 2010.