Generating Statistically Relevant Trojan Benchmarks for Microelectronics Quantifiable Assurance

Margaret Winslow, Whitney Batchelor, James Koiner, Kevin Paar, Scott Harper, Jonathan Graf
Graf Research Corporation
Blacksburg, Virginia, USA
trust@grafresearch.com

***Abstract*—Hardware trojan horse (HTH) detection metrics are used to quantify the value of trojan detection methods. These metrics, often in terms of probability of detection and probability of false alarm, can be used to help quantify the impact on design assurance when applying mitigations to a microelectronics circuit. A question arises, however, regarding how statistically sound the metric values must be to make reasonable trust and assurance decisions. Statistical relevance metrics have been used in many fields to justify confidence in claims, and benchmarks that can produce statistically relevant detection metrics are necessary to trust the quantification of microelectronics assurance. This work defines the requirements for generating statistically relevant detection metrics that are useful for quantifying microelectronics design assurance via testing with a strategically implemented circuit design benchmark set.**

Keywords—Hardware Trojan Horse; Statistical Relevance; Microelectronics; Trust; Confidence; Margin of Error; Assurance

I. INTRODUCTION

Production of quantifiably assured microelectronics requires benchmark testing to understand the expected impacts of design assurance methodologies applied during design and development processes. Those benchmarks must encompass a variety of golden circuit designs and circuit designs containing Hardware Trojan Horses (HTHs) for them to be useful in producing accurate measurements reflective of different microelectronic design styles and features. Much of HTH detection literature describes how different detection methodologies perform against HTH attacks. Many of those descriptions are based on a limited set of circuit designs, often with a limited scope of design types. We aim to answer the question: what is the sufficient level of testing required to claim accuracy of detection metrics with an acceptable level of confidence?

The value of detection metrics should not be dependent on purely random variation in testing but should rather be dependent on the technique used to produce the metric, thus the value of the metric should be statistically relevant [1]. Calculating the statistical relevance of a metric requires that the benchmark set used for testing be large and representative of the total population of circuit designs for which it claims to apply. The minimum size determination of such a benchmark set depends on the desired confidence level and margin of error to which one wants to claim that the metric holds. Representation within a benchmark set depends on the diversity of the circuit designs in the population the benchmark should represent.

We describe the requirements for the development of a benchmark containing a mix of circuit designs with HTHs and golden circuit designs to be used as a sample set for the derivation of statistically relevant and broadly applicable HTH detection metrics.

II. BACKGROUND

Empirical trojan detection efficacy metrics can be produced by running detection methods against the subset of the benchmark consisting of circuit designs that contain HTHs. Empirically derived false alarm metrics can be produced by running methods against the subset of the benchmark consisting of golden circuit designs. This approach provides both the probability of detection (P_D) and the probability of false alarm (P_{FA}) for any given assurance method. Developing such a benchmark set to use for deriving statistically relevant HTH detection metrics requires that the sample set be large and representative of the population. Statistical relevancy measurements, specifically confidence level and margin of error, depend on this sample size of the test set.

The *confidence level* describes the certainty that the true metric value for the population is near the sample metric value. A *margin of error* describes the range around the sample value that should be included when considering the population value. Confidence levels and margin of error values are in turn used to build *confidence intervals*, which describe how confident the researcher is that the true population metric value is within an interval [3]. Calculations of example benchmark sample sizes for providing quality trojan detection metrics are shown in the following section.

Creating a representative sample depends on the unique attributes of the population of circuit designs. Ideally, a simple random sample of circuits could be obtained from the circuit design space. However, in the absence of access to all circuit designs, a repository of circuits that is representative of the population must be produced. Ontologies describing unique features of circuit designs and HTHs can be generated to assist in explaining representation within a sample. The ontologies created to describe circuit designs and HTHs must be comprehensive in describing the relevant features of both circuits and HTHs that they might contain, leading to potential categories of possible designs within the population. These ontologies can then be used to describe the designs within any given sample to show that the benchmark set is representative of designs that are present within the population. An example of a set of ontologies describing ASIC and FPGA designs is included in the next section. HTH ontologies are widely available within

Distribution A: Approved for public release. This material is based upon work supported by the United States Air Force under Contract No. FA8650-20-F-1880. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the United States Air Force.

HTH literature [6,7,8], and we are adapting a variety of HTH ontologies to assist in describing the automated production of a large set of circuit designs containing HTHs.

Determining whether a benchmark set is representative of a population depends on the proportion of the population that belongs to each ontology entry. If the proportion of all distinct categories within the population is known, the benchmark should have similar proportions of designs in each category. However, if the population proportions are unknown, the benchmark must be produced using a disproportionate stratified sampling method [2]. The process of creating a set of circuit designs through a stratified sampling technique assists in producing a representative sample set. Designs should also be pulled from multiple sources, if possible, to assist in mitigating biases present in any given developer's knowledge of or approach to circuit design.

By creating a large, representative sample that limits bias, confidence intervals can be wrapped around empirically derived HTH detection metrics to describe their statistical relevancy.

III. Benchmark Production for Statistically Relevant HTH Detection Metrics

As described in the previous section, a large and representative benchmark is required to develop metrics that are statistically relevant to a population. Determining how large that benchmark must be and what is required for the benchmark to be considered representative of the population is explored here.

A. Calculating Sample Size

Calculating the minimum sample size of a test set requires four metrics: population size, expected proportion, confidence level (CL), and margin of error (MoE). Since the population size of all circuit designs is unknown and constantly increasing, it is considered infinite for calculations to ensure that the sample sizes used produce accurate confidence intervals. The expected proportion is the number of expected successes divided by the number of outcomes in a population. For example, the expected proportion of a probability of false alarm for a specific detection method would reference the number of expected detections when running the detection method on all golden circuit designs divided by the number of golden circuit designs in the population. Since this benchmark will be used to measure different efficacy metrics for many different detection methods, the sample size should be maximized to cover all possible expected proportions. The expected proportion that results in a maximal sample size is used to calculate desired sample sizes. The expected proportion multiplied by the additive inverse of the expected proportion, $\hat{p}(1-\hat{p})$, has a direct relationship with the sample size, n. Note that

$$n = \frac{\left(z_{\frac{\alpha}{2}}\right)^2 \left(\hat{p}(1-\hat{p})\right)}{\text{MoE}^2} \quad (1)$$

where $z_{\alpha/2}$ is the z-score, or the number of standard deviations from the mean, associated with the given confidence level, $\hat{p}$ is the expected proportion, and MoE is the desired margin of error [4]. To maximize sample size and ensure that a test set meets the requirements for CL and MoE no matter the expected proportion, $\hat{p}(1-\hat{p})$ should be maximized. The value $\hat{p}(1-\hat{p})$ is at most 0.25 when $\hat{p}$ is 0.5 with the requirement $0 < \hat{p} < 1$.

Proof. Let $x \in (0,1)$. Assume towards contradiction that $x(1-x) > 0.25$. It follows that

$$x - x^2 > 0.25$$
$$\leftrightarrow -x^2 + x - 0.25 > 0$$
$$\leftrightarrow -(x^2 - x + 0.25) > 0$$
$$\leftrightarrow (x^2 - x + 0.25) < 0$$
$$\leftrightarrow (x - 0.5)^2 < 0.$$
$$\Rightarrow\Leftarrow$$

Thus, there does not exist an $x \in (0,1)$ such that $x(1-x) > 0.25$, and when $x = 0.5$, $x(1-x) = 0.25$. Hence, when $x = 0.5$, the value of $x(1-x)$ is maximized. □

Given this analysis, a value of 0.5 is used for the expected proportion when determining minimal desired sample size. The last two metrics needed for calculating the sample size are the desired MoE and z-score, which is derived from the desired CL. When creating a benchmark set to be used for statistically relevant test metric production, the minimum test set sample size for varying combinations of MoE and CL values should be considered as production goals. Two common confidence levels used for verifying statistical relevancy of a metric are a 95% CL and a 99% CL. When calculating sample size, the z-score associated with the confidence level is required, and a 95% CL is associated with a z-score of 1.96, while a 99% CL is associated with a z-score of 2.58 [10].

Using the metrics described above, minimum sample sizes were calculated with varying MoE and CL requirements, as seen in Table 1. Testing an HTH detection tool against a representative subset of the benchmark with one of the sample sizes provided in Table 1 guarantees a MoE and CL specified in the column and row labels. Here, we see a 95% confidence level with a 10% margin of error can be achieved with a sample size of 97. If a specific defense method detects a specific trojan on 80 of the 97 circuit designs with an HTH, then the test set probability of detection is 82.47%. Given that the size of the test set was 97 circuit designs with the HTH, we are 95% confident that the actual probability that the defense method can detect the HTH is within a 10% MoE of 82.47%, or between 72.47% and 92.47%.

Table 1. Sample Sizes for Specified MoE and CLs

MOE	SAMPLE SIZE WITH 95% CL	SAMPLE SIZE WITH 99% CL
1%	9,604	16,641
2%	2,401	4,161
3%	1,068	1,849
4%	601	1,041
5%	385	666
6%	267	463
7%	196	340
8%	151	261
9%	119	206
10%	97	167

A common initial goal for research is to obtain a 95% CL and 10% MoE for any sample metric [5]; however, higher CL and lower MoE values yield higher quality sample metrics. At minimum, tools should be tested against at least 194 circuit designs, 97 with HTHs for detection metrics and 97 without HTHs for false alarm metrics, to claim that produced metrics are statistically relevant to the population.

Quantifiably assured microelectronics require quantified metrics that indicate the effectiveness of steps taken to trust the design. For example, assume a developer applies detection method A and detection method B to a design. The individual probability of detection values for each method can be combined into an assurance metric for how quantifiably assured the design is against a specific HTH that detection methods A and B both detect. Assuming that detection methods A and B are independent, the developer uses the following equation for the assurance metric:

$$P_D(A \cup B) = P_D(A) + P_D(B) - P_D(A)P_D(B), \quad (2)$$

as $P_D(A \cup B)$ describes the probability that the specific HTH is detected when both A and B are used. Since $P_D(A)$ and $P_D(B)$ both have associated confidence levels and MoE values, the resulting assurance metric will have a cumulative error associated with it. Cumulative error is a widely studied topic within statistics and can be applied to determine the cumulative MoE of such assurance metrics depending on how empirically derived values are combined in assurance calculations [9]. Higher levels of statistical relevancy for the individual metric values results in increasing confidence in the level of trust assigned to the microelectronics design. Having a large, readily accessible benchmark set for testing tools results in highly accurate metrics that can be combined into verifiable assurance metrics with small margins of error.

B. Producing a Representative Benchmark Set

Composing a benchmark set that is representative of a population depends on defining distinct attributes of circuit designs within the population. This can be achieved through the development of ontologies that describe varying aspects of circuit designs. Two representative ontologies that can describe the population of ASIC and FPGA circuit designs are a functionality ontology and design size ontology. Functionality describes the purpose of a circuit design. Simple designs may only have one functionality class, while large composite designs most certainly have many functionality tags. A set of functionality classes used in a representative ontology is described in Table 2. Design size categorizes a circuit design by the count of foundational technology blocks utilized in design implementation. A given circuit design can apply to multiple entries in the functionality ontology but can only apply to one design size entry. The set of functionality entries and the design size entry is considered the tag set or classification of the design.

A high-level design size ontology is useful for ensuring evaluation methods are not limited by either a lack of design elements or an approach that does not scale well in time or memory usage with element counts. It also provides an intuitive way for developers to understand how detection metrics directly relate to the size of the design. The design size ontology in Table 3 considers every category of foundational technology blocks, such as ASIC cells or FPGA primitives, within the implementation utilization report. All categories are summed linearly (i.e., all cells/primitives are evenly weighted as a single unit regardless of complexity) to calculate the size as a total count of reported foundational technology blocks.

TABLE 2. FUNCTIONALITY DESCRIPTIONS

Functionality Tag Name	***The Design Contains…***
Crypto	Cryptographic functionality such as AES, SHA, or RSA engines, or cryptographic random number generators.
External Comms	An external communications controller such as Ethernet, Serial, PCI, USB, etc., where the controller must implement the specifics of the communication protocol.
External Memory	An external memory controller such as a DDR DRAM, or SRAM controller responsible for the direct management of the memory media. This does not include memory that utilizes a multi-purpose communications bus such as QSPI, which would be covered in the External Comms class.
Internal Bus	Internal (to the FPGA/ASIC) bus interconnect or support functionality. This includes things like AXI bus interconnect, bus translators, arbiters, and network-on-chip (NOC) infrastructure.
Math	Math cores including floating point, multipliers/dividers, CORDIC, and trig functions. This does not include code-executing floating-point co-processors, which are covered under the Processor class.
Internal Memory	Internal memory structures including things such as FIFOs and distributed memory implemented using block RAM and other similar FPGA/ASIC structures.
Peripheral	Other cores typically considered CPU peripherals such as programmable interrupt controllers (PIC), direct memory access (DMA) engines, GPIO structures, and timing functionality (timers, clocks, watchdogs, etc.).
Processor	Code-executing processing units. Includes CPUs, floating point units (FPUs), graphics processing units (GPUs), and AI acceleration.
DSP	Digital signal processing (DSP) cores such as filters (FIR, IIR), modulation and waveform synthesis blocks, transforms (FFT, DFT), and other DSP-specific building blocks.
Video	Video or image processing specific functionality such as JPEG or MPEG encoders or decoders, color-space or resolution transforms, and motion detection.
Test	Test or debug structures such as built-in self-test (BIST) engines, scan chains like JTAG, or debug data capture or data injection capabilities.

TABLE 3. DESIGN SIZE DEFINITIONS

Design Size	*Number of Foundational Technology Blocks*
Small	(0, 1500)
Medium	(1500, 15000)
Large	(15000, 150000)
X-Large	(150000, 500000)
XX-Large	(500000+)

Representation of the population within a benchmark set can be determined using the functionality and design size tag ontologies. Since each design can only be associated with one design size category but can be associated with any combination of between one and eleven functionality tags, there are 10,235 unique tag sets. For example, the tag sets *{Small, Math}* and *{X-Large, Crypto, Internal Bus, Math, Internal Memory, Peripheral, Processor, External Comms, External Memory, DSP, Video, Test}* are both valid design classifications. Some tag sets are unlikely; for example, a design with the following functionality tags: *{Crypto, Internal Bus, Math, Internal Memory, Peripheral, Processor, External Comms, External Memory, DSP, Video, Test}* likely contains too many foundational technology blocks to be considered a small design. Although there are potentially 10,235 unique tag sets, some are infeasible to produce as a design.

If the population of circuit designs under consideration is known to have 50% of designs that have *{Medium, Math, Crypto}* classifications, 25% of designs that have *{Small, Math}* classifications, and 25% of designs that have *{Small, Math, Crypto}* classifications, then the benchmark set should have similar proportions of designs with those specific classifications. A different approach is needed to produce generally applicable metrics when proportions of tag sets are unknown for a population. Here, a representative benchmark set containing a wide variety of design classifications is required. Given a sample size of n, and a size ontology with five categories, a sample with a wide variety of design tag sets would have around $\frac{n}{5}$ of each size ontology category to provide an even distribution of design sizes. For the functionality ontology, obtaining a similar number of designs with and without each functionality tag provides representation amongst the functionality categories.

Generating a large and representative benchmark set of circuit designs, both with and without HTHs, has been described. However, subsets of the benchmark are used for deriving different detection metrics. To find probability of false alarm metrics, the subset of golden circuit designs is used as the sample set. To find probability of detection metrics, the subset of circuit designs with HTHs is used as the sample set. Note that each sample set must be large and representative to claim statistical relevancy metrics for the probability metrics being derived. While current Trojan benchmarks usually contain many more circuit designs with HTHs than golden circuit designs, to obtain statistically relevant probability of false alarm metrics, this benchmark requires a large set of golden circuit designs.

IV. EMPIRICALLY DERIVING DETECTION METRICS WITH BENCHMARK SETS

To illustrate the usefulness of access to a benchmark set specifically developed to assist in the derivation of statistically relevant HTH detection metrics, the determination of a detection rate metric for a trojan detection tool is outlined. Assume that a new trojan detection tool, HTHDetect, is introduced to the community. The creator of the tool wants to make a claim about the false alarm rate of the tool and the detection rate of the tool against a specific HTH, ExampleHTH.

Given access to a circuit design benchmark set, with a set of golden circuit designs and a set of designs with ExampleHTH inserted. HTHDetect can then be run on all circuits in the benchmark set. As HTHDetect is run, information about whether ExampleHTH was detected for each benchmark circuit is collected. Then, probability of detection and probability of false alarm values for the test sets can be calculated using the following formulas:

$$P_{FA}(\text{HTHDetect}) = \frac{\text{Total Detections on Golden Circuit Designs}}{\text{Number of Golden Circuit Designs in Benchmark}} \quad (3)$$

$$P_{D}(\text{ExampleHTH}, \text{HTHDetect}) = \frac{\text{Total Detections on Circuit Designs with ExampleHTH}}{\text{Number of Circuit Designs with ExampleHTH in Benchmark}} \quad (4)$$

Lastly, the statistical relevancy of each metric can be determined using the denominator of each equation as the sample size.

Assume that the developer of HTHDetect originally tests the tool against ExampleHTH using 10 designs with ExampleHTH. The developer determines that HTHDetect found ExampleHTH on 7 of the 10 articles and wants to make the claim that HTHDetect can detect 70% of all ExampleHTH trojans. However, since the tool was only tested on 10 circuit designs with ExampleHTH, the MoE for a 95% CL is 30%. The developer can only be 95% confident that the true probability of detection in the population is somewhere between 40% and 100%. Clearly, testing HTHDetect against a small set of circuit designs results in meaningless HTH detection metrics when considering the statistical relevance of such values. Also, note that with such a small set of designs, obtaining coverage of a wide variety of tag sets is extremely difficult and yields claims that only relate to the limited tag sets included in the benchmark circuit designs that may or may not apply across a more diverse set of designs.

Now, assume that the developer used a benchmark set designed for producing statistically relevant HTH detection metrics when testing HTHDetect. This benchmark set has 986 circuit designs, 601 with ExampleHTH and 385 golden designs, that cover a wide range of tag sets. Running the tool on the benchmark set, HTHDetect found ExampleHTH on 510 of the 601 circuit designs with ExampleHTH. This indicates that the benchmark probability of detection metric is 84.89%. Since a sample size of 601 was used, referencing Table 1, the developer can claim a 95% confidence with a 4% MoE. The developer is now 95% confident that the true population probability of detection is between 80.89% and 88.89%. HTHDetect also produced a false alarm on 10 of the 385 golden circuit designs,

resulting in a benchmark probability of false alarm rate of 2.6%. Since the developer used 385 golden circuit designs for testing, referencing Table 1, there is a 5% MoE for that metric. The developer is now 95% confident that HTHDetect has a probability of false alarm between 0% and 7.6% in ASIC and FPGA designs. As the number of circuit designs in the repository grows, the confidence intervals surrounding trojan benchmarks become more concrete.

V. Future Work

Graf Research is currently developing a benchmark set using the tag sets described in this work in addition to others. This benchmark currently contains 252 golden circuit designs and 3,982 designs containing various HTHs. Work is being done to increase both the number of designs and the distribution of tag sets included in the benchmark. While this work describes the potential for inserting a singular HTH category into a set of circuit designs to augment the benchmark set with circuit designs with HTHs for deriving detection metrics, Graf Research is continually exploring creating a benchmark that is simultaneously representative of FPGA and ASIC circuit designs and a variety of HTH categories. This work requires adaptation of HTH ontologies to ensure representation of varying HTH categories and their insertion into circuit designs. The creation of a sufficiently large set of golden circuit designs and circuit designs with HTHs requires automation across the spectrum of design implementation, trojan development, trojan insertion, and mitigation testing. That automation is another focus of ongoing work. Access to the resulting, and continuously growing, benchmark set that is comprehensive in circuit design and HTH categories will allow for the evaluation of HTH detection tools at an increased efficiency and elevated confidence.

VI. Conclusion

Understanding the statistical relevancy of benchmark produced metrics used to quantify microelectronics design assurance is necessary for trusting that the quantification is properly modeling the metric description. The analysis presented here clearly illustrates that more extensive testing than typically seen in research results is required to make reliable statements about HTH detection method efficacy. Access to a benchmark set that is strategically implemented to provide test metrics that are statistically relevant to ASIC and FPGA designs allows for increased efficiency in collecting metrics that are trusted and verifiable.

References

[1] J. D. Perezgonzalez, “Fisher, Neyman-Pearson or NHST? A tutorial for teaching data testing.” Front Psychol, vol. 6, p. 223, March 2015.

[2] Formplus Blog, “What is stratified sampling? Definition, examples, types.” Formplus, September 2021.

[3] R. H. Lock, P. F. Lock, K. L. Morgan, E. F. Lock, and D. F. Lock, *Statistics: Unlocking the Power of Data,* 3rd ed., Wiley, 2020, p. 234.

[4] OpenStax, *Introductory Statistics*, Section 8.3, Version 17.44, OpenStax College, 2015.

[5] *NIST/SEMATECH e-Handbook of Statistical Methods,* 2022, https://itl.nist.gov/div898/handbook/ppc/section3/ppc333.htm

[6] M. Tehranipoor and F. Koushanfar, “A Survey of Hardware Trojan Taxonomy and Detection,” IEEE Design & Test, Jan./Feb., 2010, pp. 10-25.

[7] J. Rajendran et al., “Towards a Comprehensive and Systematic Classification of Hardware Trojans,” Proc. IEEE Int’l Symp. Circuits and Systems (ISCAS10), IEEE Press, 2010, pp. 1871-1874.

[8] R. Karri, J. Rajendran, K. Rosenfeld and M. Tehranipoor, "Trustworthy Hardware: Identifying and Classifying Hardware Trojans," in Computer, vol. 43, no. 10, pp. 39-46, Oct. 2010.

[9] R. Hogan, “How to combine errors.” University of Reading, Department of Meteorology, June 2006. www.met.rdg.ac.uk/~swrhgnrj/combining_errors.pdf

[10] “Z Score Table,” https://www.z-table.com/