The AV says: Your hardware definitions were updated!

Marcus Botacin[1], Lucas Galante[2], Fabrício Ceschin[1], Paulo Santos[3], Luigi Carro[3], Paulo Lício de Geus[2], André Grégio[1], Marco Zanata[1]

[1]Federal University of Parana (UFPR-BR)
{mfbotacin, fjoceschin, gregio, mazalves}@inf.ufpr.br

[2]University of Campinas (UNICAMP-BR)
{galante, paulo}@lasca.ic.unicamp.br

[3]Federal University of Rio Grande do Sul (UFRGS-BR)
{pcssjunior, carro}@inf.ufrgs.br

Who Am I?

Background

- Computer Engineer (University of Campinas–Brazil).
- Computer Science Master (University of Campinas–Brazil).
- Computer Science PhD Candidate (Federal University of Paraná–Brazil).
- Malware Analyst (Since 2012).

Research Interests

- Malware Analysis & Detection.
- Hardware-Assisted Security.
- Security Co-Processors.

Topics

1. Introduction
   - The Problem
   - Background
2. Malware REHAB
   - Architecture
3. Evaluation
   - Goals
   - Hardware Accelerator
   - Reconfigurable Hardware
   - Hardware Requirements
4. Final Remarks
   - Limitations
   - Conclusion
   - Questions?

The Problem

Malware

- Self-Modifying, Obfuscated Code.
- Constantly evolving over time.

AntiViruses (AVs)

- Runtime Monitoring.
- Performance-Intensive Applications.
- Require periodic updates.

The Challenges

An Alternative for AVs

- Move AV to hardware.
- No monitoring overhead.

Hardware AV drawbacks

- Identify Low-Level Features.
- Deploy Hardware Updates.

Insight

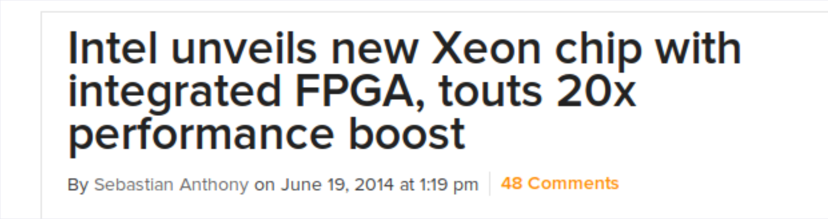
- Use Reconfigurable Hardware.

Insight Support

Past

- Hardware AVs with no update support.
- Targeting embedded systems only.

Future



Intel unveils new Xeon chip with integrated FPGA, touts 20x performance boost

By Sebastian Anthony on June 19, 2014 at 1:19 pm | 48 Comments

Source: `http://tinyurl.com/y2yabdrp`

Topics

1. Introduction
   - The Problem
   - Background
2. Malware REHAB
   - Architecture
3. Evaluation
   - Goals
   - Hardware Accelerator
   - Reconfigurable Hardware
   - Hardware Requirements
4. Final Remarks
   - Limitations
   - Conclusion
   - Questions?

Background

AV Operation Modes

- Signature-Based.
- Behavior-Based.
- Profiling-Based.

Machine Learning Classifiers

- Support Vector Machines (SVM).
- Random Forest (RF).
- Multi-Layer Perceptron (MLP).

Profiling-Based AV

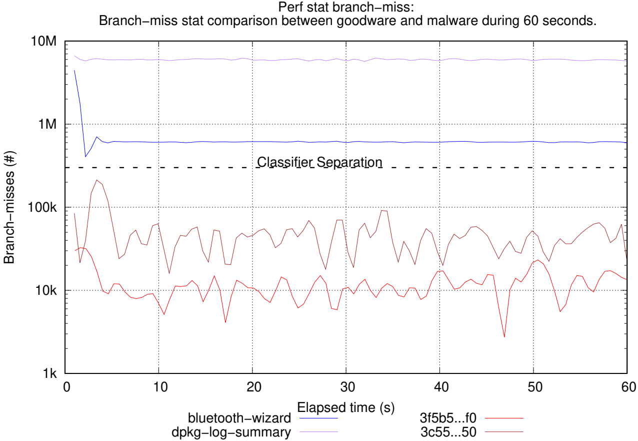


Figure: **Malware Classification using low level features.**

SVM Classifier

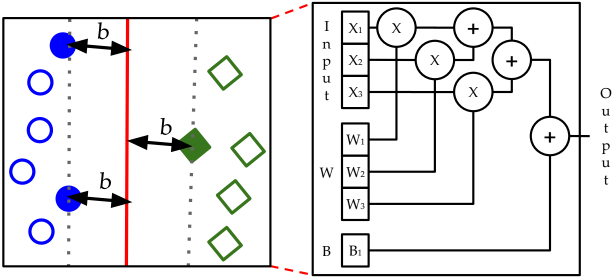


Figure: **SVM**. A hyperplane with maximum separation between two classes is created and used to predict samples (left). The circuit implemented (right) multiplies the input (x_i) by the learned parameters (w_i) and adds b.

RF Classifier

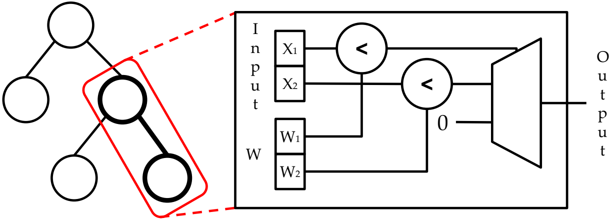


Figure: **Random Forest**. A single decision tree (from the ensemble) is shown (left) with the corresponding circuit of two nodes (right), with comparators and a MUX deciding the decision path.

MLP Classifier

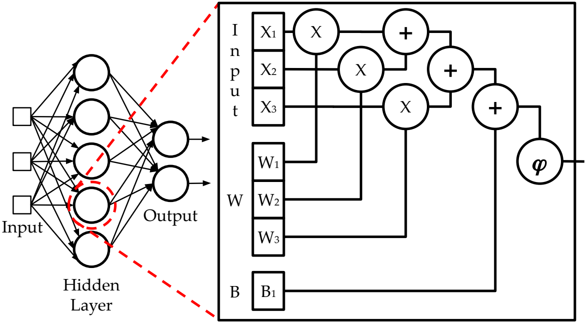


Figure: **MLP**. A feed-forward neural network composed by multiple neurons (left), which circuit implementation (right) is similar to SVM, but using an activation function to calculate the output.

Topics

1. Introduction
   - The Problem
   - Background
2. Malware REHAB
   - Architecture
3. Evaluation
   - Goals
   - Hardware Accelerator
   - Reconfigurable Hardware
   - Hardware Requirements
4. Final Remarks
   - Limitations
   - Conclusion
   - Questions?

REconfigurable, Hardware-Assisted Blocking (REHAB) of Malware

- FPGA-modelled.
- Mechanism notifying software-AV via detection interrupts.
- Runtime monitoring with no performance overhead.
- System-wide or Per-Process monitoring modes.
- Collects HPC data via memory-mapped registers.
- Profiling-based detection.
- Implement only the matching step of ML classifiers.
- Updatable via software (AV compatible).
- Updates the entire classifier and not only its weights.

REHAB Architecture

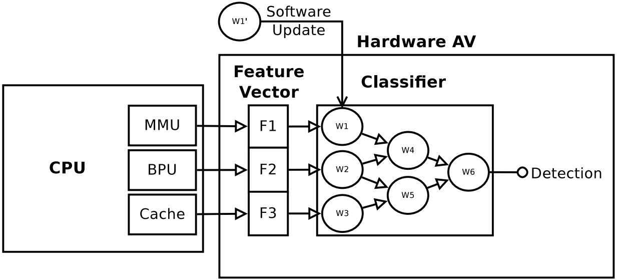


Figure: **REHAB Architecture**. CPU's HPC data is used as feature for a FPGA-based, reconfigurable ML classifier updatable via software.

REHAB Implementation

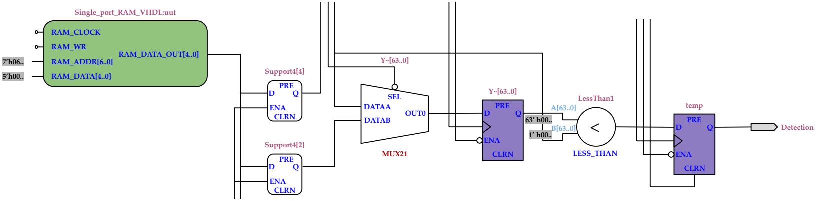


Figure: **Excerpt of a ML classifier implemented in FPGA**. ML parameters are loaded from an external memory at startup and can be updated by software writes to the external RAM memory.

Topics

1. Introduction
   - The Problem
   - Background
2. Malware REHAB
   - Architecture
3. Evaluation
   - Goals
   - Hardware Accelerator
   - Reconfigurable Hardware
   - Hardware Requirements
4. Final Remarks
   - Limitations
   - Conclusion
   - Questions?

We demonstrate:

- The need of a hardware accelerator.
- The need of reconfigurable hardware.
- Hardware requirements.

Topics

1. Introduction
   - The Problem
   - Background
2. Malware REHAB
   - Architecture
3. Evaluation
   - Goals
   - Hardware Accelerator
   - Reconfigurable Hardware
   - Hardware Requirements
4. Final Remarks
   - Limitations
   - Conclusion
   - Questions?

AV Checks Cost

Table: **Execution Speedup per AV check.** Hardware Accelerator is essential for overhead elimination.

ML algorithm →	**SVM**	**RF**	**MLP**
CPU	220μs	270μs	240μs
FPGA+Comm	124.5ns	111.2ns	158.9ns
Speedup	1.7k×	2.4k×	1.5k×

Topics

1. Introduction
   - The Problem
   - Background
2. Malware REHAB
   - Architecture
3. Evaluation
   - Goals
   - Hardware Accelerator
   - Reconfigurable Hardware
   - Hardware Requirements
4. Final Remarks
   - Limitations
   - Conclusion
   - Questions?

SVM

Table: SVM Classifier. 1000 iterations in a linear kernel results in the best accuracy for the VirusTotal dataset.

Kernel/Iter (#)	1000	10000	100000
`Poly`	0.2960	0.2960	0.2960
`Linear`	**0.8256**	0.7952	0.8088
`rbf`	0.4793	0.4793	0.4793

Table: SVM Classifier. 1000 iterations in a linear kernel results in the best accuracy for the VirusShare dataset.

Kernel/Iter (#)	1000	10000	100000
`Poly`	0.3644	0.4234	0.4234
`Linear`	**0.7705**	0.7353	0.7266
`rbf`	0.5001	0.4759	0.4759

MLP

Table: MLP Classifier. Alpha as 100 with adam solver results in the best accuracy for the VirusTotal dataset.

Solver/Alpha (#)	0.01	1	100	1000
sgd	0.4997	0.4997	0.4997	0.5003
adam	0.7098	0.7218	**0.7433**	0.7213
lbfgs	0.4997	0.4997	0.4997	0.4997

Table: MLP Classifier. Alpha as 1 with adam results in the best accuracy for the VirusShare dataset.

Solver/Alpha (#)	0.01	1	100	1000
sgd	0.4999	0.4999	0.4929	0.4999
adam	0.7288	**0.7614**	0.6951	0.7067
lbfgs	0.4999	0.4999	0.4999	0.4997

RF

Table: RF Classifier. 16 estimators and a max depth of 64 results in the best accuracy for the VirusTotal dataset.

Depth/Est (#)	8	16	32	64	128
4	0.9240	0.9172	0.9178	0.9214	0.9199
8	0.9366	0.9366	0.9398	0.9434	0.9403
16	0.9377	0.9455	0.9408	0.9445	0.9429
32	0.9350	0.9439	0.9403	0.9460	0.9445
64	0.9392	**0.9466**	0.9445	0.9434	0.9445

Table: RF Classifier. 16 estimators and a max depth of 16 results in the best accuracy for the VirusShare dataset.

Depth/Est (#)	8	16	32	64	128
4	0.9564	0.9569	0.9577	0.9601	0.958
8	0.9644	0.9642	0.9653	0.9644	0.9661
16	0.9626	**0.9671**	0.9655	0.9639	0.9671
32	0.9644	0.9642	0.965	0.9644	0.9661
64	0.962	0.965	0.9442	0.9653	0.9647

Concept Drift

Table: **Classifier's concept drift.** Whereas the MLP classifier best scored in the VirusTotal dataset, the RandomForest classifier was the best choice for the VirusShare dataset, thus showing the need of having reconfigurable AV mechanisms.

Classifier/Dataset	VirusShare	VirusTotal
Random Forest	**0.9144**	0.6953
MLP	0.881	**0.9738**
SVM	0.9079	0.5728

Topics

1. Introduction
   - The Problem
   - Background
2. Malware REHAB
   - Architecture
3. Evaluation
   - Goals
   - Hardware Accelerator
   - Reconfigurable Hardware
   - Hardware Requirements
4. Final Remarks
   - Limitations
   - Conclusion
   - Questions?

Table: **SVM Implementations.**

Classifier	Work	LUTs/REGs/MULs/DSPs
SVM	**This**	520/196/5/20
	[NF16]	832/–/–/–
	[MSKT]	748/–/–/–

Table: **RF Implementations.**

Classifier	Work	LUTs/REGs/MULs/DSPs
RF	**This**	707/40/0-7.5K/240/0/0
	[FBL09]	4k-24K/–/–/–
	[NJSS17]	600-118K/–/–/–

Table: **MLP Implementations.**

Classifier	Work	LUTs/REGs/MULs/DSPs
MLP	**This**	170/89/5-11K/690/502/38
	[FBL09]	6.7K/5K/–/–
	[eMEH14]	26.8K/4K/–/–

Topics

1. Introduction
   - The Problem
   - Background
2. Malware REHAB
   - Architecture
3. Evaluation
   - Goals
   - Hardware Accelerator
   - Reconfigurable Hardware
   - Hardware Requirements
4. Final Remarks
   - Limitations
   - Conclusion
   - Questions?

Limitations & Future Work

Limitations

- Proof-of-Concept (PoC) for future developments.
- Single Classifier.
- Current AV detection drawbacks.

Future Work

- Parallel Classifiers.
- In-Place Learning.
- Feedback Information for AV companies.

Topics

1. Introduction
   - The Problem
   - Background
2. Malware REHAB
   - Architecture
3. Evaluation
   - Goals
   - Hardware Accelerator
   - Reconfigurable Hardware
   - Hardware Requirements
4. Final Remarks
   - Limitations
   - Conclusion
   - Questions?

Conclusion

- Malware are very dynamic pieces of code.
- Malware classifiers present concept drift.
- Antivirus are performance-intensive applications.
- Reconfigurable Hardware as a promising alternative for efficient and effective malware detection.

Topics

1. Introduction
   - The Problem
   - Background
2. Malware REHAB
   - Architecture
3. Evaluation
   - Goals
   - Hardware Accelerator
   - Reconfigurable Hardware
   - Hardware Requirements
4. Final Remarks
   - Limitations
   - Conclusion
   - Questions?

Contact

mfbotacin@inf.ufpr.br

References

- Sami el Moukhlis, Abdessamad Elrharras, and Abdellatif Hamdoun, *Fpga implementation of artificial neural networks*, 2014.
- Antonyus Ferreira, Edna Barros, and Teresa Ludermir, *Fpga design of a mlp artificial neural network architecture*, `http://www.lbd.dcc.ufmg.br/colecoes/sforum/2009/0046.pdf`, 2009.
- D. Mahmoodi, A. Soleimani, H. Khosravi, and M. Taghizadeh, *Fpga simulation of linear and nonlinear support vector machine*.
- Daniel H Noronha and Marcelo Fernandes, *Implementação em fpga de máquina de vetores de suporte (svm) para classificação e regressão*, `http://www.lbd.dcc.ufmg.br/colecoes/eniac/2016/004.pdf`, 2016.
- H. Nakahara, A. Jinguji, S. Sato, and T. Sasao, *A random forest using a multi-valued decision diagram on an fpga*, ISMVL, 2017.

Circuit Growth

Table: **Decision Tree Growth.**

Depth	1	2	4	7	8	16	32	64
LUTs	63	114	370	570	707	1313	1982	2534

Table: **Adding Random Forest Trees.**

Trees (#)	1	2	3	4	8	16
LUTs	707	908	1132	1411	1708	7511

Table: **Adding MLP Layers.**

Perceptrons	1	2	4	8	16	64	128
LUTs	170	328	520	1446	2816	5196	11004