A hybrid framework to analyze Web and OS malware

Vitor Monte Afonso [1]
Dario Simões Fernandes Filho [1]
André Ricardo Abed Grégio [1 2]
Paulo Lício de Geus [1]
Mario Jino [1]
[1] University of Campinas (UNICAMP), Brazil
[2] Renato Archer IT Research Center (CTI), Brazil

JUNE 2012

Agenda

- Introduction
- OS malware analysis
- Web malware analysis
- Developed framework
- Tests and results
- Conclusion and future work

Agenda

- Introduction
- OS malware analysis
- Web malware analysis
- Developed framework
- Tests and results
- Conclusion and future work

Introduction [1]

- Infected systems
  - Steal information, send SPAM messages, attack other systems
- Web as the main infection vector
  - Web malware – malicious code inside Web pages
  - Client-side exploits – several components (flash, pdf, java, etc)
- User infection
  - Infection of benign pages
  - Phishing messages
- Analysis systems
  - Study and understand the threats
  - Existing systems have limitations

Introduction [2] – This work

- Combines analysis of Web and OS malware
- Better detection rate of Web malware than existing systems
- OS behavior monitor can operate in emulated, virtual or bare metal environments, allowing our framework to correctly analyze samples that detect virtual or emulated environments

Agenda

- Introduction
- OS malware analysis
- Web malware analysis
- Developed framework
- Tests and results
- Conclusion and future work

OS malware analysis

- Static analysis – packer problem
- Dynamic analysis – controlled environment
- Virtual Machine Introspection (VMI) - Anubis
  - Emulated or virtual system
  - Intermediary layer monitor actions
- System Service Dispatch Table (SSDT) hooking - Our
  - Modify kernel, redirecting native functions
  - No need for emulated/virtual env., but problem with rootkits
- API hooking - CWSandbox
  - Modify binary, redirecting system APIs
  - High-level information, but easy to detect

Agenda

- Introduction
- OS malware analysis
- Web malware analysis
- Developed framework
- Tests and results
- Conclusion and future work

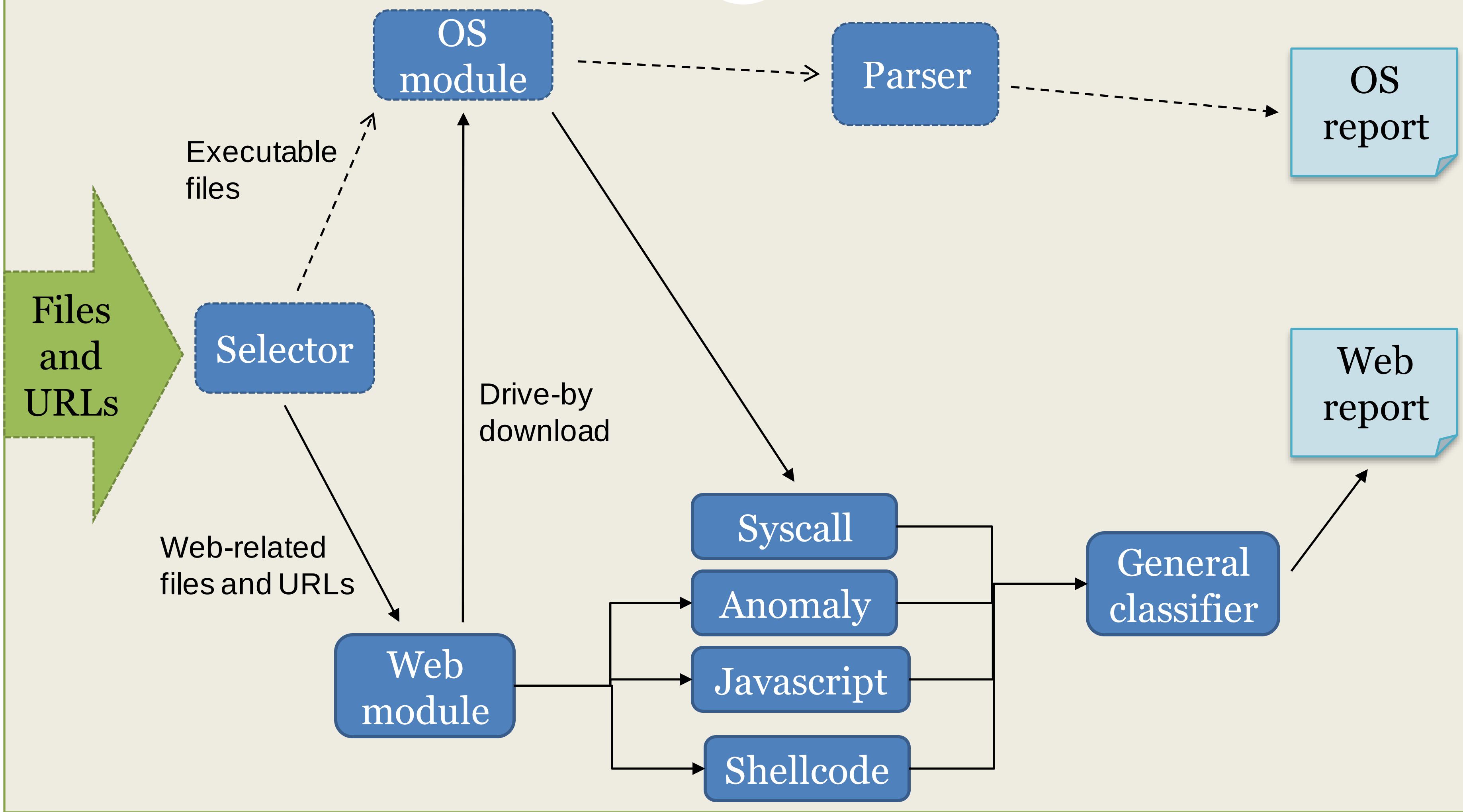
Web malware analysis

- Low-interaction honeyclient
  - Use browser emulator
  - + Executes faster
  - - Differences in page handling make it easy to detect
  - Ex: Jsand, PhoneyC
- High-interaction honeyclient
  - Use real browser
  - + Real browser behavior
  - - Delay to restore virtual/emulated environment
  - Ex: Capture-HPC, our Web module

Agenda

- Introduction
- OS malware analysis
- Web malware analysis
- Developed framework
- Tests and results
- Conclusion and future work

Developed Framework - Architecture



OS Malware Analysis

- Windows kernel driver implements SSDT hooking
- Monitor system calls
- First on emulated environment
  - Error or specific packer -> bare metal environment
- File, registry, mutex, process, driver, network and memory operations
- Parser
  - Selects relevant actions
  - Relevant -> modify system or leak information
  - Produces report

Web Malware Analysis

- Windows DLL hooks IE libraries
  - Extract Javascript behavior
- Binaries resulted from DBD -> OS module
- Anomaly detection
  - 8 features extracted from JS behavior
  - Weka framework, ThresholdSelection and Random forest
- Javascript signatures
  - Information stealing
- Shellcode detection (Libemu)
- System call signatures

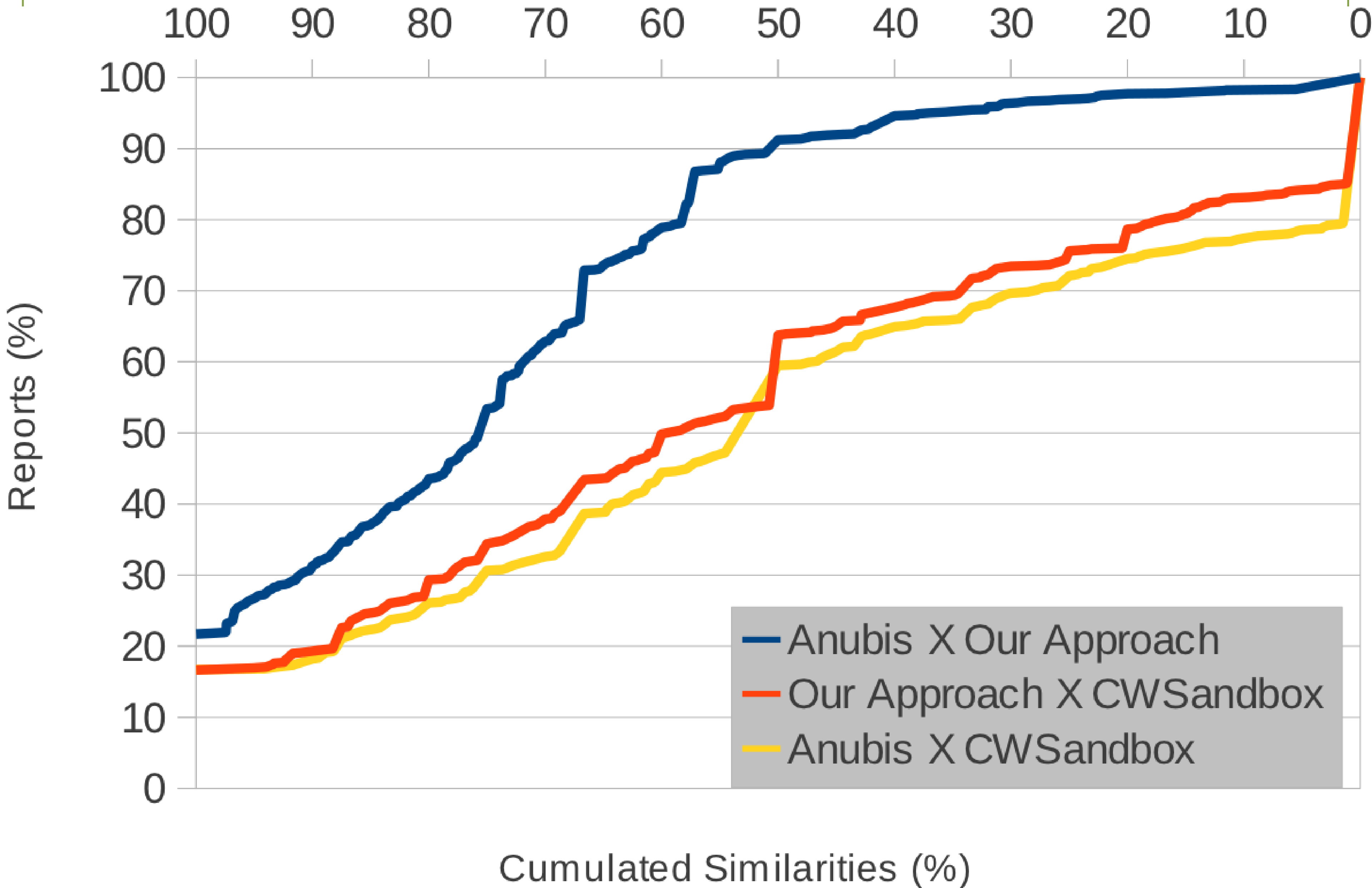
Agenda

- Introduction
- OS malware analysis
- Web malware analysis
- Developed framework
- Tests and results
- Conclusion and future work

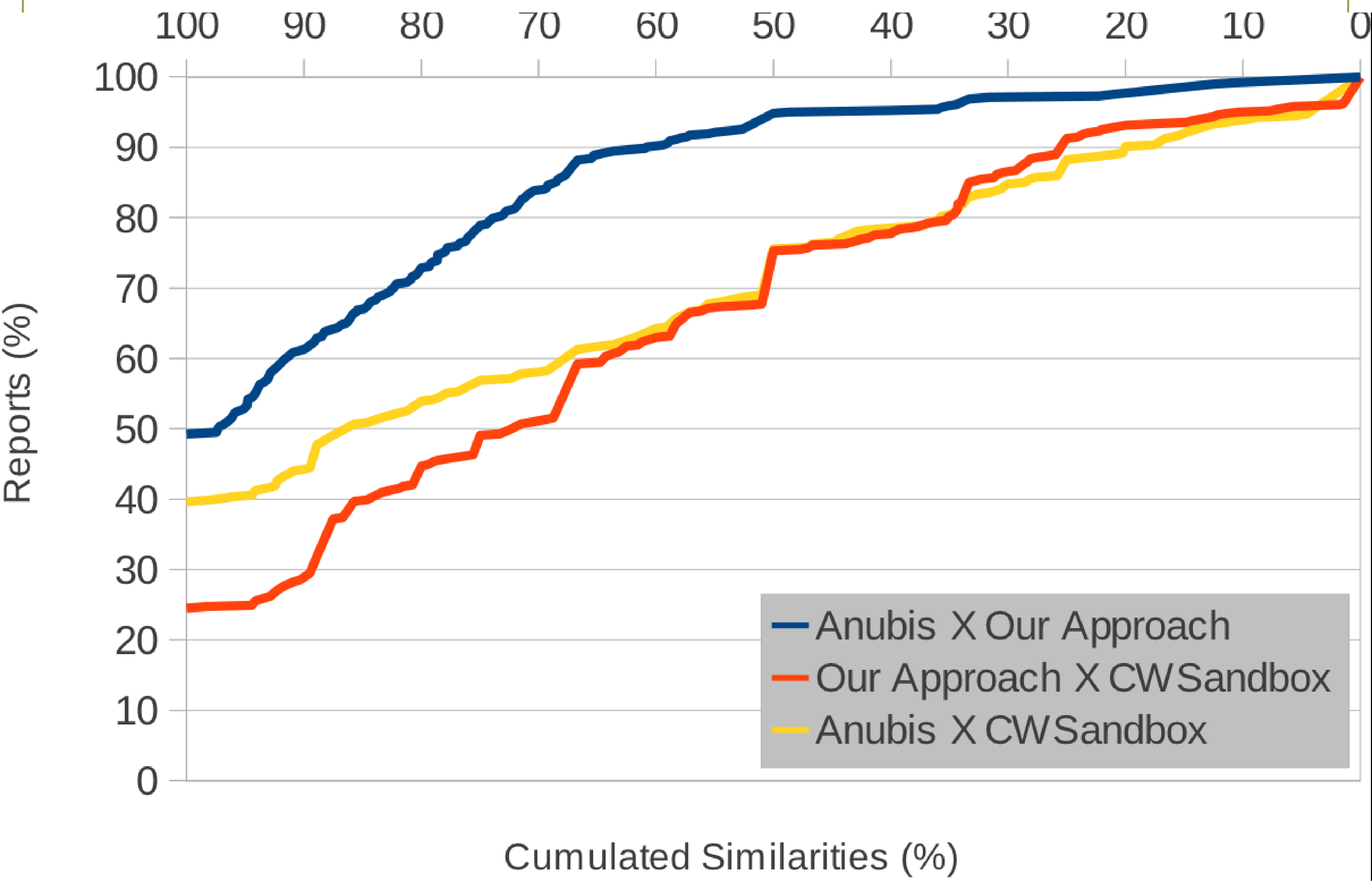
OS Malware Tests

- Test effectiveness of the monitoring component
- Compared to Anubis and CWSandbox
  - Referenced systems with public interface
- 1,744 samples obtained from collectors
  - Separated : have or don't have anti-emulation packer
- Normalization of reports
  - Removal of irrelevant actions
- Similarity between 2 reports
  - Percentage of the smaller contained in the other

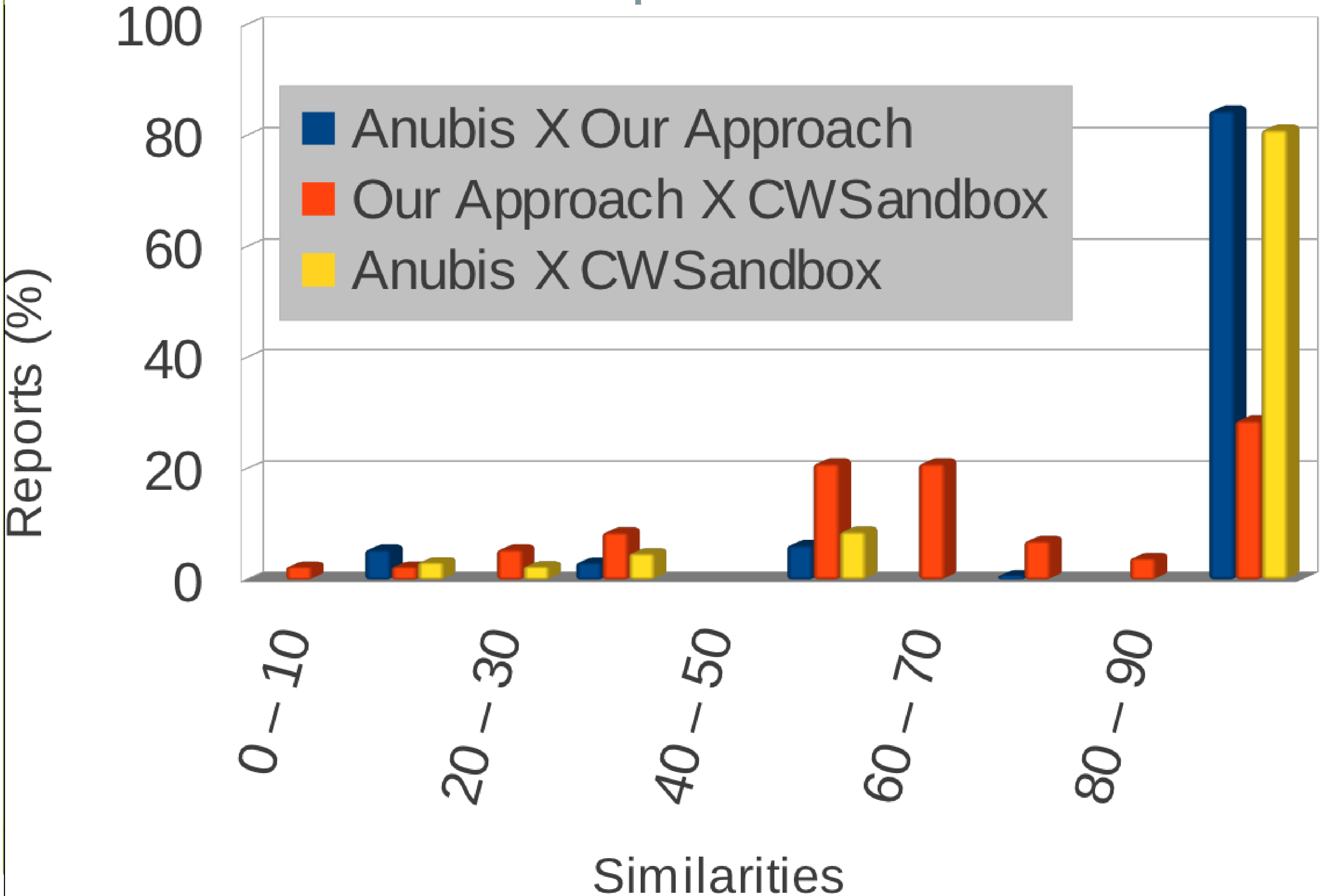
Without anti-emulation packer



With anti-emulation packer



Malware with packer tElock



Results

- For samples without tElock our report was very similar to Anubis
- For samples with tElock we could extract the behavior but Anubis did not
- We also noted that CWSandbox reports contained a lot of irrelevant information

Web Malware Tests

- Compare detection rate with other systems
- Jsand (Wepawet), PhoneyC and Capture-HPC
  - Referenced systems with available code or public interface
- 1,400 malicious and 6,781 benign samples
  - Malicious – Public lists of domains hosting Web malware and VxHeavens database
  - Benign - Alexa TOP sites (except those classified as malicious by Google safe browsing)
- Samples divided into training (anomaly detection) and test (comparison)

Results [1]

System	FP(%)	FN(%)	TP(%)	TN(%)
Our approach	0,4	23,4	76,6	99,6
Jsand	2,2	76,2	17,1	97,8
PhoneyC	0	88,7	11,3	100
Capture-HPC	0,2	94,3	5,8	99,8

- Samples with error – taken from the sum

Results [2]

- More than one characteristic to compare
- Harmonic mean – used in intrusion detection
- Uses precision and recall
  - Precision – samples classified as malicious that really are malicious
  - Recall – malicious samples correctly classified

System	Recall(%)	Precision(%)	Harm. mean (%)
Our approach	76,6	99,4	86,6
JSand	18,3	88,5	30,3
PhoneyC	11,3	100	20,3
Capture-HPC	5,8	96,6	10,9

Agenda

- Introduction
- Common client-side attacks
- Analysis systems
- Developed system
- Tests and results
- Conclusion and future work

Conclusion and Future Work

- Conclusion
  - OS module can capture more information when analyzing malware packed with “tElock” and produces more objective reports
  - Web module showed better detection rates (more than 2x the second place) than the existing systems
- Future work
  - Extend the anomaly detection (best results) of the Web module to other script languages
  - Extend the OS module to better analyze rootkits

The End

- Thank you
- Questions ?

Email: vitor@las.ic.unicamp.br