Systems Security at VU University Amsterdam

Herbert Bos, Lorenzo Cavallaro, and Andrew S. Tanenbaum
Systems & Security Group
Department of Computer Science
Vrije Universiteit Amsterdam

***Abstract*—The systems and network security, and the secure and reliable systems groups carry out research in computer and network dependability at the Vrije Universiteit Amsterdam. The former group, led by Prof. Herbert Bos, has a strong and historical background in attack detection, dynamic analysis, and reverse engineering of software. The secure and reliable systems group, led by Prof. Andrew S. Tanenbaum, is instead traditionally rooted on studying and guaranteeing dependability properties of systems. The two groups interact in a natural way and blend their knowledge and expertise to contribute on the security of networks and systems.**

I. Scope of research

With three full professors, four associate professors, and two assistant professors, the Computer Systems section at the Vrije Universiteit Amsterdam (VU University) is one of the largest in the Netherlands. Moreover, the last research assessment of Dutch universities in Computer Science over the period 2002-2008 (published in 2010), gave the Computer Systems section of the VU a top ranking (with maximum scores on all evaluation criteria).

The broad area of security at systems level features prominently in the section's research interests. At VU University, the area is covered by two separate, but closely collaborating groups:

1) a group headed by Herbert Bos that works on Systems and Network Security;
2) a group headed by Andrew S. Tanenbaum that works on Secure and Reliable Systems.

The focus on Tanenbaum's group leans towards reliability, but also covers aspects of security, as witnessed by security researchers like Lorenzo Cavallaro and Bruno Crispo that make up the group. The focus in Bos' group is mostly on system-level security both at the host and in the network. Both groups work primarily in low-level systems security. So, while the groups do much at the VM, OS and assembly level, there is less work on, say, Web security.

In the remainder of this document, we describe both research thrusts in more detail. Examples of research topics in the two groups are shown in Figure 1.

II. Research in the Systems and Network Security Group

The Systems and Network Security Group is very active in the area of attack detection, dynamic analysis, and reverse engineering.

Reverse engineering and security of legacy binaries: In terms of resources, the main research area in this group is that of reverse engineering of complex software. Funded by an ERC Starting Grant as well as several smaller grants, the group aims to tackle a fundamental problem in computer science: to revert low-level assembly to high-level source code. To do so, we use both static and dynamic analysis.

Most of the commercial software industry assumes that compilation (the translation of source code to binary code), is irreversible in practice for real applications. The research question for our group is whether this irreversibility assumption is reasonable. Specifically, we aim to demonstrate that the assumption is false.

The application domains for this research direction are two-fold. First, we want to be able to analyse what software is doing. For instance, if we buy a program, we would like to verify that it does what it is advertised to do (and not what we would not like it to do). Second, we would like to fix bugs in binary software. Specifically, we aim to protect legacy binaries from memory corruption attacks.

In addition, the reverse engineering techniques that we develop will be interesting for malware analysis. While malware may use obfuscation techniques, it is difficult (or at least expensive) to hide the use of certain data structures. In general, code obfuscation will be an important research area in our group. Specifically, we will work on detection, circumvention and improvement of code and data obfuscation techniques.

Dynamic analysis: The research team developed a variety of taint analysis solutions, of which the Argos full system emulator is probably best known. Argos is used by many organisations in many projects, mostly as a honeypot or malware analysis engine. In general, we work on a variety of high-interaction honeypot solutions—both for the client-side and for server applications. In other projects, taint tracking also features prominently. For instance, we work on solutions for attack detection, (decoupled) protection of mobile devices, intrusion recovery, that all depend on dynamic taint analysis. In addition, we have developed our own, very fast, dynamic binary translator with support for taint tracking. We expect to build on this

Attack detection and analysis: Systems-level attack detection and analysis permeates the research group. We

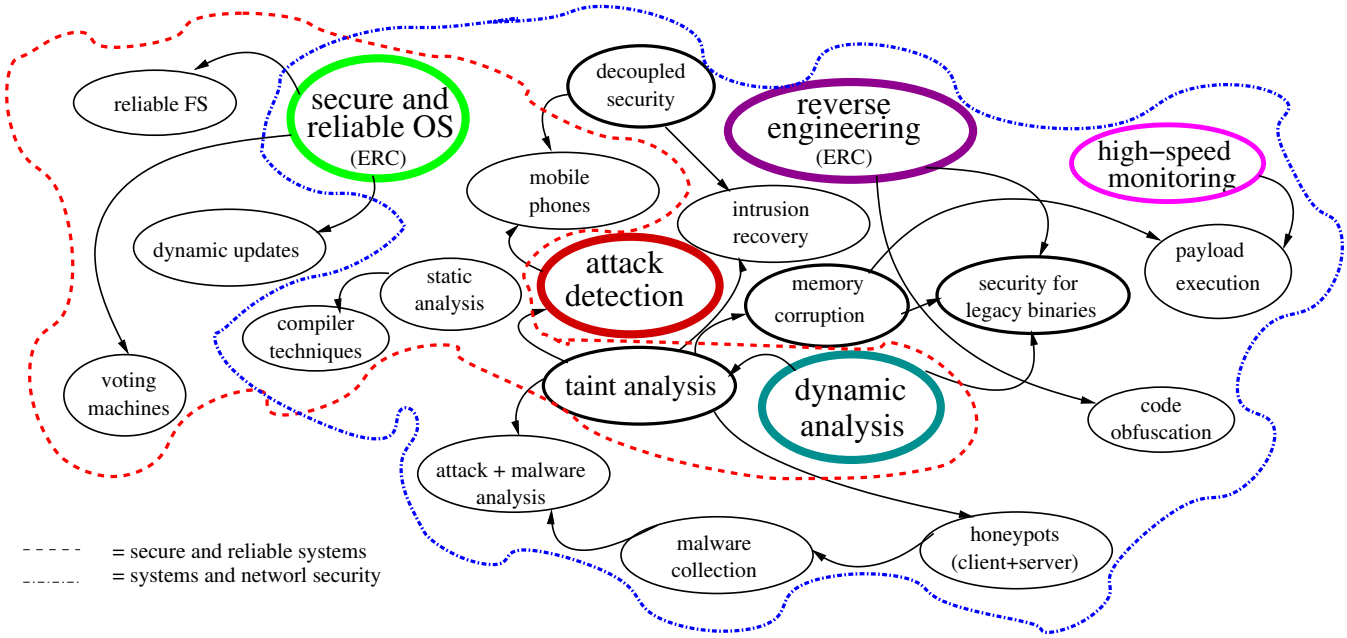

Figure 1. Examples of research topics in the system security groups at VU University

already mentioned that we work on techniques to detect attacks by means of taint tracking. However, we also work on efficient solutions to detect attacks in the network. Part of our effort here is in building network monitoring tools, another part in abstract payload execution. Also, we collect, run and analyse large traces of malware. Finally, we conduct research in detecting attacks in constrained environments, such as mobile phones and other ultraportables.

III. Research in the Secure and Reliable Systems Group

The research interests of the Secure and Reliable Systems Group broadly fall in the area of software dependability. Supported by an ERC Advanced Grant, the research team currently places particular emphasis on making operating systems (OSes) more reliable and secure.

Monolithic operating systems (OSes) are in fact complex pieces of software that usually offer very little reliability and security guarantees. Faulty user-space applications can generally be restarted without affecting the existing concurrent communications but those involving the faulty processes. On the other hand, in a monolithic OS design, the kernel and all its components share a common address space and any component can potentially invoke arbitrary kernel functions. In this scenario, it becomes extremely complicated–if not impossible–to isolate and restart faulty kernel components as it is generally hard to define their boundaries and interactions (e.g., what kernel control paths are executed and how information are shared). Unfortunately, run-time bugs are not the only security threats an OS must deal with. For instance, malicious components may undermine the security of the whole system from its root: kernel rootkits can be installed on the system to replace or modify the legitimate behavior of arbitrary subsystems of the OS to fulfill criminals will.

To withstand such threats, the group is exploring approaches to combine a careful OS design with automated compiler-based instrumentation techniques. This spins off a number of interesting research directions. For instance, combining a modular operating system design with compiler-based techniques enables low-overhead runtime address space randomization (ASR) and fine-grained live update support for arbitrary OS components. This ultimately shows that is possible to build a polymorphic OS that constantly re-randomizes itself, while keeping the overhead to a minimum. Similarly, the same carefully planned OS design does not expose any recovery infrastructure to the OS programmer and drastically reduces the complexity of the problem space considered. This allows effective crash recovery using automatic instrumentation in a nonintrusive way, achieving transparent and component-agnostic recovery from crashes occurring anywhere during the execution of the component's task.

Automated compiler-based instrumentation techniques on their own enable a number of other interesting research directions. For instance, program analysis can be leveraged to identify interesting programs invariants that can be made available at runtime. Such properties can then be asynchronously and dynamically checked by idle cores, improving the dependability requirements of the whole system, while keeping the runtime overhead to a minimum. Moreover, compiler-based instrumentation enables to

experiment with generic fine-grained program transformations, e.g., taint-tracking, obfuscation, and address space randomization, which improve the overall security of the transformed application.

IV. Overlap

As indicated by Figure 1, the two research groups collaborate closely on a variety of projects. The heads of the groups co-supervise Ph.D. students in the design of secure and reliable operating systems, while members of the groups also work together on several projects.

V. Research environment

The Computer Systems section at VU University is one of the largest Systems departments in the country with 3 full professors and a strong reputation in operating systems and security (Minix, CVS, Argos, and Amoeba all started here). The work on Minix sparked the development of Linux. In recent years, prof. Tanenbaum was awarded a prestigious Dutch Royal Society of Science (KNAW) Professorship, and an equally prestigious ERC Advanced Grant. Similarly, there was an ERC Starting Grant for prof. Herbert Bos, bringing the current total to two—more than any other CS department in the Netherlands. Two of the full professors are among the top 10 most cited computer scientists in The Netherlands (at nr. 1 and nr. 9, respectively), indicating the excellence of the research environment.

Many of the section's former Ph.D. students rank among the top researchers in the world. Examples include Frans Kaashoek (MIT), Robert van Renesse (Cornell), Leendert van Doorn (AMD), Sape Mullender (Bell Labs), and Werner Vogels (Amazon). One important quantitative measure of academic reputation is citation impact. The Report on Science and Technology Indicators issued by the Netherlands Observatory for Science and Technology (NOWT, 2008), shows that the CS Department at the VU ranks highest of all CS departments in the Netherlands in impact score. The high score is corroborated by other, independent studies. The last assessment of research of Dutch universities in Computer Science in over the period 2002-2008 (published in 2010), gave the Computer Systems section of the VU the top ranking (with a maximum score on all evaluation criteria).

VI. Outlook and future research directions

The research directions currently pursued by the system security groups are relatively stable. Both groups are partially funded by ERC grants that provide clout to the research efforts. The group of Secure and Reliable systems has a strong tradition of incorporating the fruits of the research in a working system, centered on Minix-3. Unlike most other operating systems the design of Minix-3 is centered first on reliability, and second on performance. Meanwhile, a lot of the effort in the Systems and Network Security group in the next five years will be invested in reverse engineering—with the goal of increasing the security of systems. At the same time, attack detection, prevention, and analysis remain very much on our radar too.

References

[1] Jorrit N. Herder Andrew S. Tanenbaum and Herbert Bos. Can We Make Operating Systems Reliable and Secure? *IEEE Computer, ISSN 0018-9162*, 39(4):44–51, May 2006.

[2] Herbert Bos, Willem de Bruijn, Mihai Cristea, Trung Nguyen, and Georgios Portokalidis. FFPF: Fairly Fast Packet Filters. In *Proceedings of OSDI'04*, San Francisco, CA, December 2004.

[3] Herbert Bos and Kaiming Huang. Towards software-based signature detection for intrusion prevention on the network card. In *Proceedings of Eighth International Symposium on Recent Advances in Intrusion Detection (RAID2005)*, Seattle, WA, September 2005.

[4] Herbert Bos and Bart Samwel. Safe kernel programming in the OKE. In *Proceedings of the Fifth IEEE Conference on Open Architectures and Network Programming (OPENARCH'02)*, pages 141–152, New York, USA, June 2002.

[5] Willem de Bruijn, Herbert Bos, and Henri Bal. Application-tailored I/O with Streamline. *ACM Transactions on Computer Systems (TOCS)*, 2011.

[6] Willem de Bruijn, Asia Slowinska, Kees van Reeuwijk, Tomas Hruby, Li Xu, and Herbert Bos. SafeCard: a Gigabit IPS on the network card. In *Proceedings of 9th International Symposium on Recent Advances in Intrusion Detection (RAID'06)*, Hamburg, Germany, September 2006.

[7] Cristiano Giuffrida, Lorenzo Cavallaro, and Andrew S. Tanenbaum. We Crashed, Now What? In *Proceedings of the 6th Workshop on Hot Topics in System Dependability (HotDep'10)*, Oct 2010.

[8] Lorenzo Martignoni, Aristide Fattori, Roberto Paleari, and Lorenzo Cavallaro. Live and Trustworthy Forensic Analysis of Commodity Production Systems. In *13th International Symposium on Recent Advances in Intrusion Detection (RAID)*, 2010.

[9] Srijith Nair. *Remote Policy Enforcement Using Java Virtual Machine*. PhD thesis, VU University Amsterdam, 2010.

[10] N. Paul and A. S. Tanenbaum. Trustworthy Voting: From Machine to System. *IEEE Computer*, pages 23–29, May 2009.

[11] Georgios Portokalidis and Herbert Bos. Eudaemon: Involuntary and On-Demand Emulation Against Zero-Day Exploits. In *Proceedings of ACM SIGOPS EUROSYS'08*, pages 287–299, Glasgow, Scotland, UK, April 2008. ACM SIGOPS.

[12] Georgios Portokalidis, Philip Homburg, Kostas Anagnostakis, and Herbert Bos. Paranoid Android: Versatile Protection For Smartphones. In *Proceedings of the 26th Annual Computer Security Applications Conference (ACSAC)*, Austin, Texas, December 2010.

[13] Georgios Portokalidis, Asia Slowinska, and Herbert Bos. Argos: an Emulator for Fingerprinting Zero-Day Attacks. In *Proc. ACM SIGOPS EUROSYS'2006*, Leuven, Belgium, April 2006.

[14] Christian Rossow, Christian J. Dietrich, Herbert Bos, Lorenzo Cavallaro, Marteen van Steen, Felix C. Freiling, and Norbert Pohlmann. Sandnet: Network Traffic Analysis of Malicious Software. In *1st Workshop on Building Analysis Datasets and Gathering Experience Returns for Security (BADGERS)*, April 2011.

[15] Asia Slowinska and Herbert Bos. The age of data: pinpointing guilty bytes in polymorphic buffer overflows on heap or stack. In *23rd Annual Computer Security Applications Conference (ACSAC'07)*, Miami, FLA, December 2007.

[16] Asia Slowinska and Herbert Bos. Pointless Tainting? Evaluating the Practicality of Pointer Tainting . In *Proceedings of ACM SIGOPS EUROSYS*, Nuremberg, Germany, March-April 2009.

[17] Asia Slowinska, Traian Stancescu, and Herbert Bos. Howard: a dynamic excavator for reverse engineering data structures. In *Proceedings of NDSS 2011*, San Diego, CA, 2011.

[18] Cristiano Giuffrida Stefano Ortolani and Bruno Crispo. Bait your Hook: a Novel Detection Technique for Keyloggers. In *Proceedings of the 13th International Symposium on Recent Advances in Intrusion Detection*, pages 200–217, 2010.

[19] Yves Younan, Pieter Philippaerts, Lorenzo Cavallaro, R. Sekar, Frank Piessens, and Wouter Joosen. PAriCheck: an Efficient Pointer Arithmetic Checker for C Programs. In *Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security*, pages 145–156, Beijing, China, 2010. ACM.