Editor: **Fei-Yue Wang,** Chinese Academy of Sciences, feiyue@ieee.org

Toward Digital Asset Protection

Christian Collberg, *University of Arizona*
Jack Davidson, *University of Virginia*
Roberto Giacobazzi, *University of Verona, Italy*
Yuan Xiang Gu, *Irdeto, Canada*
Amir Herzberg, *Bar-Ilan University, Israel*
Fei-Yue Wang, *Chinese Academy of Sciences*

Man-at-the-end (MATE) attacks are an understudied branch of computer security. These attacks involve an adversary gaining an advantage by violating software or hardware under their control, directly or via a remote connection. On an individual scale, MATE attacks could violate the privacy and integrity of medical records and other sensitive personal data, and on a larger scale, they could cripple a national infrastructure (such as a power grid and the Internet itself). The goal of software protection (SP) research is to make software safe from such MATE attacks by preventing adversaries from tampering, reverse engineering, and illegally redistributing software.

In July 2011, the Digital Asset Protection Association (DAPA) was launched to address the challenges specific to MATE attacks and SP research in general. As DAPA activities and efforts get underway, the ultimate goal is to establish standards and baseline definitions for SP research and to promote coordinated, open efforts among academia and industry.

Threats to Digital Assets

In everyday life, whether for business or personal use, untrusted environments are ubiquitious—from home networks to consumer devices, the public Internet, the cloud, and the Internet of Things—where traditional computer and network security are inadequate to address MATE attacks.

Techniques for software protection originated with the need to protect license-checking code in software, particularly in games. Sophisticated techniques, such as white-box cryptography, were developed to prevent adversaries from circumventing media antipiracy protection in Digital Rights Management (DRM) systems. However, it has become increasingly clear that software protection is no longer just needed for intellectual property protection in the entertainment world; it is now also necessary in more sensitive scenarios, including protecting individuals' privacy and protecting the integrity of our national infrastructure.

In this context, insider attacks are a particularly insidious form of MATE attack. Here, the attacker is a trusted individual within an organization authorized to perform certain actions. For example, a system administrator might be authorized to perform system upgrades, add users, and run backups but is not allowed to modify database entries or read individual users' email. Malicious insiders will use their credentials to perform unauthorized actions. For example, a disgruntled system administrator could modify a backup script (something he or she is authorized to do) to include a logic bomb that, were he or she to get fired, would destroy important user files. Such insiders are partially trusted and might have the ability to cover up their tracks (for example, by modifying log files), which makes such attacks particularly difficult to counter.

Because of the powerful attacks an adversary can launch in a MATE scenario, we typically do not expect any SP technique to fend off an attack indefinitely. Rather, we use techniques such as diversity (generating multiple unique instances of the same program), renewability (frequently updating our defenses), and defense-in-depth (layering multiple levels of defenses) to construct defense perimeters that will last for as long as possible. SP primitives are categorized as tamperproofing

1541-1672/11/$26.00 © 2011 IEEE
Published by the IEEE Computer Society

(protecting the integrity of a piece of software, for example, by preventing or detecting unauthorized modification of code), watermarking (ensuring that unauthorized copies can be traced by embedding unique identifiers in a piece of software), obfuscation (protecting the intellectual property in software by making it difficult to reverse engineer), and birthmarking (detecting illegal reuse of software by evaluating the similarity of two pieces of code).

With the realization that MATE attacks present a real threat both to important aspects of our lives as individuals and to the security of commercial and government institutions, we must become better at building, deploying, and evaluating SP techniques. Developing effective procedures for software protection algorithm evaluation, in particular, has been a serious concern for the SP community for some time. Additionally, the community is still lacking a universally accepted set of attack models, without which it becomes difficult to develop and evaluate novel defense strategies. Negative results on the impossibility of perfect and universal obscurity[1] did not dishearten researchers from developing methods and algorithms for hiding sensitive information in programs. As well as Rice's theorem represented the greatest challenge for the development of automatic program analysis and verification tools in the last 30 years, the impossibility of obfuscation against malicious attacks is a major challenge for developing robust, concrete techniques that sufficiently delay attacker attempts to defeat them.

SP attacks and defenses are difficult to analyze, model, and evaluate for three fundamental reasons. First, the attacker is human and thus utilizes creativity, motivation, and ingenuity in circumventing SP defenses. Second, the attacker has unlimited access to the attack target. Third, all defenses will only stand up to a determined attacker for some certain period of time. Therefore, any successful model of attacks and defenses must account for the capabilities of a human attacker, and the result of an evaluation procedure must express the length of time that a particular defensive algorithm will stand up against such an attacker.

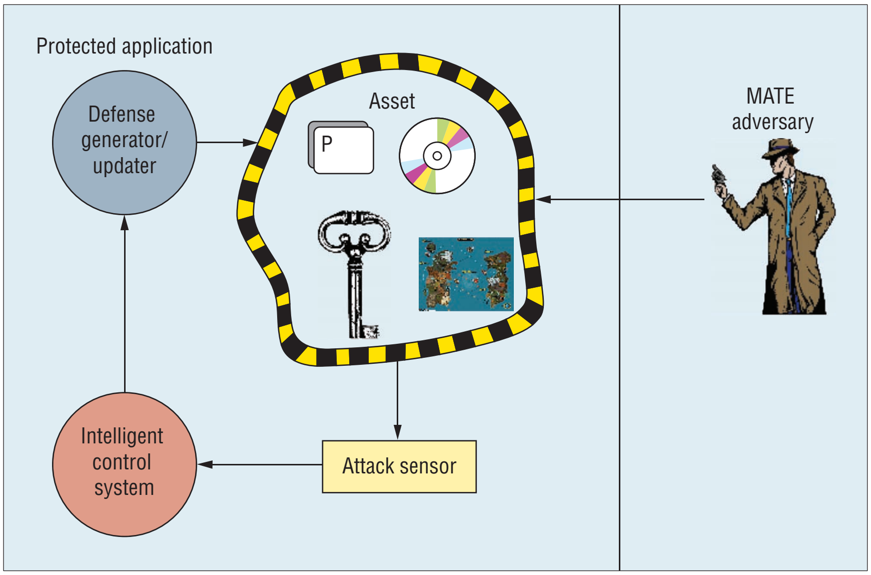

Figure 1. Modeling dynamic software protection algorithms. The protected application contains an asset under attack from a man-at-the-end (MATE) adversary. Sensors monitor the asset so an intelligent control system can determine whether an attack is underway.

Formal methods on their own are powerful tools for modeling automatic systems that can be used by a human attacker, but they are inadequate for providing a comprehensive model of a human-based MATE attack. Such a model is therefore naturally interdisciplinary, involving many areas of computer science. In particular, intelligent systems, in conjunction with software design, development and evaluation technologies, formal methods for program analysis and understanding, and program synthesis, will play a key role in building models of human malicious reverse-engineering abilities.

As an example, consider how intelligent systems can be used to model dynamic SP algorithms (see Figure 1). These algorithms continuously generate or update MATE defenses at runtime. In the figure, the protected application contains an asset (such as a proprietary algorithm, digital media, cryptographic keys, or other data resources) under attack from a MATE adversary. At runtime, sensors monitor the health of the application, the assets, and the defense perimeter. An intelligent control system reads the sensor and, if it determines that an attack is underway, instructs the defense generator/updater to modify or strengthen the defense perimeter.

Software Protection Evaluation Procedures

In July 2011, the First Software, Security, and Protection Workshop (http://www.ieeeisi.org/ssp.html) was held in Beijing, in conjunction with the IEEE International Intelligence and Security Informatics Conference. The workshop's goal was to bring together researchers and practitioners to make inroads into what has, arguably, been the main stumbling block for significant progress in the field: the lack of universally accepted evaluation standards and benchmarks that would allow uniform comparison of different protection algorithms.

A successful evaluation procedure will be able to determine mechanistically the effectiveness of a SP algorithm. Specifically, the evaluation should yield a set of properties about the algorithm, such as how it handles the trade-off between protection and performance, how much information it leaks (stealth), and how difficult it is for an adversary to disable (resilience). An important first step toward evaluation procedures must be the development of a set of benchmark programs, similar to those found in other fields (such as Standard Performance Evaluation Corporation [SPEC] benchmarks for system software performance evaluation or Transaction Processing Performance Council [TPC] for database evaluation).

Both synthetic and real benchmarks will be needed. Synthetic benchmarks should be small, should contain a clearly identified asset that needs to be protected, and should allow qualitative analysis of the nature of the protection to be added to the code. Real benchmarks should be open source, should implement an application similar to common uses of software protection (DRM, computer games, SCADA systems for infrastructure protection, and so on), and be close enough to real-world implementations to allow meaningful quantitative analysis of an algorithm, in particular its protection and performance trade-off.

Because evaluation procedures must be designed with respect to some class of attacks, the workshop discussions also centered on how to develop universally accepted attack models for SP research. Such models must take into account the power of reverse-engineering tools and the strategies commonly employed by adversaries. From a practical perspective, an understanding of the power of currently available tools is important when we implement particular protection algorithms because it lets us select a particular trade-off between security and performance. From a research perspective, however, we need to formalize the notion of a reverse-engineering tool and abstract the capabilities and limitations of such tools.

Attack models must further define what it means for an attack to be successful. Success in a SP scenario is much harder to define than in, say, cryptography or network security.

Can we learn from cryptography and try to emulate this success in the area of software protection?

For example, consider an attacker who wants to discover the important algorithms contained in our application. A first step might be trying to reconstitute as much of the source as possible, by disassembling and/or decompiling the binary executable. The situation is more complicated when the executable has been obfuscated to prevent disassembly. But attackers might not need to recover all the code to accomplish their goals; only recovering the parts that relate to important algorithms might suffice. Or, maybe all they need is a dynamic disassembly that gives accurate instructions for an executed path through the program, rather than a static one of the complete program. The attackers might be so accomplished that they are comfortable reading raw machine code and do not need to disassemble or decompile it at all. Given these ambiguities, how do we define a successful attack or defense?

SP Primitives and Criteria: Lessons from Crypto?

Defining security requirements is challenging because attackers will often try to exploit subtle differences between the real requirements and our definitions and benchmarks. Furthermore, it can be challenging to define and measure the attacker's capabilities as well as exact criteria for successful attacks. In fact, practitioners do not always use consistent, well-defined terminology to describe the functionality of security mechanisms.

For example, consider commonly mentioned, basic SP goals such as tamper resistance and obfuscation. What is really required here? Both requirements are often mentioned as "defense against reverse engineering," but what does this really mean? And in fact, is there a single meaning, or are there multiple useful notions of tamper resistance and obfuscation? Can we create precise definitions that will also be meaningful in practice?

Cryptography, a key security discipline, was studied and practiced for many years before precise definitions were adopted. Well-defined requirements were not published until in the late 1970s and early 1980s. However, once well-defined requirements were introduced, a dramatic increase in cryptography research followed, with a rapid flow of innovation—beautiful, insightful theoretical results—as well as useful and practical designs and systems. Can we learn from cryptography and try to emulate this success in the area of SP? Can we use the same style of definitions? Are some cryptographic definitions relevant to SP?

One lesson we can take from cryptography is the importance of identifying and focusing on a few basic building blocks. Indeed, although there are numerous definitions of different cryptographic schemes, there is a vast amount of research investigating relationships between these definitions and, mainly, proving reductions between schemes. In fact, most works defining a new type of scheme include a proof of reduction, showing how the new type of scheme can be securely implemented using well-known basic building blocks. There is a relatively small number of such basic building blocks, such as block ciphers (such as the Advanced Encryption Standard and Data Encryption Standard), collision-resistant hash functions and one-way functions (such as the Secure Hash Algorithm family of functions), and trapdoor permutations (such as the RSA [Rivest, Shamir, and Adleman] algorithm). Most other schemes have provably secure implementations using these basic blocks, including such important schemes as shared- and public-key encryption and digital signatures. Thus, the SP community must establish the security of the few basic building blocks and use them to build new complex, provably secure mechanisms.

Another lesson is the importance of clean, simple definitions of security and of adversarial resources and capabilities. In particular, one of the underlying principles of cryptography is Kerckhoffs' principle, established in 1883 by the Belgian military cryptographer Auguste Kerckhoff: "A cryptosystem should be secure even if everything about the system, except the key, is public knowledge."[2] Kerckhoff's principle stands in contrast to the approach of "security through obscurity," which is still the norm in the field of SP. The many advantages of designs following Kerckhoffs' principle are well known, but the question is, can we find reasonable SP systems that do not depend on obscurity?

Digital Asset Protection Association

To address these challenges, the Digital Asset Protection Association (DAPA) was launched during the First SSP Workshop. DAPA is a nonprofit organization with the goal of improving the state of the art in the SP field. Our hope is that DAPA will serve a role for SP similar to the role SPEC plays for computer systems and TPC plays for development in the database field.

The goal of the Digital Asset Protection Association is to bring together vendors, academics, software developers, and government agencies.

In particular, our goal is for DAPA to bring together vendors, academics, software developers, and government agencies interested in SP, attacking the main challenges currently hampering development in the field by

- building community standards for evaluating the effectiveness of SP algorithms;
- galvanizing the research community around SP research and practice, overcoming the lack of funding sources and promoting SP as a legitimate branch of computer security; and
- providing high-quality publication venues dedicated to disseminating results in SP research.

Thus, important initial tasks for DAPA will be to

- create awareness of the seriousness of MATE attacks among politicians, government agencies, software developers, and the general public;
- galvanize the scientific community around the main open problems and issues in SP in a MATE attack context;
- bridge communities, such as cryptography, programming languages, intelligent systems, and software engineering, to cooperate in the SP field;
- set community standards for algorithm evaluation;
- lobby funding agencies (such as the US National Science Foundation, European Science Foundation, and DARPA) to create programs in support of SP research and development;
- develop curricula and educational resources in support of undergraduate and graduate-level SP courses; and
- organize challenge competitions, workshops, and conferences and publish journals.

History

DAPA is the result of a long-standing activity that involved academics, industry, and practitioners. Since the pioneering research on code protection by obfuscation, diversification, and watermarking in the early 1990s, there has been a growing interest in this subject. In the industrial community, SP is viewed as a critical asset, often kept secret or for internal use only. This view has impeded the dissemination of information and expertise. In recognition of this problem, Cloakware, a leading producer of SP technology, initiated fruitful cooperation with academics in 2005, opening the field for a concrete involvement of industry and academia in joint research projects and setting up the basis for a larger community in SP, including researchers and end users.

In academia, the interest in SP has been the meeting point for various communities, from cryptography to programming languages, from hardware design to formal methods. The academic community working in SP is therefore naturally scattered, with a diversified background, but it is active in most countries. Activities include theoretical approaches as well as practical tools and algorithms, where various heterogeneous methods and approaches share a common goal and often common principles.

The need to devise a common infrastructure including widespread activities and goals was clear, leading a group of researchers from the Universities of Arizona, Virginia, and Verona; CAS; and Cloakware (now Irdeto) in 2010 to organize the first training program for PhD students and practitioners devoted to spreading the philosophy and technology behind SP. The successful organization of the first International Summer School on Information Security and Protection in Beijing in July 2010 (http://isisp10.scienze.univr.it/Welcome.html) was doubled in Ghent, Belgium, in 2011 (http://issisp.elis.ugent.be/?file=kop1.php).

Tasks

Building an academic and industrial organization such as DAPA from scratch will take time. Over the next five years, we foresee the following activities.

Certification. DAPA will strive to become a reference for certifying technology in digital-asset protection. This task involves a number of different activities including defining and maintaining benchmarks for evaluating technologies, providing metrics and formal methods for evaluating protection mechanisms, building and maintaining a network of experts in code protection, building one or more red teams whose main goal is to validate technological solutions with respect to the state of the art in tools and methodologies of attack, and providing continuous feedback for benchmark and metric evaluation.

DAPA seeks the development of secure, dependable, and trusted digital-asset infrastructures.

Support. DAPA will build and maintain the necessary infrastructure for fundraising, promoting access to DAPA-developed content (schools, workshops, seminars, and conferences) to young researchers and practitioners interested in pursuing SP research. Furthermore, DAPA will provide the necessary information, literature, tutoring support, and career opportunities for anyone interested in entering the digital-asset protection field.

Dissemination. DAPA will maintain a website and will be present in most social networks, bringing together those interested in the field. DAPA will organize summer schools, conferences, and workshops covering technical, legal, and social aspects connected with SP and DRM. Connected with these activities, we will plan industrial days to disseminate the DAPA philosophy among potential new end users. DAPA will also encourage the scientific development of the field through a scientific journal devoted to publishing high-quality, original research in the digital-asset protection field. This magazine will focus on topics related to MATE attacks, including reverse engineering (malicious by hackers or analytical by malware analyzers), malware (design, disable, and detection), technical and software aspects of DRM, tamper proofing, obfuscation, watermarking, birthmarking for software protection, theoretical basis for software protection, practical studies of SP and reverse-engineering tools, secure and tamperproof hardware, whitebox cryptography, information hiding, and steganography in code.

Technology transition. DAPA will encourage the development of new research labs and companies around the themes of digital-asset protection. Technology transition will be achieved by supporting business intelligence activities and by identifying the grand challenges in the field, supporting virtual-lab infrastructures coordinating inter-university and industry-academy cooperation and lab networking.

Our Vision

Most of the information in digital societies is embedded in software, which has become increasingly pervasive. The long-term vision of DAPA is establish a unifying approach, with accompanying of tools and metrics, for addressing the real problem of information protection in the digital era.

A deeper understanding of the basic aspects of concealing and unveiling in programming languages and in software design can be the basis to extend these methods to any area where computational entities characterize dynamical behavior. DAPA seeks the development of secure, dependable, and trusted digital-asset infrastructures, in particular on the side of provable, secure information technologies; measurable security; metrics and benchmarks for their comparative evaluation; and definitive support for certification and standardization of procedures and methodologies in digital-asset protection, with particular emphasis on software protection.

DAPA targets both advanced software technologies, which impact the software development process and forensics, as well as foundational issues such as new developments in programming languages and software engineering.

These objectives will be achieved by considering the widest possible set of tools, including formal methods, empirical testing, and AI methods. We expect to expand the areas of DAPA application from the traditional DRM and intellectual property protection (IPP) contexts to new areas, where the protection of critical information is crucial. This includes financial systems, conditional access systems (CASs), copy-protection mechanisms, digital content protection, health information management systems, online systems, weapon systems, wireless systems, and any security solutions for security-critical systems. ■

References

1. B. Barak et al., “On the (Im)possibility of Obfuscating Programs,” *Proc. 21st Ann. Int’l Cryptology Conf. Advances in Cryptology*, Springer-Verlag, 2001, pp. 1–18.
2. A. Kerckhoffs, “La cryptographie militaire” [Military Cryptography], *Journal des Sciences Militaires*, vol. IX, Jan. 1883, pp. 5–83.

Christian Collberg is an associate professor at the University of Arizona. His research interests include intellectual property protection of software, compilers, and visualization. Collberg has a PhD in computer science from the Lund University, Sweden. Contact him at christian@collberg.com.

DAPA objectives will be achieved by considering the widest possible set of tools, including formal methods, empirical testing, and AI methods.

Jack Davidson is a professor at the University of Virginia. His research interests include computer security, programming languages and compilers, and embedded systems. Davidson has a PhD in computer science from the University of Arizona. Contact him at jwd@virginia.edu.

Roberto Giacobazzi is a professor at the University of Verona, Italy. His research interests include static program analysis, abstract interpretation, semantics, code obfuscation, and malware detection. Giacobazzi has a PhD in computer science from the University of Pisa. Contact him at roberto.giacobazzi@univr.it.

Yuan Xiang Gu is a chief architect at and co-founder of Cloakware of Irdeto in Canada and a guest professor of Northwest University of China. His research interests include compiler- and programming-language-based security, software security and protection, security modeling and metrics, digital asset protection, secure platforms and execution environments, security lifecycle management, and security in cloud computing and Internet of Things. Gu has graduated from computer science from Northwest University of China. Contact him at yuan.gu@irdeto.com.

Amir Herzberg is an associate professor at Bar-Ilan University, Israel. His research interests include security of information, communication, software and e-commerce, applied cryptography, and usable security. Herzberg has a DSc in computer science from Technion, Israel Institute of Technology. Contact him at amir.herzberg@gmail.com.

Fei-Yue Wang is the director of the State Key Laboratory of Management and Control for Complex Systems at the Chinese Academy of Sciences. His research interests include social computing, Web science, complex systems, and intelligent control. Wang has a PhD in computer and systems engineering from Rensselaer Polytechnic Institute. He is the editor in chief of *IEEE Intelligent Systems*. Contact him at feiyue@ieee.org.