Changing the Game of Software Security

Timothy Vidas | Secureworks
Per Larsen | Immunant
Hamed Okhravi | MIT Lincoln Laboratory
Ahmad-Reza Sadeghi | Technische Universität Darmstadt

DARPA's Grand Challenges are meant to invoke a type of innovation that is difficult to attain through traditional research avenues. Perhaps the most memorable, the 2004 effort toward self-driving vehicles, was simply dubbed the "Grand Challenge" at the time. Several such challenges have been designed and executed by DARPA since, each pushing the boundaries of science and technology. In computer and network security, a similar drive and innovation are present in a contest environment known colloquially as "capture the flag" or simply CTF.

The term "CTF" is borrowed from the physical game of capturing and defending literal flags. Today, the more apt analogy is likely the virtual variety found in first-person shooter video games. A computer security CTF often has digital flags, typically a sequence of secret bytes, that participants must defend and/or attack. Without delving into the details of the now rich and diverse community of such CTF contests, suffice it to say that the contests have grown in complexity and difficulty since the mid-1990s. Furthermore, the most difficult and well-regarded CTFs attract competitive teams that curate strategy and capability for years.

At its core, the Cyber Grand Challenge (CGC) was meant to discern whether an autonomous, purpose-built system could compete in the highest levels of computer security CTFs. Years of effort culminated in the summer of 2016 as the CGC Final Event (CFE) was held in conjunction with the DEF CON conference.

This issue of *IEEE Security & Privacy* explores several aspects of autonomy with respect to computer hacking from varying perspectives centered on the CGC. As such, it is worthwhile to introduce the CGC parlance used throughout this issue. The competitors in the CFE were autonomous machines, physical racks of high-performance computing gear, dubbed cyber reasoning systems (CRSs). Obviously humans designed and programmed the inner workings of

1540-7993/18/$33.00 © 2018 IEEE

each CRS, but at the CFE, humans were mere spectators. The "course" that every CRS had to "navigate" was in the form of novel, known-vulnerable software. These challenge sets (CS) were composed of challenge binaries (CB) that were uniformly distributed to each CRS, which in turn had to 1) determine the vulnerable conditions and 2) prove that the conditions existed on opponents while simultaneously thwarting such attempts by others.

Every few minutes, a new round would begin, meaning that CSs may be introduced or removed, and proofs of vulnerability (PoVs) could be launched several times against various opponents. Concomitant, each CRS could elect to mitigate vulnerabilities; however, CRS-fielded CBs (and network IDS signatures) were readily made available to opponents, mimicking some properties of real-world patching paradigms.

Many articles in this issue mention DECREE, a CGC-specific operating system interface specification. DECREE was created to narrow both the space in which the contestants competed and also the risk present from evaluating competitor-provided software. Unlike the hundreds of system calls present in modern operating systems, DECREE employs seven. The seven specific calls were meant to be expressive enough to model most memory-related vulnerability classes. The binary format for DECREE borrows heavily from the common Executable Linkable Format (ELF) file format, and the competition framework integrity team implemented DECREE on both 32-bit Linux and 64-bit FreeBSD.

In the end, seven diverse CRS finalists all successfully participated in the CFE. After 96 rounds, or just over nine hours, one emerged victorious (the winning team contributed the article on page 52 of this issue). Foremost, the CGC proved that a CRS could be built—that is, a computer could play in a CTF-style event, by itself. The CGC also provided a specification for an autonomous, brokered CTF that has already been reused in other events, as has the special binary specification for CBs. Such reuse and the various CGC-related corpora are giving researchers common ground on which to further advance that state of the art.

Much work remains, however. Most CRS creators will readily admit that the reasoning aspects of their CRS (for instance, game theory, artificial intelligence, and machine learning) were rudimentary. Indeed, as the CFE was the first contest of its kind, there was no historical record to guide training. Similarly, it is difficult to discreetly articulate individual advances in any particular component domain employed by a CRS. For instance, fuzzing technology materially advanced during the CGC timeframe, but would the same advancements have occurred absent the CGC? Perhaps the most telling metric, of the 82 CSs employed in CFE, vulnerabilities were only proven in 20 (that is, less than a quarter).

No automobiles completed DARPA's 2004 challenge course. Just months later, in 2005, not only was a victor declared, but 22 of 24 contestants successfully navigated the rural course. In 2007, six contestants similarly completed an urban course. Now, 13 years later, we are seeing fully autonomous vehicles navigate public roads alongside human drivers. In 2016, every contestant in the world's first autonomous computer hacking tournament demonstrated a level of proficiency in autonomous vulnerability discovery, proof, and mitigation. It makes one wonder what levels of autonomy will be achieved in software security in the coming decade. ■

Timothy Vidas is a senior distinguished engineer at Secureworks. Contact at tvidas@secureworks.com.

Per Larsen is currently the CEO of Immunant. Contact at perl@immunant.com.

Hamed Okhravi is a senior staff member at MIT Lincoln Laboratory. Contact at hamed.okhravi@ll.mit.edu.

Ahmad-Reza Sadeghi is a professor of computer science at Technische Universität Darmstadt. Contact at ahmad.sadeghi@trust.cased.de.