The Sixth IEEE S&P Workshop on Language-Theoretic Security (LangSec)

The Sixth *IEEE* S&P Workshop on *Language-Theoretic Security (LangSec)* co-located with the 41st *IEEE Symposium on Security and Privacy* solicits contributions related to the growing field of language-theoretic security.

Venue: The Hyatt Regency, San Francisco, CA

Date: May 21, 2020

Website: `http://spw20.langsec.org/`

About LangSec

The Language-theoretic approach (LangSec) is mission assurance for connected software and hardware exposed to attacks via malicious inputs. LangSec offers a practical data and code co-design methodology accessible to developers of everyday software.

In a nutshell, since any software that processes inputs is, in fact, an interpreter, and any inputs are thus its programs, input validation is not essentially different from program verification. To be trustworthy, input validation must therefore be grounded in models that describe precisely what the properties of valid inputs are. Luckily, such models exist, and can be made accessible to regular developers.

The language-theoretic approach (LangSec) is mission assurance for connected software and hardware exposed to attacks via malicious inputs–through a practical data and code co-design methodology and filtering of legacy formats down to safe subsets.

LangSec regards the Internet insecurity epidemic as a consequence of ad hoc input handling. LangSec posits that the only path to trustworthy computer software that takes untrusted inputs is treating all valid or expected inputs as a formal language, and the respective input-handling routine as a parser for that language. Only then can any correctness guarantees be assured for the input-handling code. Ambiguity of message/protocol specification is insecurity; ad hoc parsing is an engine of exploitation; overly complex syntax can make judging security properties of input impractical or even undecidable.

LangSec explains why ad hoc "input sanitization", "sanity checking", and other advice to be more careful with inputs is not enough, and why numerous secure programming initiatives have not ended input-driven exploitation. LangSec is also a code and protocol auditing methodology.

Treating input-handling code as an automaton allows the defender to reason about its behavior. The more limited computational power of the automaton, the easier the reasoning. The root cause of many bugs, memory corruptions, and exploitation is trying to validate inputs with inappropriate automata (e.g., much of XSS is due to "validating" context-free HTML with regexps). The recognizer automaton should be just as powerful as warranted by the message format, and no more; unnecessary complexity is computational power given to the attacker.

The 6th installation of the workshop will continue the tradition and further focus on methodologies (1) that can infer formal language specifications from samples of electronic data, (2) that can generate secure parsers from formal specifications of electronic data, and (3) that describe the complexity hierarchy of verifying parser implementations.

Important Dates (tentative)

- Paper submissions due: 15 January 2020, 11:59 PM Pacific
- Research Reports, Panels, and Proof-of-concept submissions due: 1 February, 2020, 11:59 PM Pacific

- Notification to authors: 15 February 2020
- Final files due: 5 March, 2020

Previous workshops:
`http://spw14.langsec.org/` (keynoted by Caspar Bowden and Felix 'FX' Lindner)
`http://spw15.langsec.org/` (keynoted by Dan Geer)
`http://spw16.langsec.org/` (keynoted by Douglas McIlroy)
`http://spw17.langsec.org/` (keynoted by Perry Metzger)
`http://spw18.langsec.org/` (keynoted by Mike Walker)
Full papers and presentations freely available at the workshop websites above.