NSF SaTC Investigator Meeting—January 10, 2017

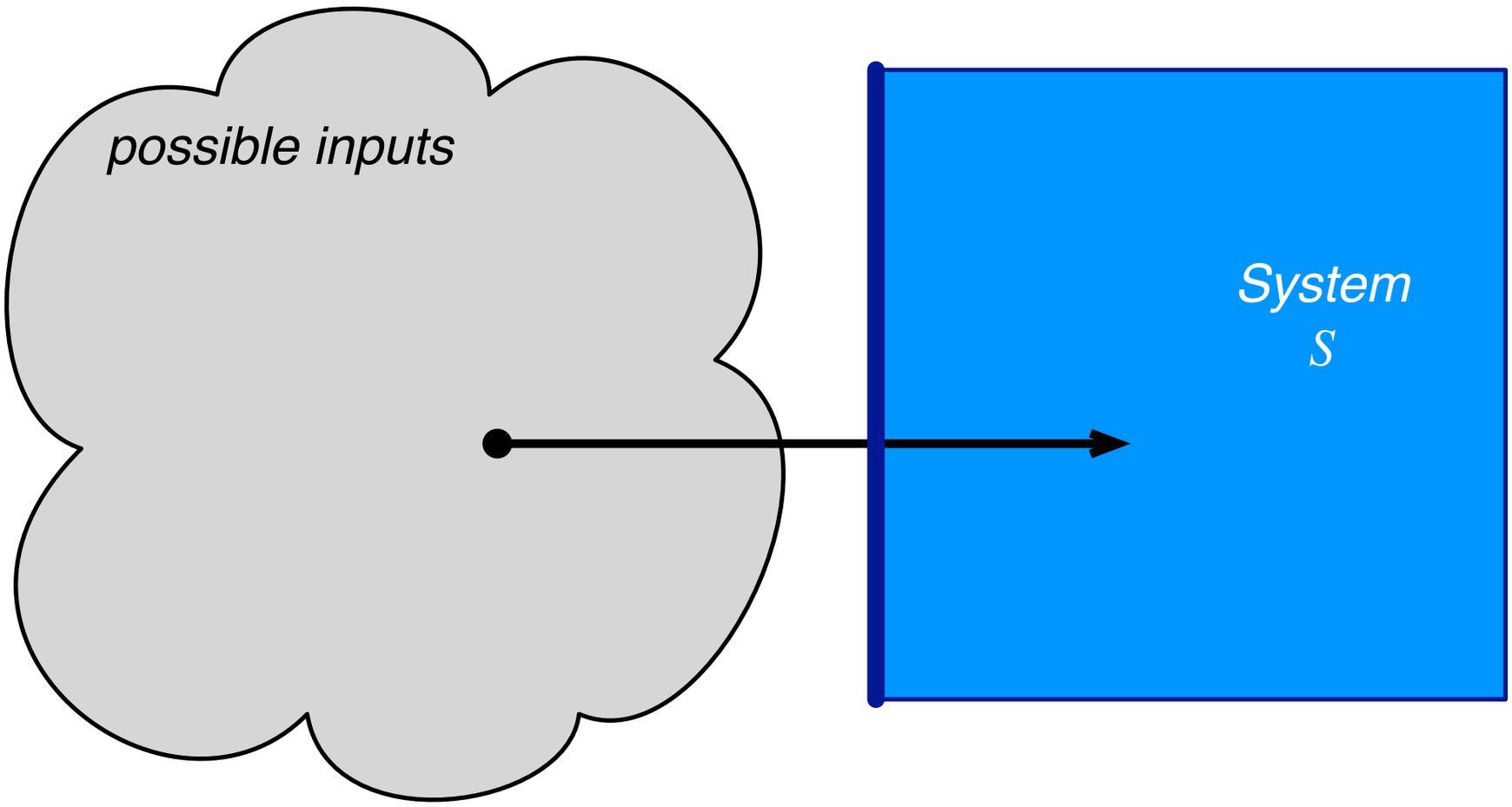
Stopping 0-Days with Formal Languages

Sergey Bratus
Sean W. Smith
{sergey,sws}@cs.dartmouth.edu

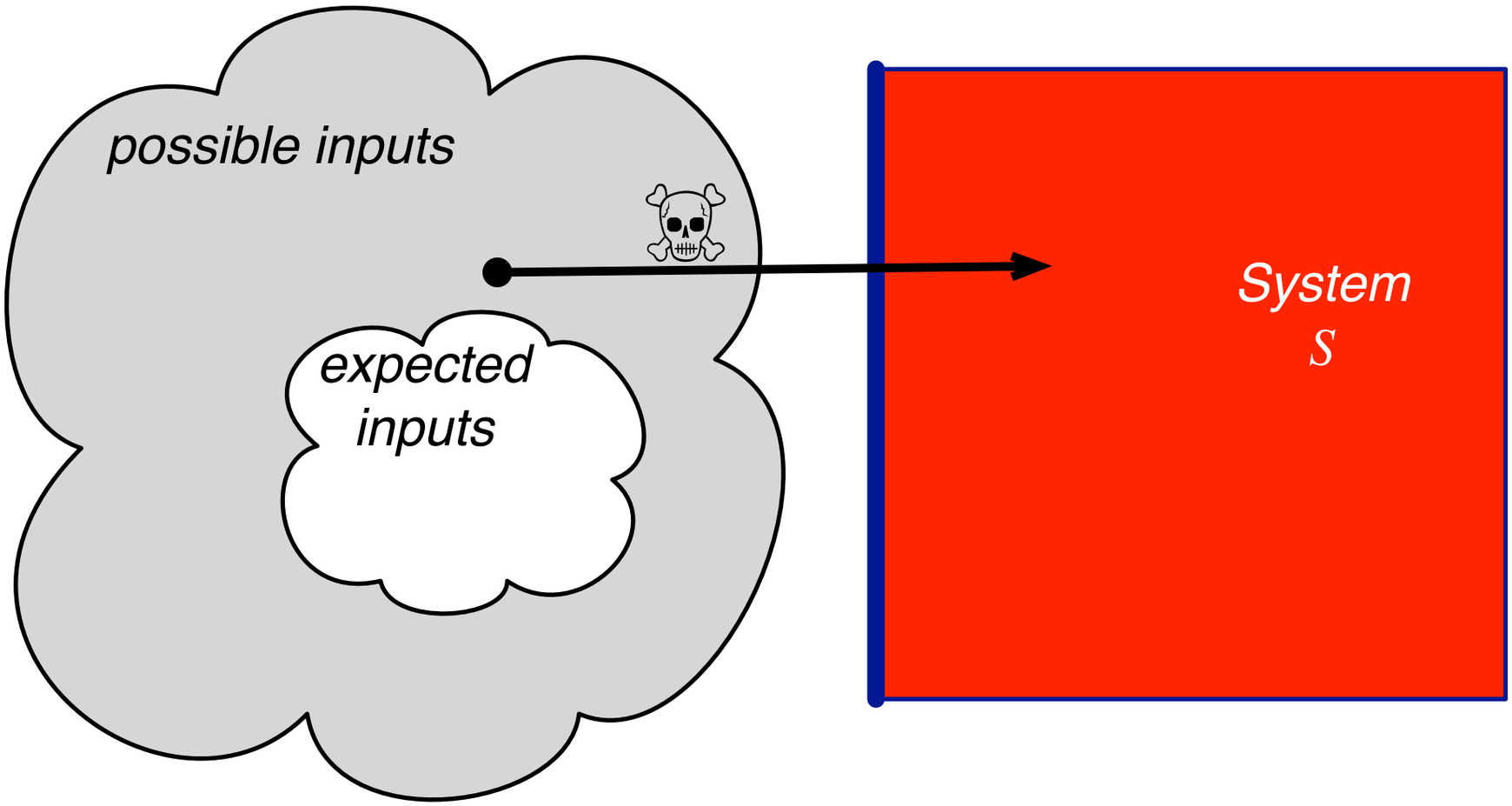
Department of Computer Science/
Institute for Security, Technology, and Society

Dartmouth College

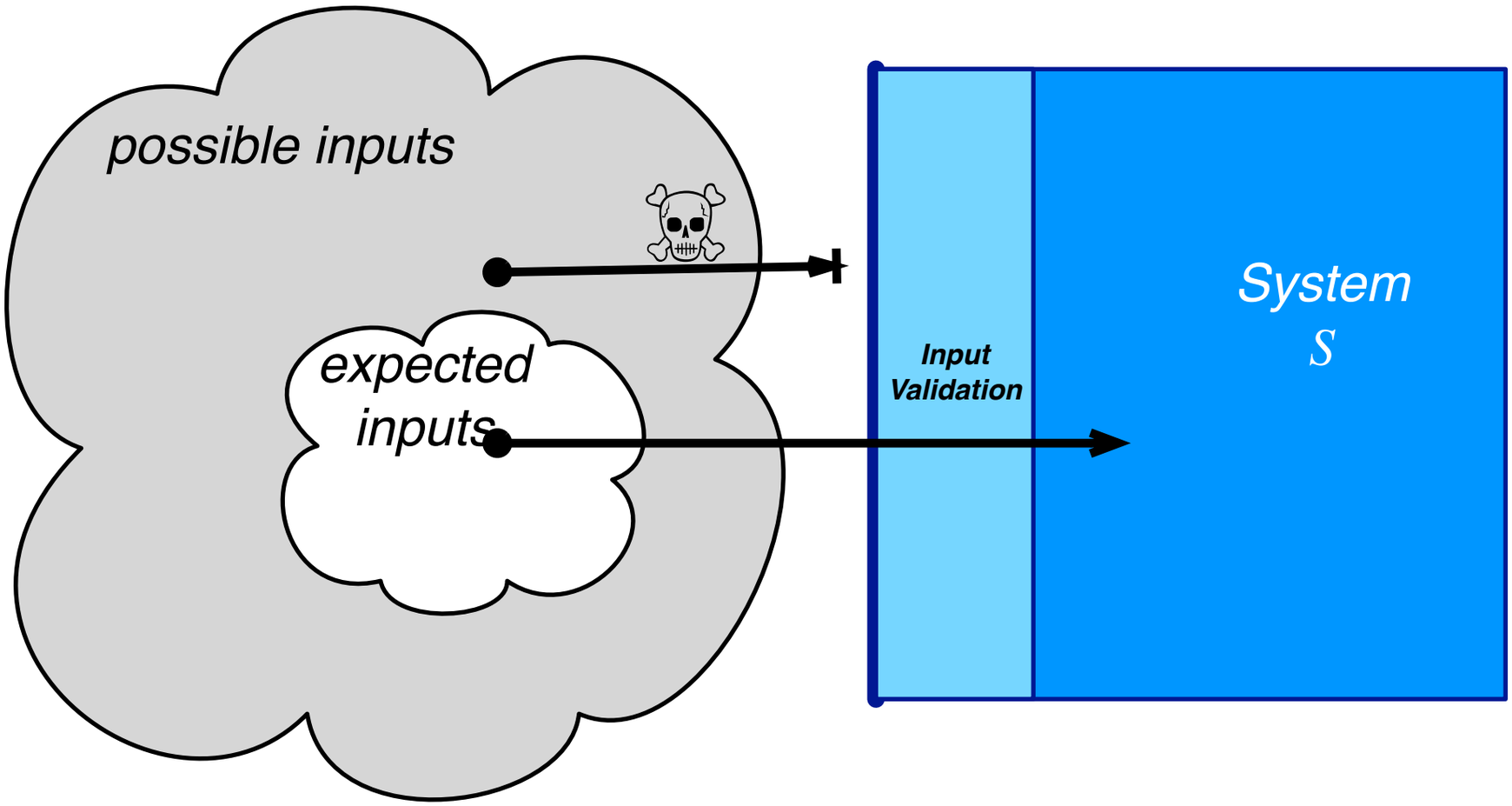
The Big Picture



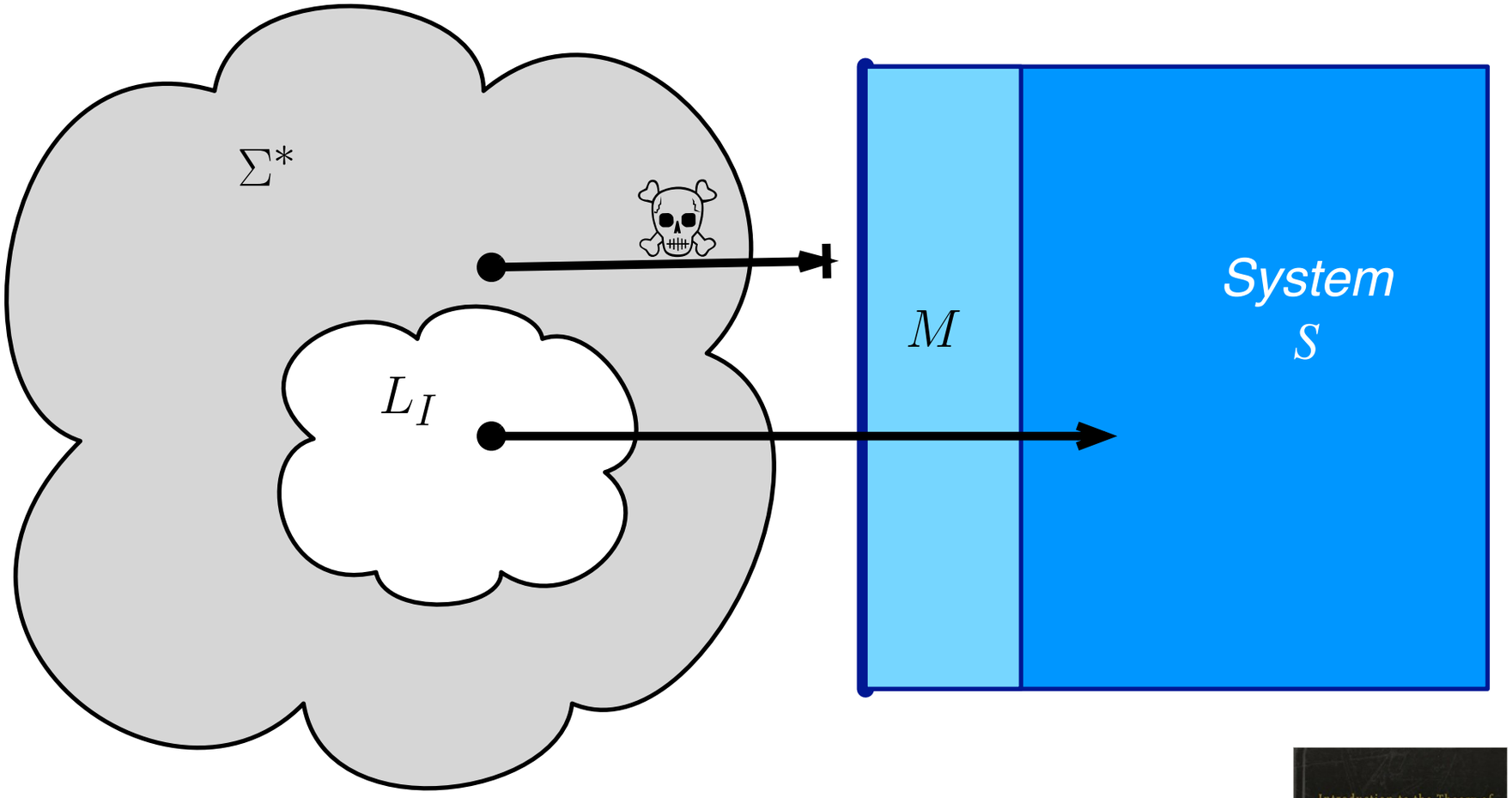
The Big Picture



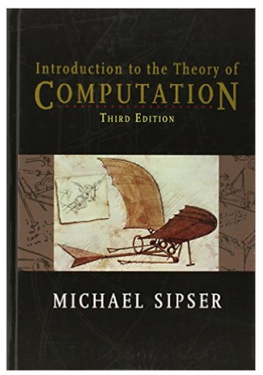
The Big Picture

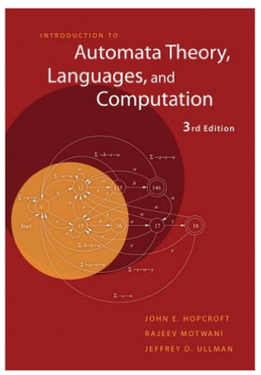


Language Theory!

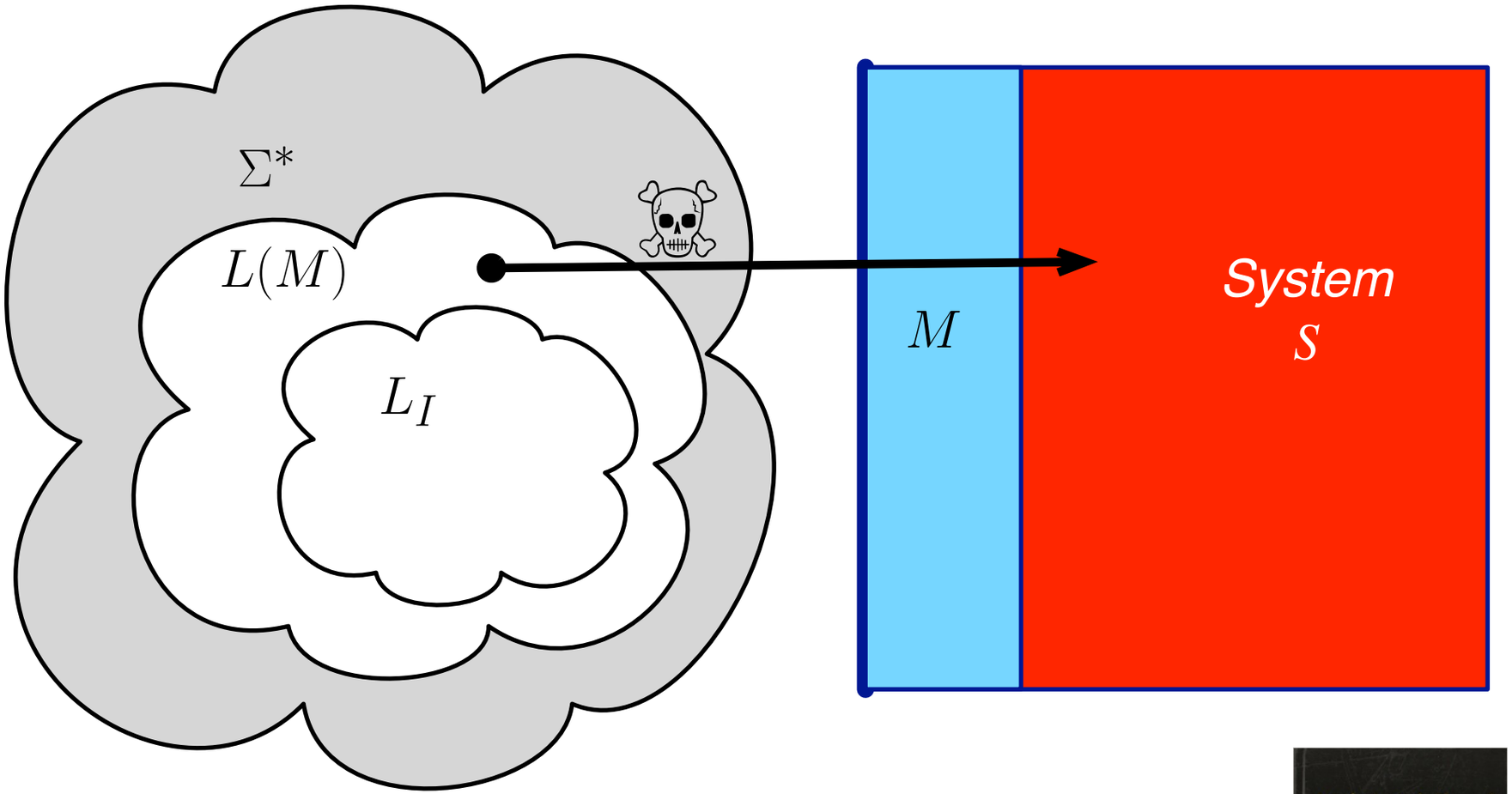


$$L_I \stackrel{?}{=} L(M)$$

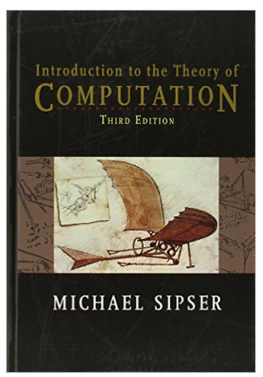


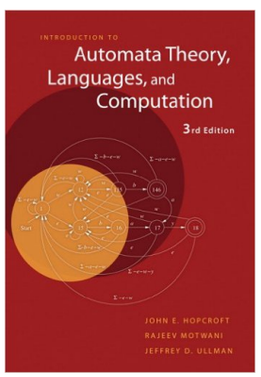


Language Theory!

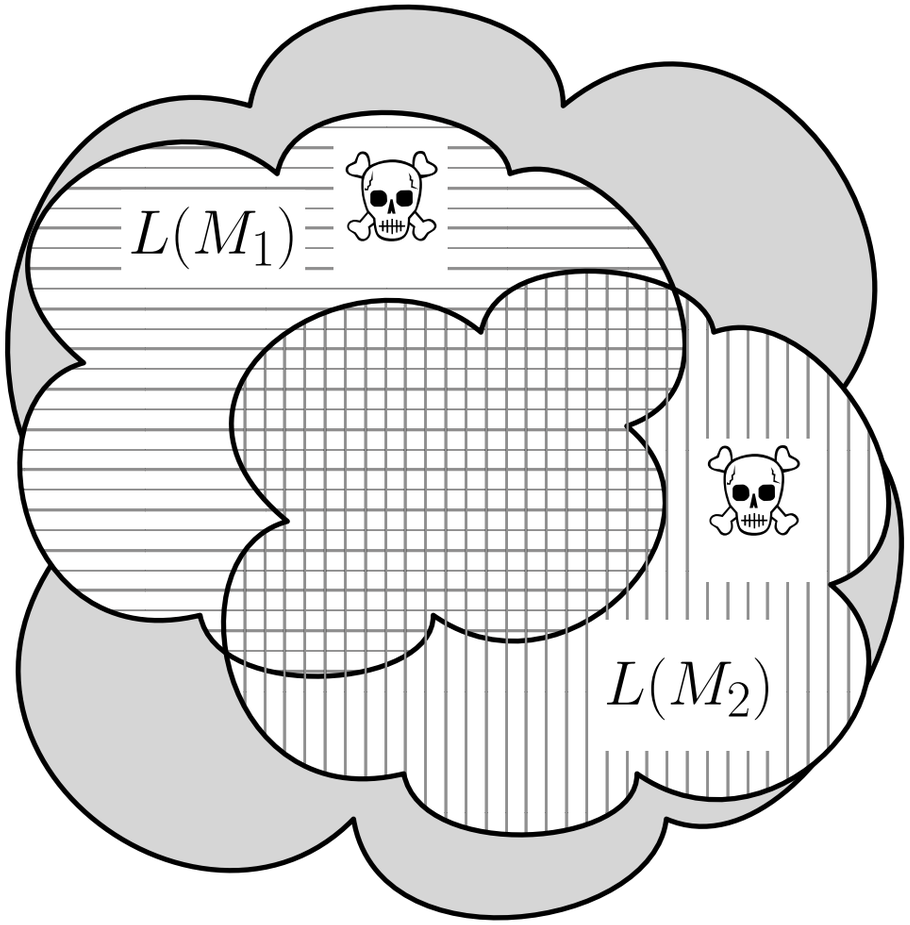


$$L_I \neq L(M)$$

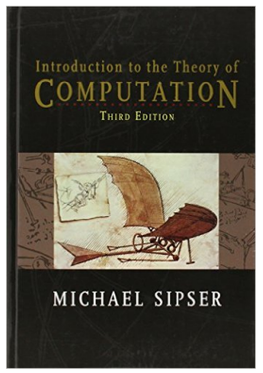


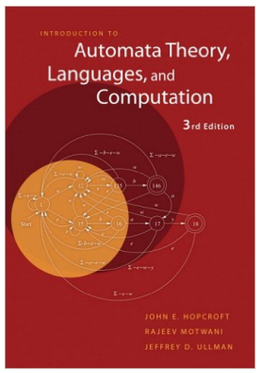


Differential Parsing

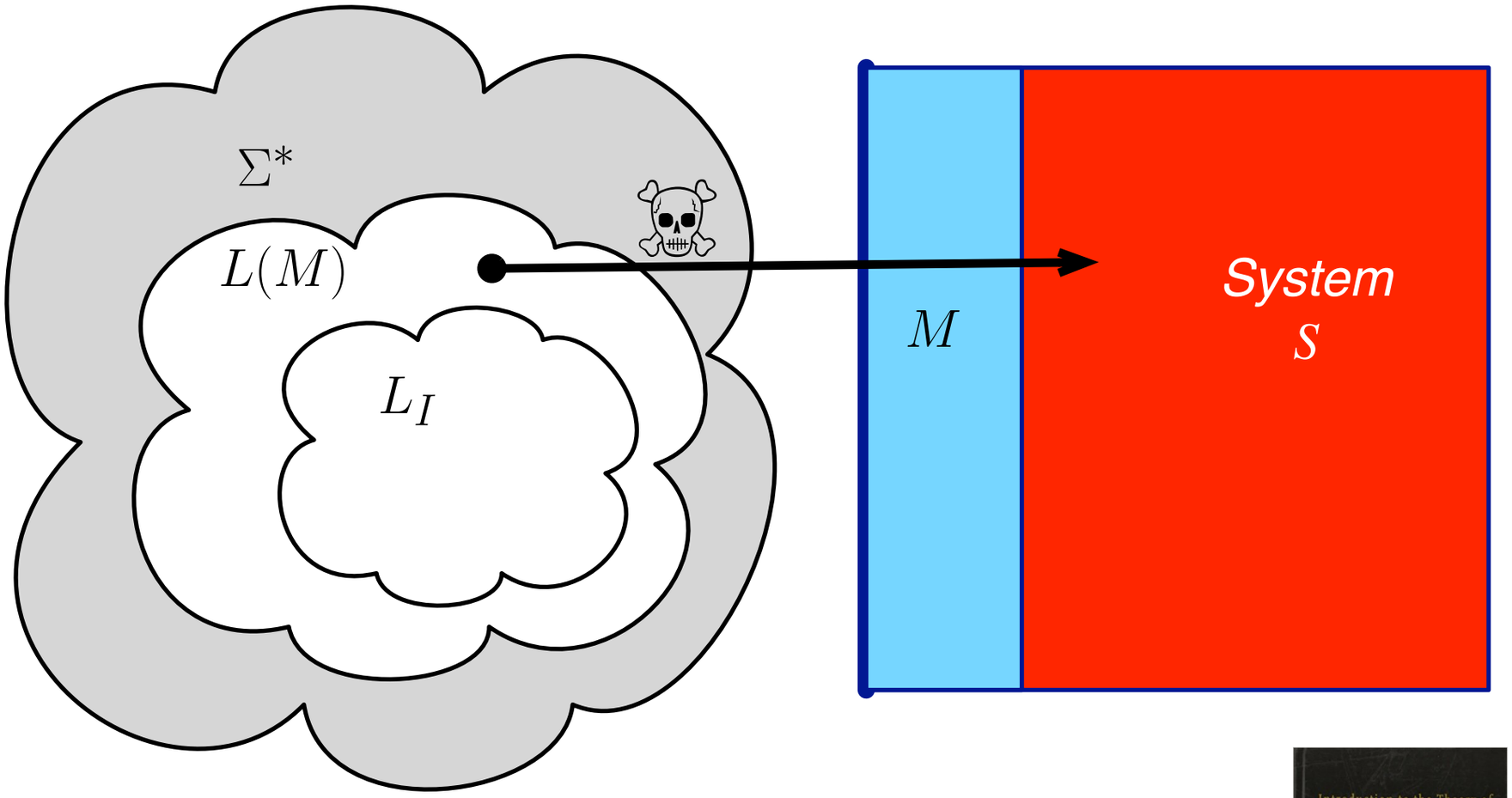


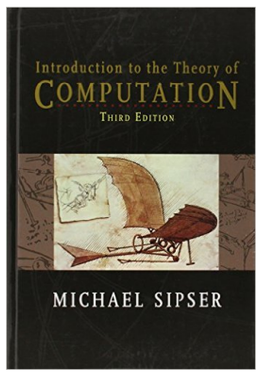
$$L(M_1) \stackrel{?}{=} L(M_2)$$

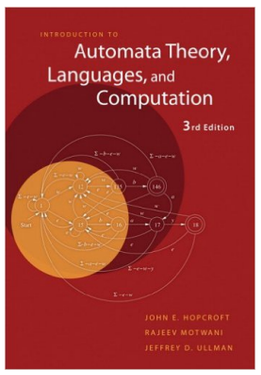




Weird Machines







$$\{M' \ : \ \langle M', \cdot \rangle \in L(M)$$
$$\forall w \in \Sigma^* \ S(\langle M', w \rangle) = M'(w)\}$$

Input validation is a **verification** problem

- Being "valid" should be a **judgment** about **behavior** of inputs on the rest of the program

- "Every input is a program". Judging programs is very hard, unless they are **very simple**.

    - Cf. *"Everything is an interpreter"*, Greg Morrisett

- "Input validation" is thus a **verification problem**, though unusually posed

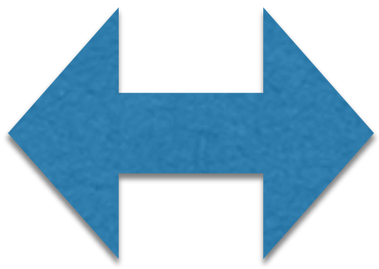
    - But not unusual for attackers, who think of crafted data payloads as instructions, vulnerable code as CPUs/VMs

Complex input language makes security undecidable

- "Validating input" is **judging** what **effect** it will have on code
    - "Is it **safe** to process?" == "Will it cause **unexpected computation** on my program?"
- Make the judgment as **simple** as possible: e.g.: "regular or context-free, syntactically valid == safe"
    - Comp. power of recognizer **rises** with language's syntactic complexity (Chomsky hierarchy)
- Rice's theorem, halting problem: you **can't** judge effects of Turing-complete inputs. (Cf. Ethereum/DAO)

"trouble"/
weakness

Data format



Parser Structure

"Data format is code's destiny"

"Everything is an interpreter (=parser)"

"Every sufficiently complex input processor
is indistinguishable from a VM
running inputs as bytecode"

Typical anti-pattern: "shotgun parser"

- Parsing and input-validating code is **mixed with** and **spread across** processing code
- Input checks are **scattered** throughout the program
- **No** clear **boundary** after which the input can be considered fully checked & **safe** to operate on
- It's unclear from code **which properties** are **being** checked & which **have been** checked

Input handling must be minimalistic, data & code co-designed

- Input-handling code should do **nothing** more than **consume** input, **validate** it (correctly) & **deserialize** it
    - Use the **exact** complexity needed to validate & create **well-typed** objects
    - Reflection, evaluation, etc. **don't belong** in input-handling code (even if "sanitized")
- Any extra computational power exposed is **privilege** given away to **attacker**

Recognizer must be **equal** in power to input language

http://stackoverflow.com/questions/1732348/regex-match-open-tags-except-xhtml-self-contained-tags

You can't parse [X]HTML with regex. Because HTML can't be parsed by regex. Regex is not a tool that can be used to correctly parse HTML. As I have answered in HTML-and-regex questions here so many times before, the use of regex will not allow you to consume HTML. Regular expressions are a tool that is insufficiently sophisticated to understand the constructs employed by HTML. HTML is not a regular language and hence cannot be parsed by regular expressions. Regex

will **devour your HT**ML parser, application and existence for all time like Visual Basic only worse *he comes he com*es *do not fi*ght h**e com~~es, hi~~s uŋholy radiańcé de*stroying all enlightenment, HTML tags **leaking from your eyes like liq**uid *p*ain, the song of re*g*ular expr~~ession parsing~~ will ext*inguish the voices of mor***tal man from the sp***here I can see it can you see it it is beautiful t*he f `inal snuf` fing o*f the lie***s of Man ALL IS LOST A*LL IS L**OST the *pony he come*s he ~~comes he comes~~ **the ich**or permeate*s al*l MY FAC*E MY FACE oh god n***o NO NOOOO N**Θ stop t*he an* `gles are` **not real ZALGO IS TONY THE PONY COMES**

Input specification must be **formal** and **complete**

- Incompleteness leads to parser differentials (many in X.509/ASN.1)

- Without clear assumptions, the C.A.R. Hoare's P {Q} R **chain** of assumptions & checks breaks

  - What is "valid" input? What's to be rejected?

- Doomed if more than one module (or programmer) is involved

  - Cf.: OpenSSL CVE-2016-0703, LibNSS CVE-2009-2404, ...

Thank you!

{sergey,sws}@cs.dartmouth.edu

Join us for

4th IEEE Security & Privacy LangSec Workshop

May 25, 2017
San Jose, CA

http://spw17.langsec.org

http://langsec.org