An Initial Investigation of Protocol Customization

David Ke Hong, Qi Alfred Chen, Z. Morley Mao

University of Michigan

Today's protocols are feature-rich

- Widely-used protocols contain a rich set of features and extensions
  - Around 15 extensions for the functionality provided by the TLS protocol message formats
  - Different usage scenarios
    - TCP extensions
  - Performance consideration
    - Various HTTP/2 features
  - **Implemented as a one-size-fits-all library**

Vulnerabilities caused by unnecessary features

- Not all features are desirable in a particular deployment scenario, and unused features enlarge attack surface
  - **HeartBleed** attack caused by an implementation flaw in **TLS/DTLS heartbeat extension**
    - Optional in many deployment scenarios

  - **FREAK** attack exploiting weak **RSA_EXPORT cipher suites**
    - Stronger cipher suites already available

Protocol Customization

- Modify and specialize a standard protocol to enable only desirable features
- Compile-time disabling
  - 97 **OpenSSL_NO*** compiler flags

OpenSSL

- Runtime disabling or parameter tuning
  - **mod_*** parameters for module disabling

Apache HTTP SERVER

Existing customization practices

- Existing customization practices are ad-hoc
  - Often relying on configurations offered by the protocol implementation

- Case study
  - Per-feature disabling on **HTTP/2 features** is not supported in Apache HTTP server
  - HPACK bomb vulnerability (CVE-2016-1544, CVE-2016-6581)
    - Developer failed to cover this customization option

HTTP/2

Systematic way of protocol customization is needed

- Call for a ***systematic*** approach to overcome existing limitations
  - Minimizing human efforts and errors
  - Covering customization on important features
  - Supporting customization of fine-grained features

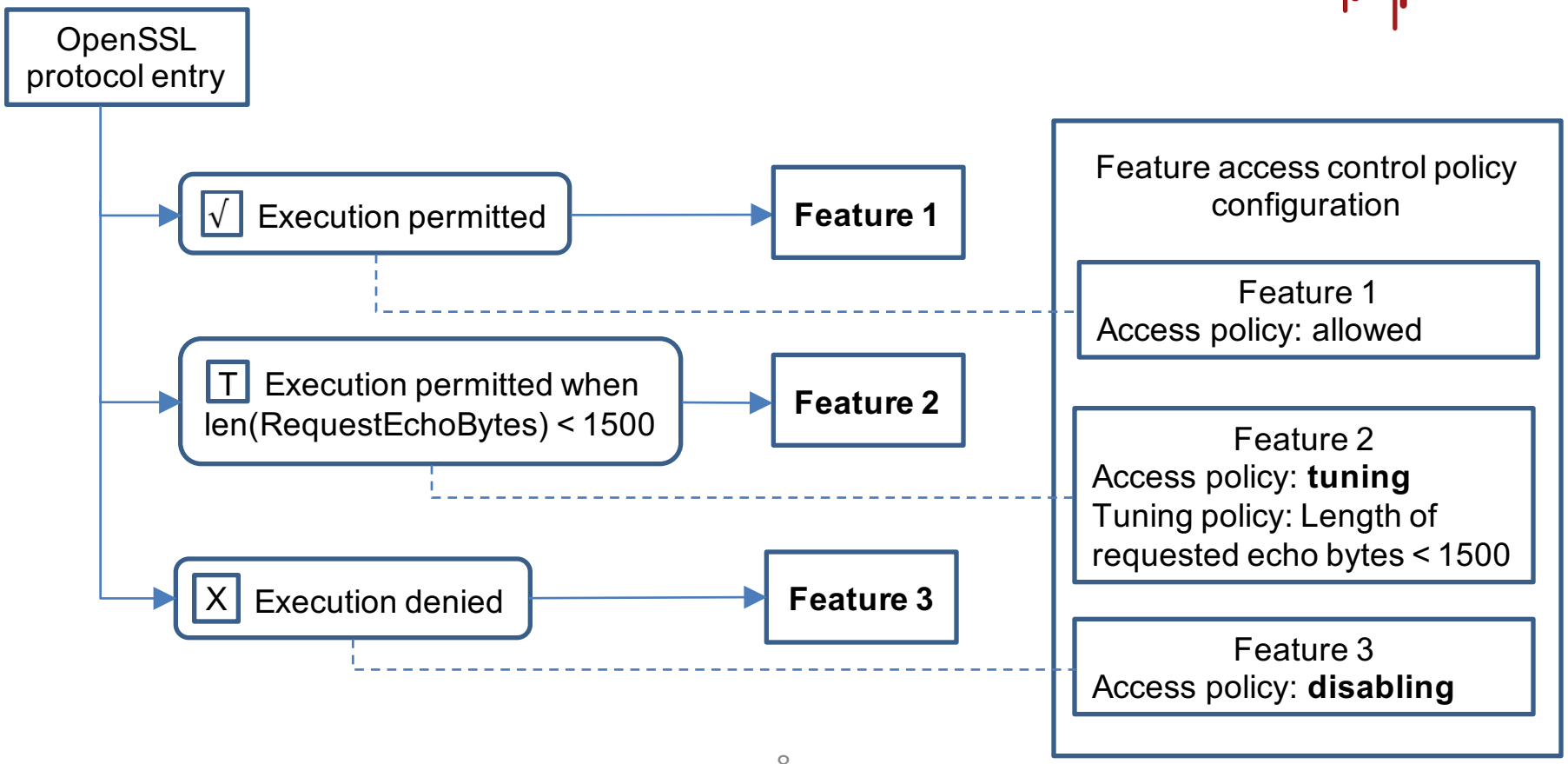
- Question: can we ***systematically*** customize a standard protocol to reduce its attack surface with sufficient automation?

Solution direction

- **Protocol feature access control**
  - A systematic framework to unify common protocol customization practices
  - Access control resource: protocol feature
  - Two types of access control policy
    - Feature disabling policy
    - Feature tuning policy
  - **Validation**: 17 out of 20 CVE patches can be expressed by feature disabling or tuning policy

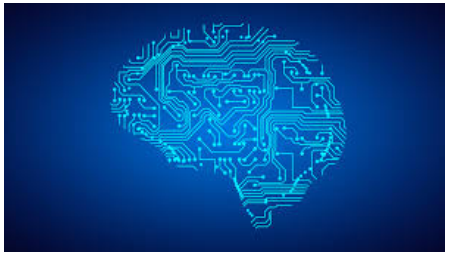
Access control example: HeartBeat

- **To prevent HeartBleed vulnerability**

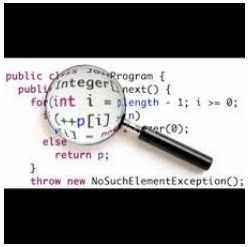


Research challenges

- How to systematically **identify features and locate its code-level implementation**
  - Bridging the gap between user-level and code-level features
    - Natural language processing
    - Deep neural networks

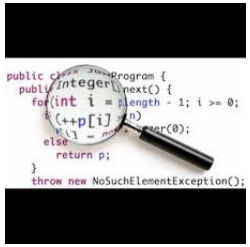


  - Systematically locating code-level feature-related implementation
    - Control and data flow analysis



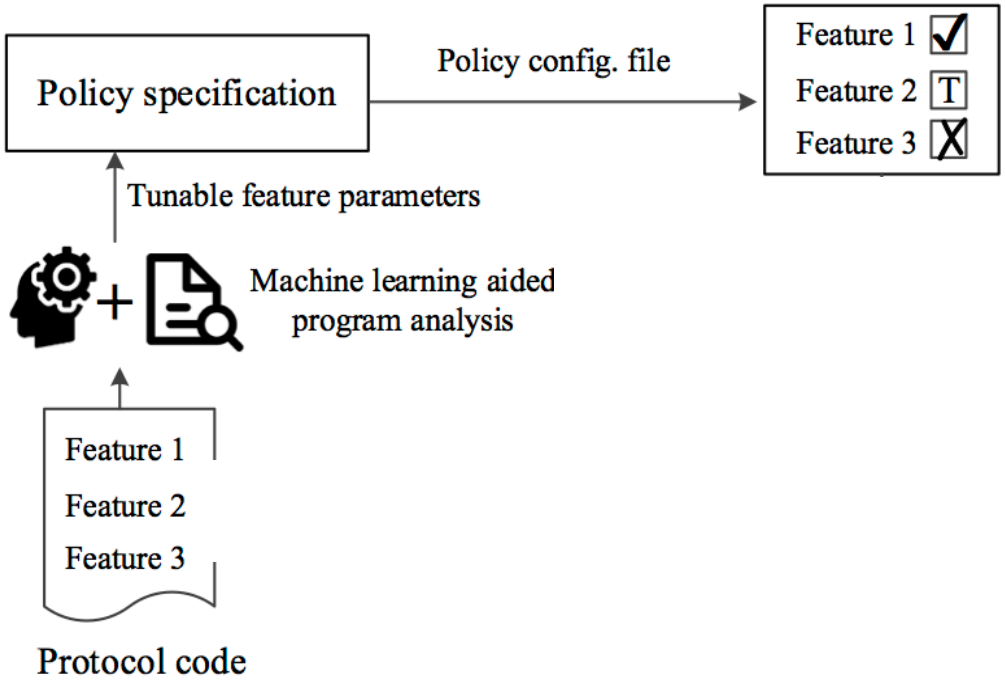
Research challenges

- How to effectively **support diverse types of protocol customization** with **minimized manual efforts**
  - Enforcing policies without assuming that the code base structure is ready for customization by design
    - Control and data flow analysis
  - Supporting feature disabling and tuning policy
    - Control and data flow analysis
    - Symbolic execution



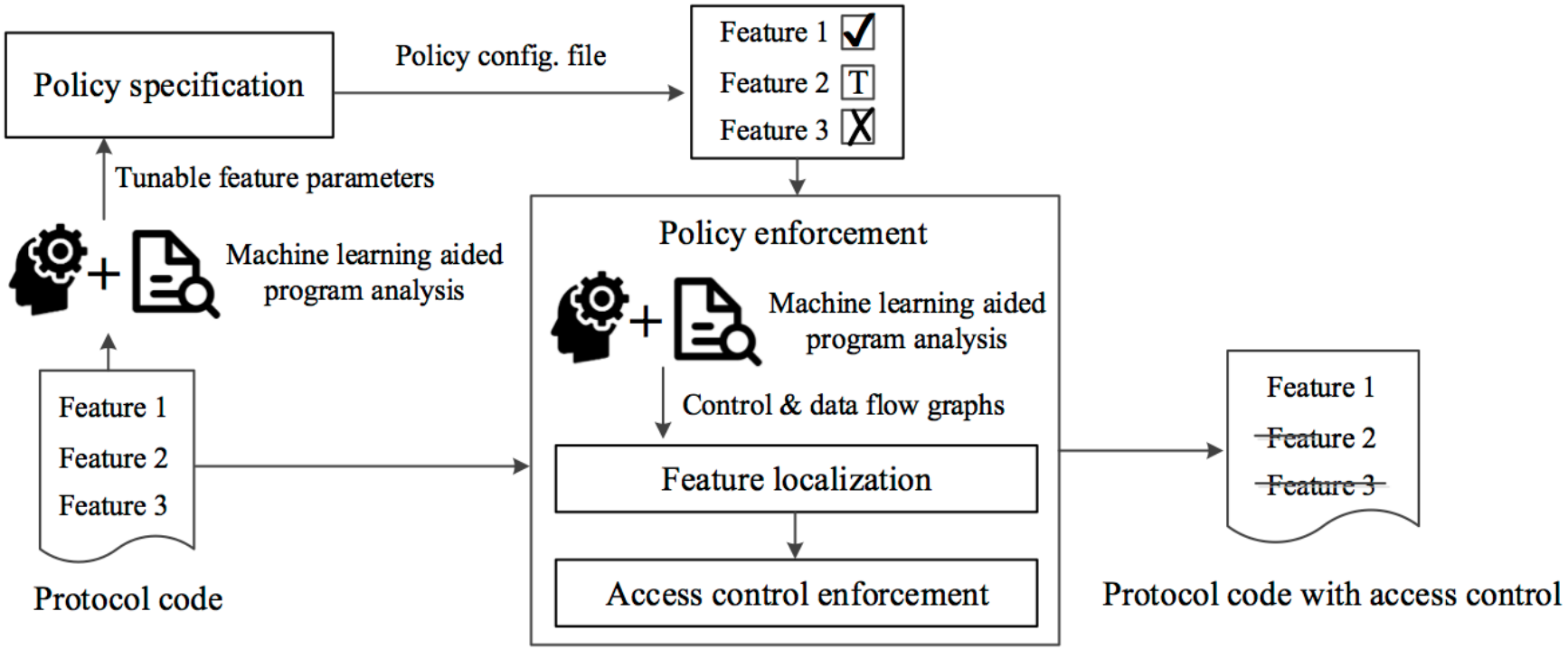
Preliminary system design

Input: features to be customized, protocol software



Preliminary system design

Input: features to be customized, protocol software



Limitation

- Protocol customization alone is insufficient in addressing some vulnerability cases
  - Vulnerability related to core functionality that requires significant change to the details of a protocol feature
    - TLS vulnerability caused by the weakness in key generation

Summary

Perform an initial investigation of protocol customization for reducing attack surface of a standard protocol

- Identify key research challenges for systematic and sufficiently automated protocol customization
- Propose an access control mechanism to unify existing protocol customization practices

Future work

- Feature identification using NLP techniques
- Feature access control: more detailed design and impl.

Thank you!

- Questions?