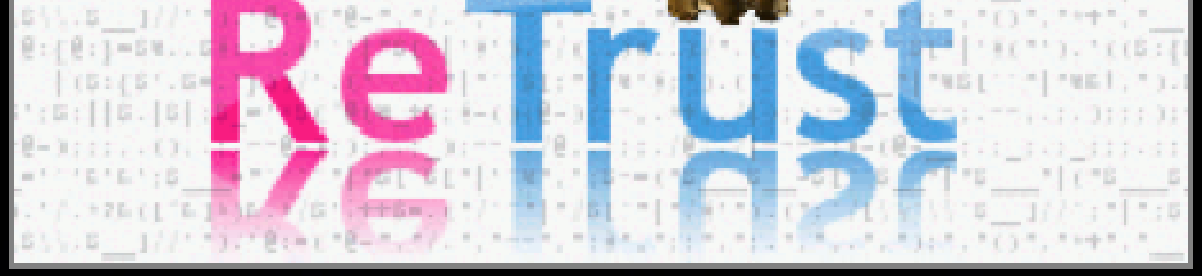
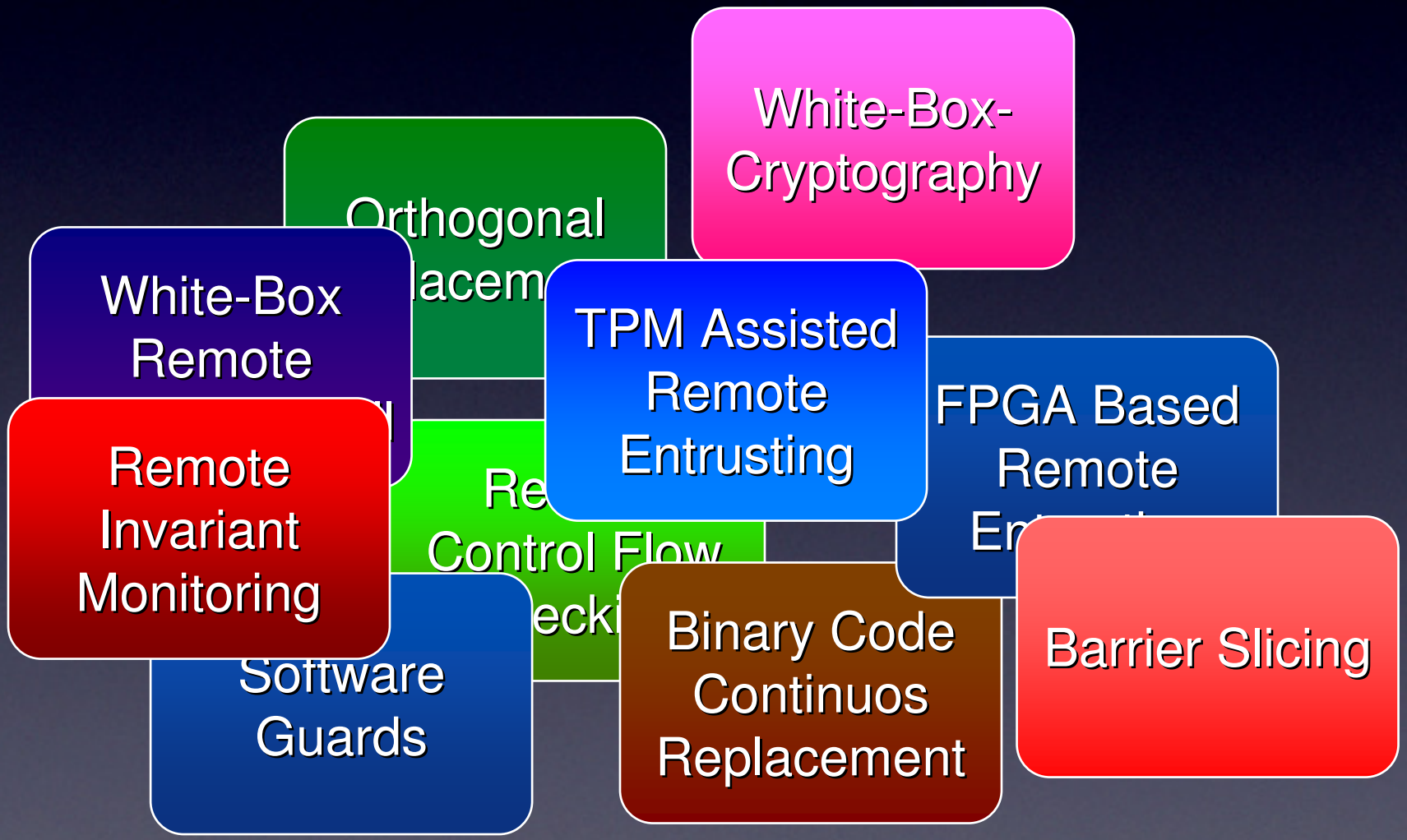


Re-Trust Demonstrators

Riva del Garda - October 2nd, 2009

What has been done?



What has been done?

White-Box Remote

Orthogonal

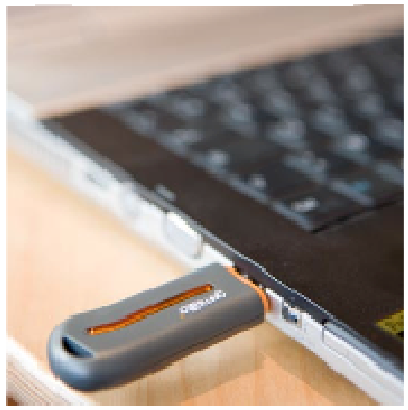
White-Bo Cryptogra

TPM Assisted Remote Entrusting

Remote

Slicing

Code



- Secure drive
- Run-time checking (Thumbp

What to demonstrate?

Feasibility on complementary application domains

Limitations:

Complexity of techniques

Need of automation to deploy protection in big applications

Software/Hardware protection

Case Study Application

- Fat-Client Network Game Application
  - Local knowledge of the track
  - Limited speed
  - Fuel Monitoring
  - Images and advertising control
  - ...

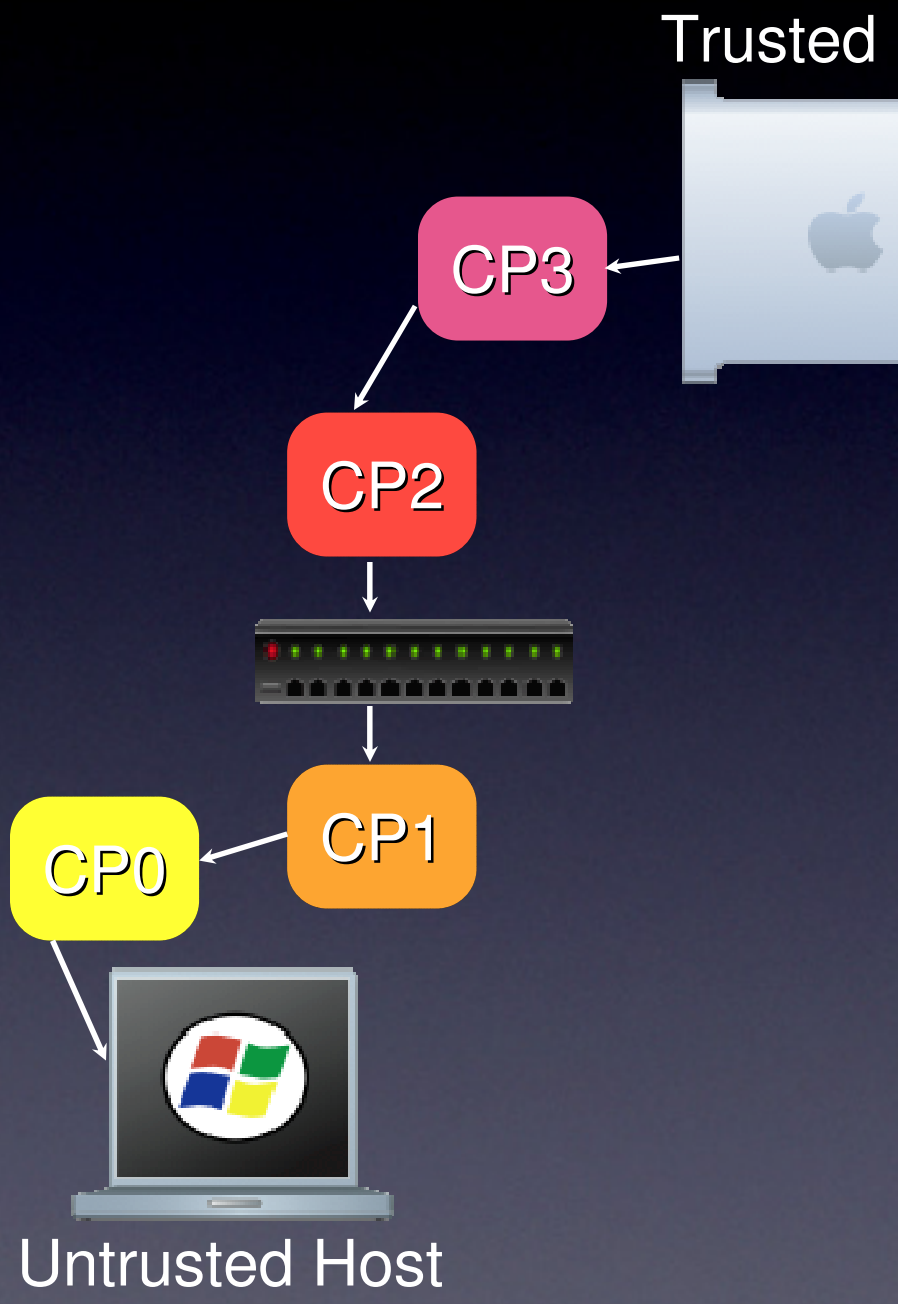
Java Demonstrator

Orthogonal replacement

Periodically replace the client code with a new version

Orthogonal (obfuscated)

Semantically different (due to interlocking)



Java Demonstrator

- **Obfuscation**: code is hard to understand and attack
- **Orthogonality**: non-determinism in the obfuscation to generate many possible orthogonal clients

Java Demonstrator

- **Interlocking**
- The code of each block split between rusted and untrusted host
  - Orthogonality with respect to previous clients
  - An expired client can not longer be used (it would not work with the new server)

Java Demonstrator

- **Replacement**
- GEMALTO SMART DONGLE
  - The dongle is tamper-resistant (read-only)
  - Whenever a new client is released, it is downloaded into the dongle
  - If the update is refused, the old code does not run because of interlocking

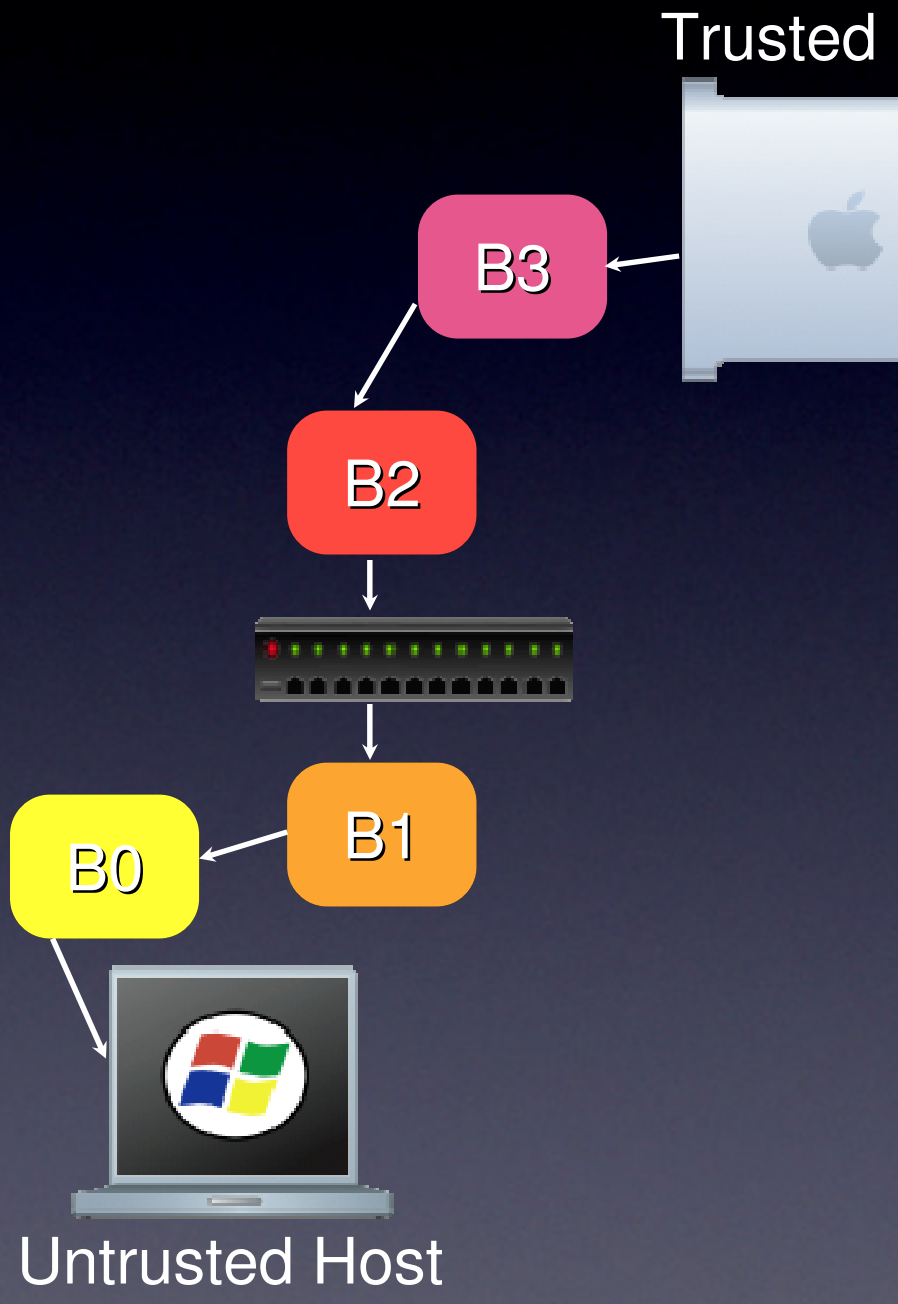
C++ Demonstrator

Binary obfuscation through continuous code replacement

Client split into blocks of code

Blocks sent at run-time

Blocks continuously relocated in memory to make dynamic analysis hard



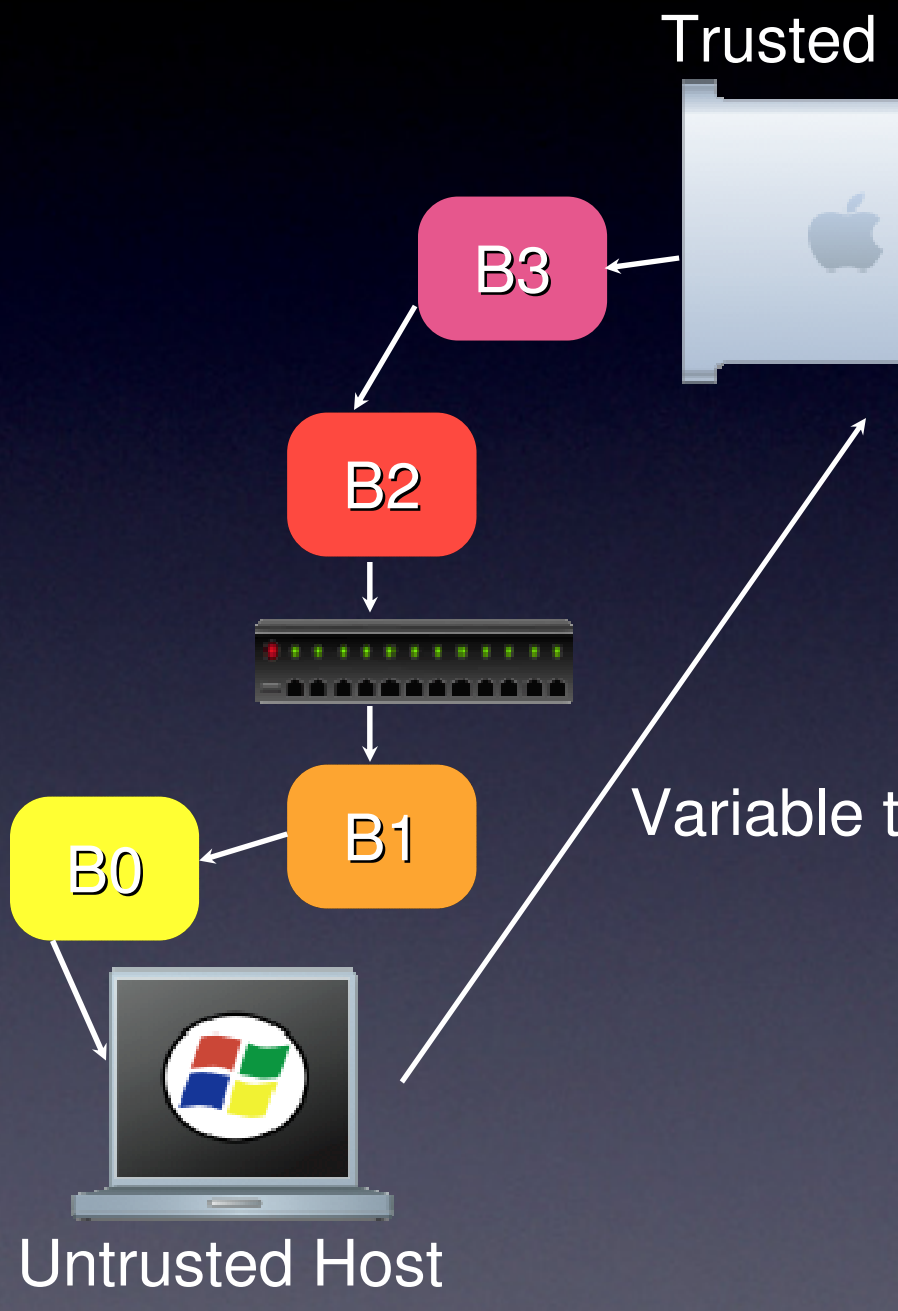
C++ Demonstrator

Invariant and Control Flow monitoring

Selected properties of the program are continuously checked and run-time

Properties checked in precise points of the control flow

Any violation of these properties stops the game

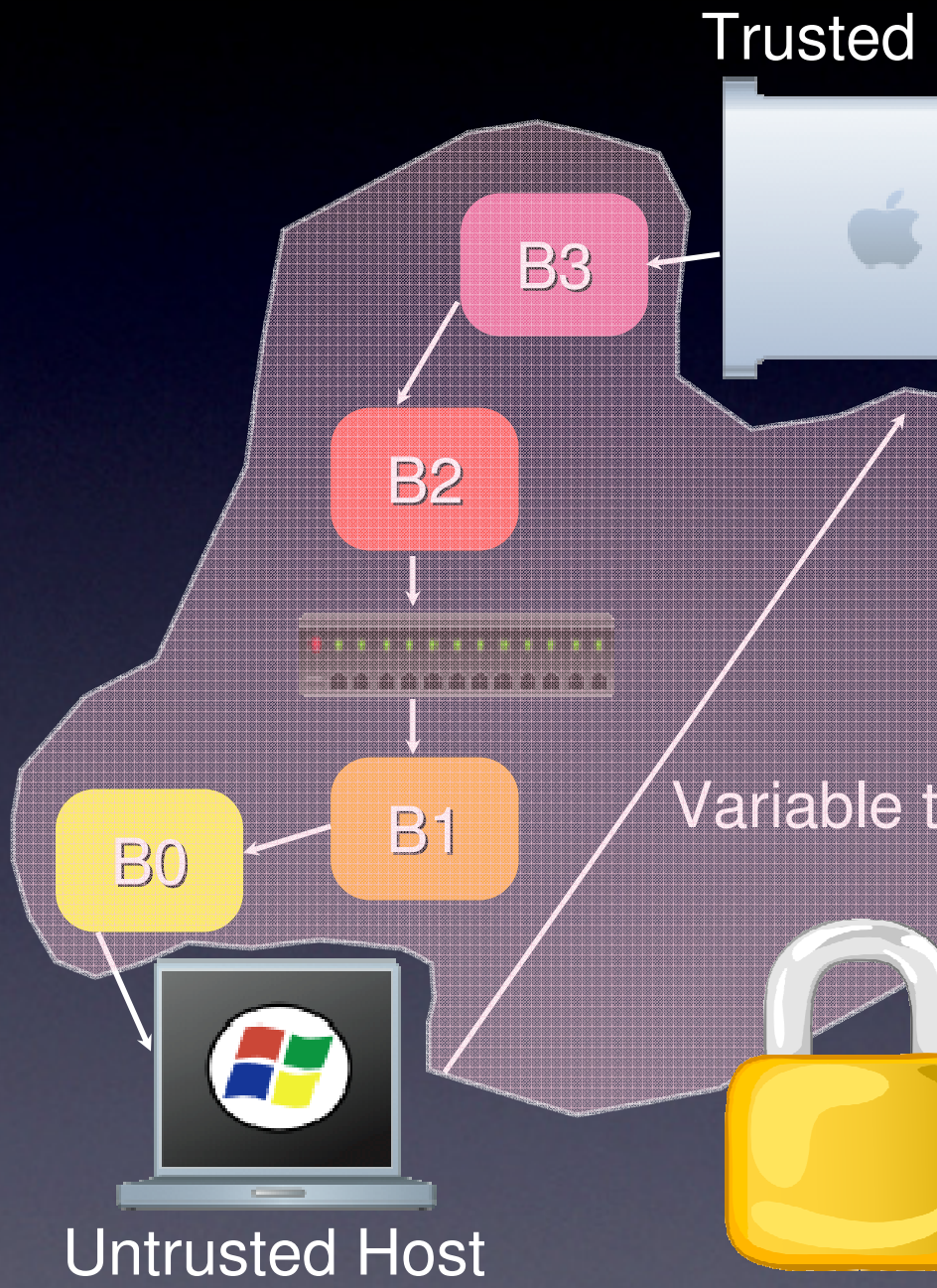


C++ Demonstrator

Re-Trust Protocol

A simple implementation of the RE-TRUST protocol

C++ SECURE SOCKET CLASS



C++ Demonstrator

Untrusted Host

Variable Tracing

Trusted

Integration with SMART DONGLE

Provides additional protection to the code

Not combinable with continuos replacement