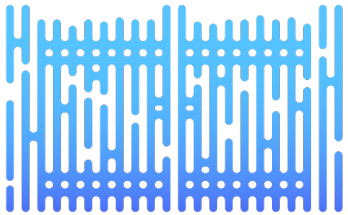


PALISADE

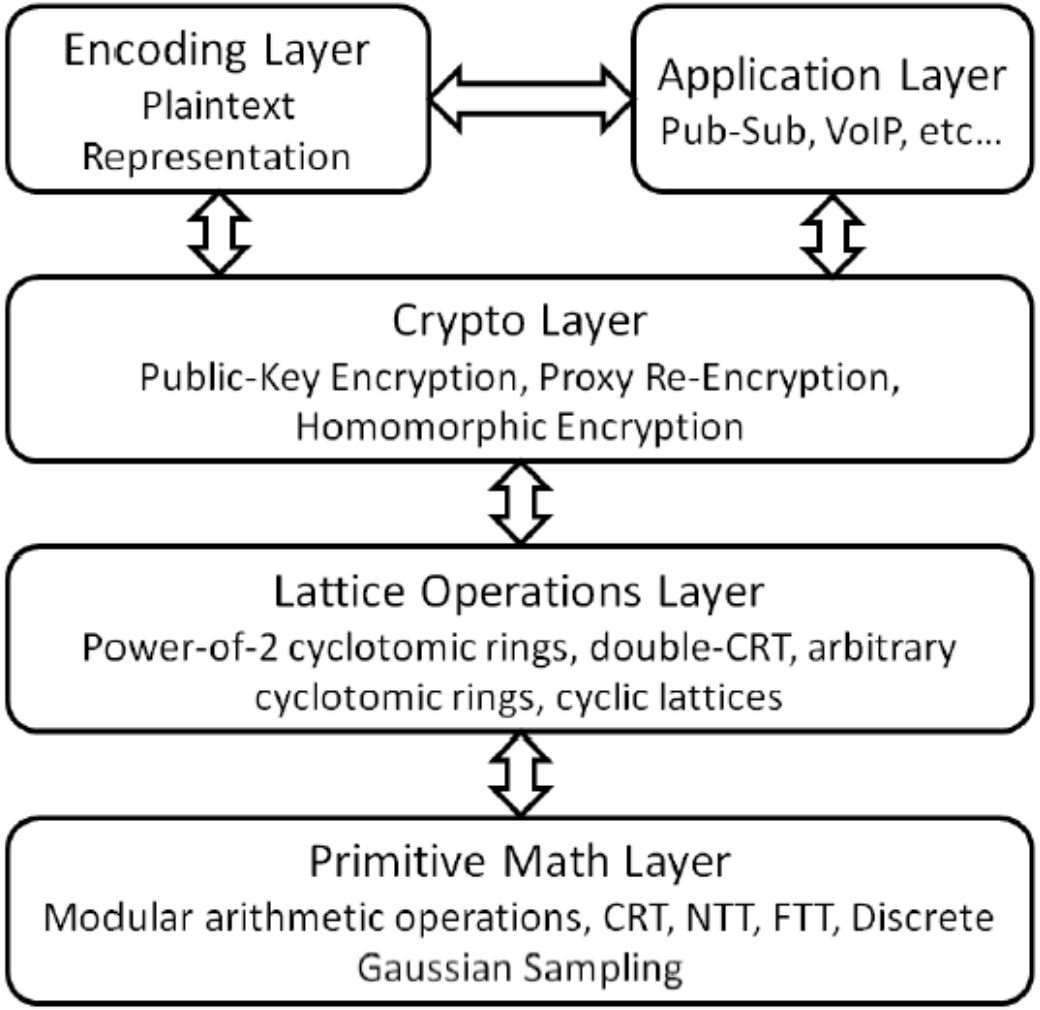
OVERVIEW OF ARCHITECTURE, CAPABILITIES, AND DOCUMENTATION

Yuriy Polyakov
ypolyakov@dualitytech.com

MOTIVATION FOR PALISADE ARCHITECTURE

- Extendible framework and library for homomorphic encryption and lattice cryptography
  - Ex: multiple protocols, schemes and lattice / math back-ends.
  - Low-level plugin optimization can be modularized/“outsourced”
- Develop crypto APIs for application developers
  - API should be scheme-agnostic
- Good software engineering with focus on usability
  - Standards-based design and style
  - Unit tests and benchmarking environment
  - Documentation and sample code

MODULAR/LAYERED ARCHITECTURE



PALISADE

MATHEMATICAL BACKENDS

- Modular vector and integer arithmetic is supported by multiple mathematical "backends"
  - The backends can be switched using compile-level flags
- Multiprecision mathematical backends
  - Fixed-size array of native integers (default)
  - Dynamic-size array of native integers
  - NTL implementation
- Native integer backends
  - 64-bit integers with 128-bit integer support (default)
  - 64-bit integers without 128-bit integer support
  - 32-bit integers with 64-bit integer support

SPECIFICATIONS

- PALISADE is a multi-threaded library written in C++11
- Supported operating systems
  - Linux/Unix
  - Windows (MinGW)
  - macOS
- Supported compilers
  - g++ v6.1 and later
  - clang (llvm) v6.0 and later
- CMake is used for building PALISADE
- PALISADE is distributed under the BSD 2-clause license
- The default install of PALISADE has no external dependencies
  - The users can optionally use GMP/NTL (for a multiprecision math backend) and TCMAlloc (for multi-threaded block allocation) if desired

AVAILABILITY

- PALISADE stable release (https://gitlab.com/palisade/palisade-release)
  - Includes the latest stable release of PALISADE (currently v1.9.2) and prior stable releases
- PALISADE preview release (https://gitlab.com/palisade/palisade-development)
  - Includes the latest preview release of PALISADE (currently v1.10.2)
  - A preview release gets converted to a stable release once all known critical bugs reported by the PALISADE community are fixed
  - The “master” branch also houses experimental (research) capabilities that do not get included in releases
- PALISADE Python3 port (https://gitlab.com/palisade/palisade-python-demo)
  - An example showing how to use PALISADE in Python
- FreeBSD port (https://www.freshports.org/security/palisade)
  - A PALISADE package for FreeBSD users

CURRENT CAPABILITIES

- Fully Homomorphic Encryption (FHE) – all FHE schemes use the parameters suggested in the HomomorphicEncryption.org security standard (https://eprint.iacr.org/2019/939)
  - Brakerski/Fan-Vercauteren (BFV) scheme for integer arithmetic
  - Brakerski-Gentry-Vaikuntanathan (BGV) scheme for integer arithmetic
  - Cheon-Kim-Kim-Song (CKKS) scheme for real-number arithmetic
  - Ducas-Micciancio (FHEW) and Chillotti-Gama-Georgieva-Izabachene (TFHE) schemes for Boolean circuit evaluation
  - Stehle-Steinfeld scheme for limited integer arithmetic
- Multi-Party Extensions of FHE (to support multi-key FHE)
  - Threshold FHE for BGV, BFV, and CKKS schemes
  - Proxy Re-Encryption for BGV, BFV, and CKKS schemes

CURRENT CAPABILITIES (CONT'D)

- Efficient lattice trapdoor toolkit with the following applications
  - Digital signature
  - Identity-based encryption
  - Ciphertext-policy attribute-based encryption
- Experimental (research) capabilities
  - Key-policy attribute-based encryption
  - Program obfuscation

MORE DETAILS ABOUT FHE SCHEMES: BGV, BFV & CKKS

- All three schemes are implemented in full RNS (a.k.a double CRT) for efficiency
  - All known key switching methods are supported, including
    - BV “digit” decomposition
    - hybrid (using an auxiliary RNS basis)
    - GHS (special case of hybrid with a “large” auxiliary modulus)
    - “SEAL” (special case of hybrid with a “small” auxiliary modulus)
- All RNS implementations are designed to be as usable as possible
  - Maintenance operations, e.g., rescaling in CKKS and modulus switching in BGV, are done automatically
  - Same-size small primes are used for RNS, e.g., BGV in PALISADE is as easy to use as BFV
  - Parameters are chosen before the computation, and no dynamic noise estimation is needed
- CKKS in RNS is designed to minimize the approximation error
- Selected ideas were presented in our Simons Institute lattice workshop talk: https://www.youtube.com/watch?v=ZtJc6B7C8Pg
- Further details on the variants of CKKS and BGV implemented in PALISADE to appear in IACR ePrint in August/September

MORE DETAILS ABOUT FHE SCHEMES: FHEW & TFHE

- PALISADE provides an HE-standard-compliant implementation of FHEW and TFHE for arbitrary Boolean circuit evaluation
  - Both use uniform ternary secrets
  - Runtime for FHEW and TFHE based on ternary secrets is roughly the same
  - For ternary secrets, the bootstrapping key is smaller for TFHE
  - Main difference between FHEW and TFHE is in the bootstrapping procedure used
- Current bootstrapping runtime for a 128-bit security setting on a commodity workstation (w/o AVX extensions): ~**90 ms**
- More details on the FHEW and TFHE implementation in PALISADE are presented in https://eprint.iacr.org/2020/086

FUNCTIONALITY COMPARISON MATRIX

Library/ Scheme or Extension	BGV	BFV	CKKS	FHEW	TFHE	Threshold FHE (MP)	Proxy Re-Encryption (MP)
FHEW				✓			
HEAAN/HEAAN-RNS			✓				
HELib	✓		✓				
Lattigo		✓	✓			✓	
PALISADE	✓	✓	✓	✓	✓	✓	✓
SEAL		✓	✓				
TFHE					✓		

Directory Structure

Directory	Description
benchmark	Code for benchmarking PALISADE library components, using the Google Benchmark framework
build	Binaries and build scripts (this folder is created by the user)
doc	Documentation of library components, including doxygen documentation generated by the make process.
src	Library source code. Each subcomponent has four or five subdirectories: include, lib, unittest , examples, and optionally extras
third-party	Code for distributions from third parties (includes NTL/GMP + git submodules for tcmalloc, cereal, google test, and google benchmark)
test	Google unit test code

```
|- benchmark
|- build
|- doc
|- src
|   |- binfhe
|   |   |- include    -- Boolean-circuit FHE header files
|   |   |- lib    -- Boolean-circuit FHE source files
|   |- core
|   |   |- include    -- PALISADE core header files
|   |      |- math
|   |      |- lattice
|   |      |- encoding
|   |      |- util
|   |   |- lib    -- PALISADE core source files
|   |      |- math
|   |      |- lattice
|   |      |- encoding
|   |      |- util
|   |- abe
|   |   |- include    -- Attribute-based encryption crypto layer header files
|   |   |- lib    -- Attribute-based encryption crypto layer source files
|   |- pke
|   |   |- include    -- Homomorphic scheme header files
|   |      |- scheme
|   |         |- bfv -- Original BFV (integ. arithm. - slower than RNS variants)
|   |         |- bfvrns -- HPS RNS variant of BFV scheme (integer arithmetic)
|   |         |- bfvrnsb -- BEHZ RNS variant of BFV scheme (integer arithmetic)
|   |         |- bgv -- BGV scheme (integer arithmetic)
|   |         |- bgvrns -- full RNS variant of BGV scheme (integer arithmetic)
|   |         |- ckks -- CKKS scheme (real-number arithmetic)
|   |         |- null -- NULL scheme (integer arithemtic)
|   |         |- stst -- Stehle-Steinfeld scheme (limited  HE operations)
|   |   |- lib    -- Public key encryption crypto layer source files
|   |      |- scheme
|   |         |- bfv -- Original BFV (integ. arithm. - slower than RNS variants)
|   |         |- bfvrns -- HPS RNS variant of BFV scheme (integer arithmetic)
|   |         |- bfvrnsb -- BEHZ RNS variant of BFV scheme (integer arithmetic)
|   |         |- bgv -- BGV scheme (integer arithmetic)
|   |         |- bgvrns -- full RNS variant of BGV scheme (integer arithmetic)
|   |         |- ckks -- CKKS scheme (real-number arithmetic)
|   |         |- null -- NULL scheme (integer arithemtic)
|   |         |- stst -- Stehle-Steinfeld scheme (limited  HE operations)
|   |- signature
|      |- include    -- Signature crypto layer header files
|      |- lib    -- Signature crypto layer source files
|- test          -- Test software scripts
|- third-party   -- External third-party software
```

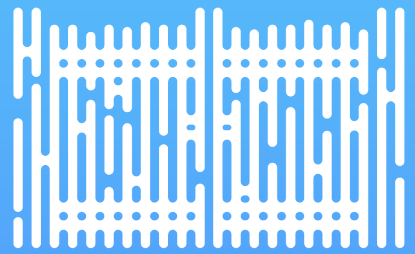
DOCUMENTATION WALKTHROUGH

The Wiki (https://gitlab.com/palisade/palisade-development/-/wikis/home) is the main documentation source providing links for both beginners and advanced PALISADE users

- Getting Started with PALISADE
  - How to build PALISADE and customize it using CMake flags
  - How to include PALISADE in your own projects
  - Code examples for integer and real-number arithmetic, and Boolean circuits
- More advanced documentation
  - PALISADE user manual
  - PALISADE API (generated using doxygen)
  - Release notes
  - Publications describing scheme implementations in PALISADE
- Documentation for PALISADE contributors

HOW TO REQUEST FEATURES OR REPORT BUGS

- We use the Gitlab issue tracking system to track user requests and bugs: https://gitlab.com/palisade/palisade-development/-/issues
  - Please provide as much information as possible when reporting a bug, e.g., the build error console output, runtime error console output, version/commit of PALISADE, environment where PALISADE is run/built.
- Issues are then labeled, e.g., as a “Minor Bug”, and assigned to milestones
- Milestones are used to track issues for specific releases



THANK YOU

contact@palisade-crypto.org