

# Chipwhisperer 분석

# Chipwhisperer

- 아날로그 캡처 하드웨어, 대상 장치, 캡처 소프트웨어 및 분석 소프트웨어를 포함한 부채널 분석 툴 박스
- 오픈 소스로 고도화된 모듈화가 특징

## 1. CW 도식도

패키지 조직도, 클래스 다이어그램

## 2. 클래스 세부분석

코드 문서화

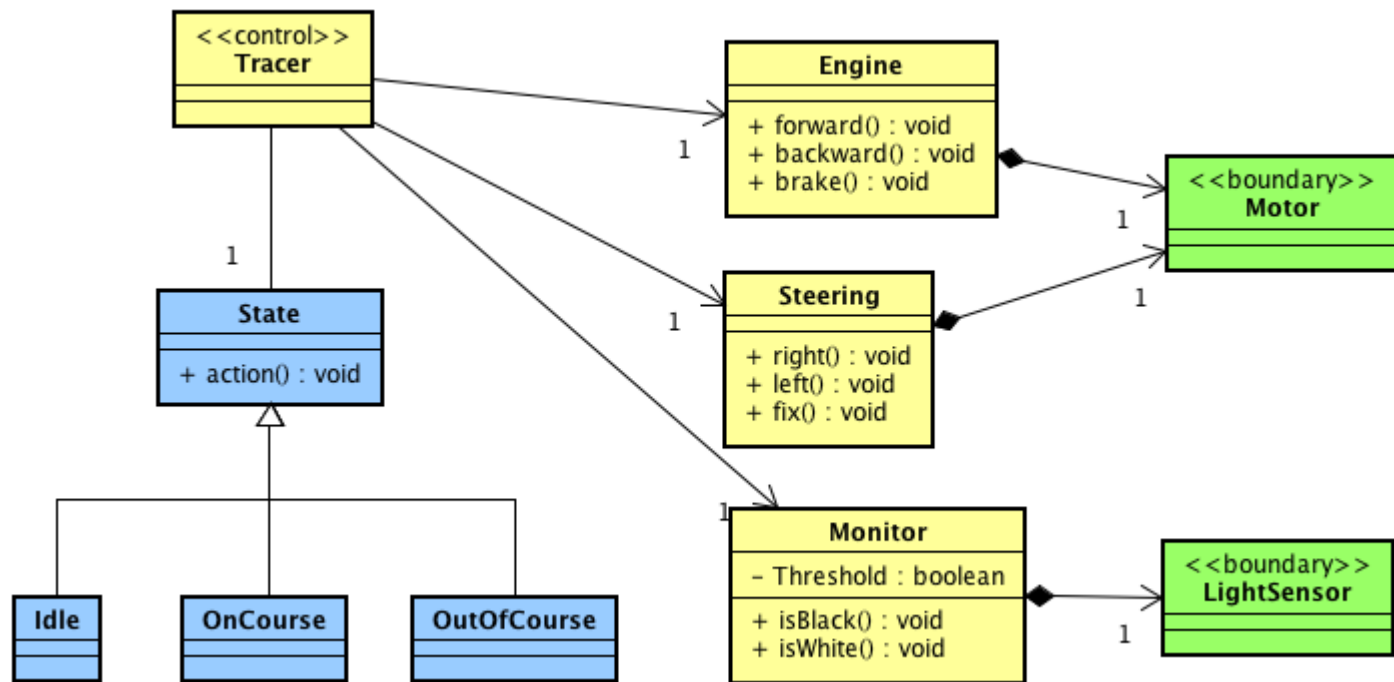
## 3. CW 5.0.1 alpha vs 3.5.4

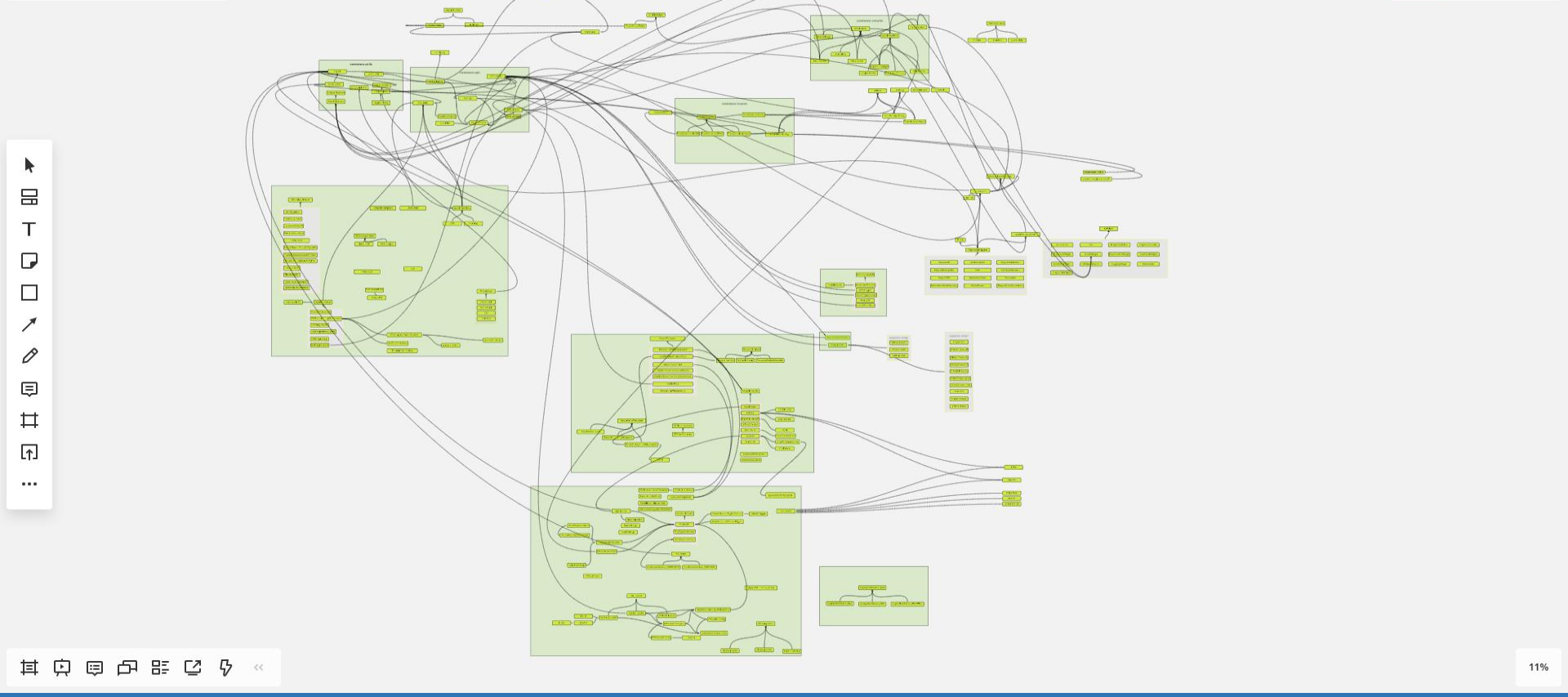
디렉토리와 파일 구조 기반으로 두 버전 간 차이 비교 분석

# 클래스 다이어그램



# 클래스 다이어그램





# 문서화

- 먼저 Epydoc 파이썬 코드 문서화 모듈을 사용하여 문서화

## Epydoc

*Automatic API Documentation Generation for Python*

### Overview

Epydoc is a tool for generating API documentation for Python modules, based on their docstrings. For an example of epydoc's output, see the API documentation for epydoc itself ([html](#), [pdf](#)). A lightweight markup language called [epytext](#) can be used to format docstrings, and to add information about specific fields, such as parameters and instance variables. Epydoc also understands docstrings written in [reStructuredText](#), Javadoc, and plaintext. For a more extensive example of epydoc's output, see the API documentation for [Python 2.5](#).

### Documentation

#### Epydoc manual

- [Installing Epydoc](#)
- [Using Epydoc](#)
- [Python Docstrings](#)
- [The Epytext Markup Language](#)
- [Epydoc Fields](#)
- [reStructuredText and Javadoc](#)
- [Reference Documentation](#)

#### Related Information

- [Open Source License](#)
- [Change Log](#)
- [History](#)
- [Future Directions](#)
- [Related Projects](#)
- [Regression Tests](#)

#### API Documentation

#### Frequently Asked Questions

### Feedback

- [Report a bug](#)
- [Suggest a feature](#)
- Author: [Edward Loper](#)

### Latest Release

The latest stable release is [Epydoc 3.0](#). If you wish to keep up on the latest developments, you can also get epydoc from the [subversion repository](#). See [Installing Epydoc](#) for more information.

### Screenshots



### News

#### Epydoc 3.0 released [January 2008]

Epydoc version 3.0 is now available on the [SourceForge download page](#). See the [What's New](#) page for details. Epydoc is under active development; if you wish to keep up on the latest developments, you can get epydoc from the [subversion repository](#). If you find any bugs, or have suggestions for improving it, please report them on [sourceforge](#).

#### Presentation at PyCon [March 2004]

Epydoc was presented at [PyCon](#) by Edward Loper. [Video and audio from the presentation](#) are available for download.

# Common

- 공통으로 사용되는 매개변수에 대한 기능, 프로젝트 저장 및 가져오기, trace 파일 사용, 데이터 플로팅, '스크립팅' 시스템으로 구성

## api

autoscript

CWCoreAPI

dictdiffer

ProjectFormat

settings

TraceManager

## scripts

base

## results

\_plotdata

attacksettings

base

corrtraceplot

outputvstimeplot

pgevstraceplot

save

scatterplot

table

tracerecorder

Waveform\_widget

## ui

**images**

CWMainGUI

GraphWidget

HelpWindow

KeyScheduleDialog

logger\_widget

ParameterTypes  
Custom

PreferencesDialog

ProgressBar

projectdiffwidget

## traces

\_base

\_cfgfile

TraceContainer  
DPAv3

TraceContainer  
MySQL

TraceContainer  
Native

TraceContainer  
Types

## utils

aes\_cipher

analysissource

parameter

pluginmanager

preprocess\_traces

qt\_tweaks

serialport

timer

Tracereader\_  
dpacontestv3

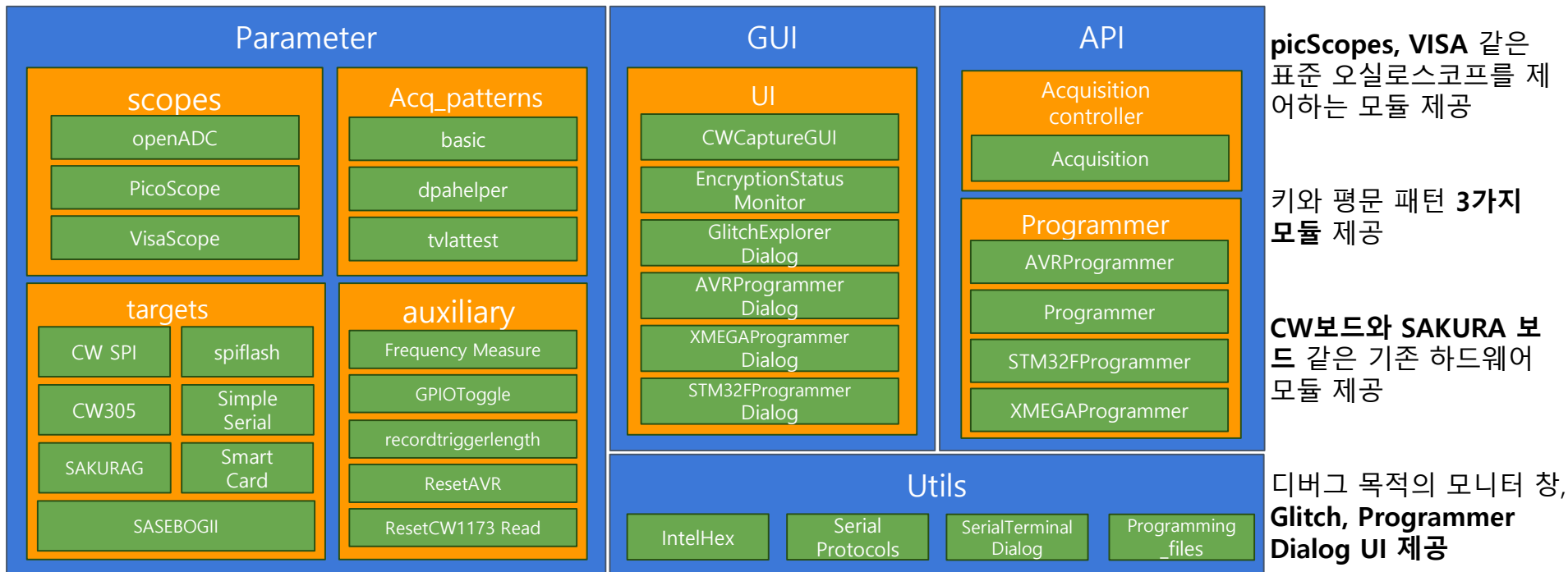
util

기본 저장 방법은 Python  
NumPY 라이브러리의 기본 저  
장 및 로드 명령을 사용

추적을 MySQL 서버에 저장  
DPAContestv3 도구에서 사용  
하는 대안 제공

# Capture 모듈 구성

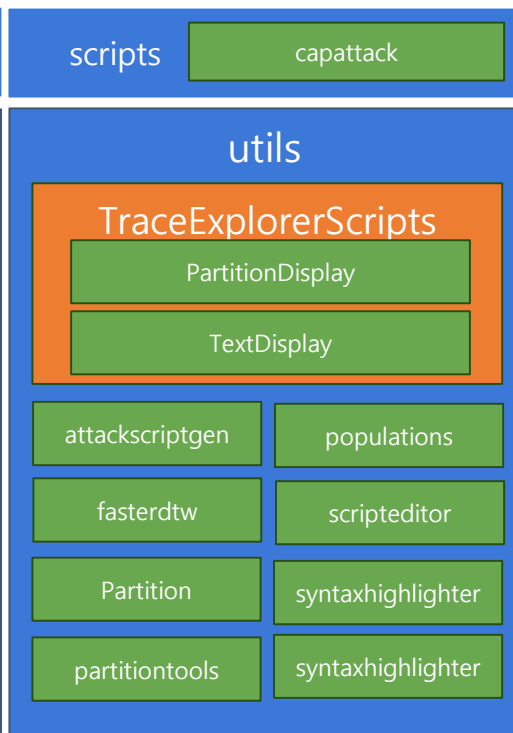
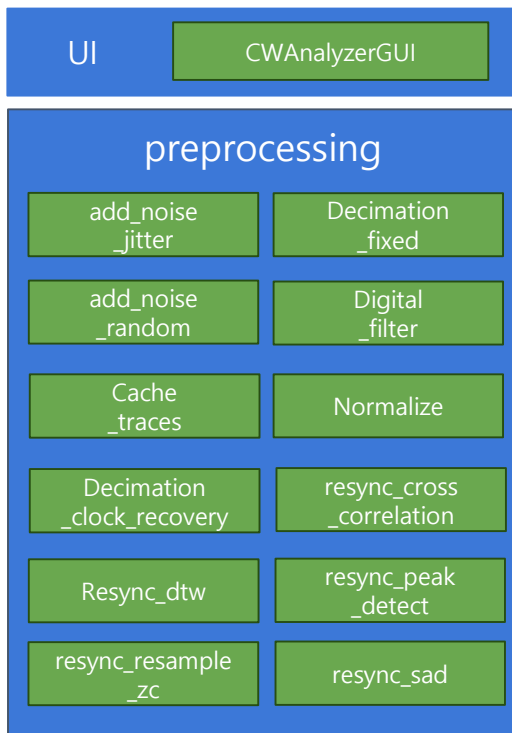
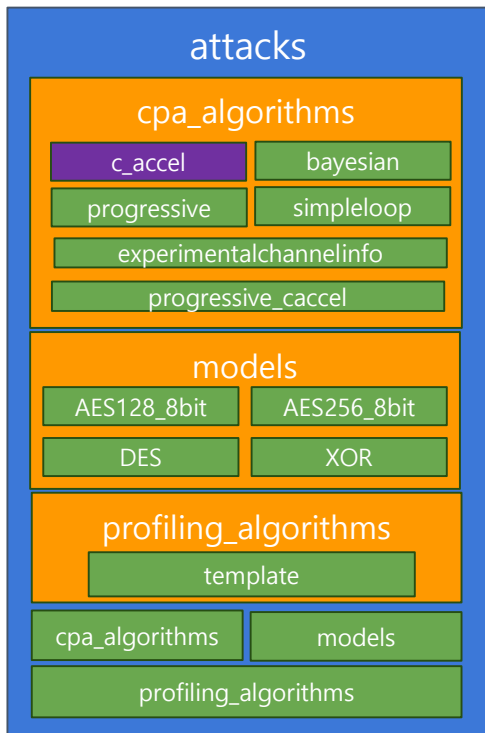
- 매개 변수 값 설정, Programmer와 수집 기능, GUI로 구성





# Analyzer 모듈 구성

- Preprocessing, 공격 알고리즘과 모델, GUI로 구성



공격 전에 작동하는  
몇 가지 기본 사전 처리 모  
듈 제공하며 임의의 순서로  
함께 연결 가능

주로 CPA 공격 모듈과  
profiling 공격 모듈 제공

암호 알고리즘 모델 AES 8-  
bit, DES, XOR 제공

# 모듈 관련 클래스

- Plugin 클래스
  - 모듈을 식별하고 패키지에서 로드할 수 있도록 도와주는 마커 인터페이스(이름만 있는)
- Parameter 클래스
  - 데이터에 대한 접근을 체계적으로 잡아주는 기본 데이터의 단위를 정의한 클래스
  - 고급 위젯을 통해 화면에 출력(QT & PyQtGraph가 지원이 되는 경우)
  - 각 매개변수에는 이름, 유형, 값 및 동작을 수정하는 다양한 속성이 존재함
  - 값은 내부적으로 Parameter에 저장 혹은 set/get 메소드를 사용해 외부적으로 저장 가능
  - Action 메소드는 매개 변수 값이 변경 될 시 호출

지원하는 타입	group, list, label, str, text, bool, action, int, float, range, rangegraph, file, fileList, color, menu
지원하는 속성	name, type, key, values, value, set, get, limits, step, linked, default, tip, action, visible, children, readonly, help, graphwidget, siPrefix, suffix, psync, addLoadSave

- Parameterized 클래스
  - Parameter를 만들기, 가져오기, 찾아오기 기능을 위한 추상 클래스

# 실행 흐름 Capture

```
# User commands here
self.api.setParameter(['Generic Settings', 'Scope Module', 'ChipWhisperer/OpenADC'])
self.api.setParameter(['Generic Settings', 'Target Module', 'Simple Serial'])
self.api.setParameter(['Generic Settings', 'Trace Format', 'ChipWhisperer/Native'])
self.api.setParameter(['Simple Serial', 'Connection', 'NewAE USB (CWLite/CW1200)'])
self.api.setParameter(['ChipWhisperer/OpenADC', 'Connection', 'NewAE USB (CWLite/CW1200)'])
```

```
self.api.connect()
```

```
# Example of using a list to set parameters. Slightly easier to copy/paste in this format
lstexample = [['CW Extra Settings', 'Trigger Pins', 'Target I04 (Trigger Line)', True],
               ['CW Extra Settings', 'Target I0n Pins', 'Target I01', 'Serial RXD'],
               ['CW Extra Settings', 'Target I0n Pins', 'Target I02', 'Serial TXD'],
               ['OpenADC', 'Clock Setup', 'CLKGEN Settings', 'Desired Frequency', 7370000.0],
               ['CW Extra Settings', 'Target HS IO-Out', 'CLKGEN'],
               ['OpenADC', 'Clock Setup', 'ADC Clock', 'Source', 'CLKGEN x4 via DCM'],
               ['OpenADC', 'Trigger Setup', 'Total Samples', 3000],
               ['OpenADC', 'Trigger Setup', 'Offset', 1250],
               ['OpenADC', 'Gain Setting', 'Setting', 45],
               ['OpenADC', 'Trigger Setup', 'Mode', 'rising edge'],
               # Final step: make DCMs relock in case they are lost
               ['OpenADC', 'Clock Setup', 'ADC Clock', 'Reset ADC DCM', None],
               ]
```

```
# Download all hardware setup parameters
for cmd in lstexample: self.api.setParameter(cmd)
```

```
# Let's only do a few traces
```

```
self.api.setParameter(['Generic Settings', 'Acquisition Settings', 'Number of Traces', 50])
```

```
# Capture a set of traces and save the project
```

```
self.api.captureM()
```

```
self.api.saveProject("../.../projects/test.cwp")
```

## Scope와 Target을 선택하는 Parameter 세팅

### 연결

```
def connect(self):
    """Connect both: scope and target"""
    return self.connectScope() and self.connectTarget()
```

```
def connectTarget(self):
    """Connect to the selected target"""
    try:
        if self.getTarget():
            self.getTarget().con(scope=self.getScope())
    except Warning:
        sys.excepthook(*sys.exc_info())
        return False
    return True
```

## 선택된 Scope와 Target에 대한 추가적인 Parameter 세팅

### 트레이스 수 세팅

### 수집 saveProject

# 모듈 추가

- 확장 용이한 모듈  
scopes,  
targets,  
Acq\_patterns(입력 key/text 패턴),  
Preprocessing,  
attack algorithms,  
target algorithms model

# 모듈 추가 간단한 예시

- Plugin 클래스를 상속하고있는 \_base의 클래스를 상속하여 모듈 추가
- 모듈 추가가 GUI에 반영

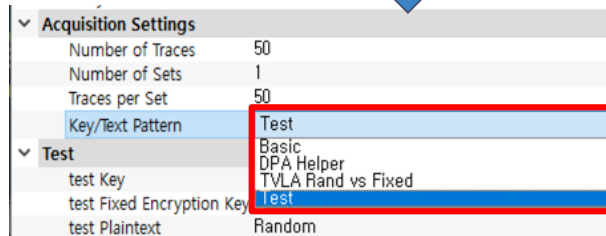
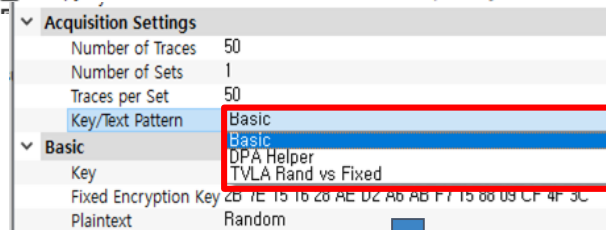
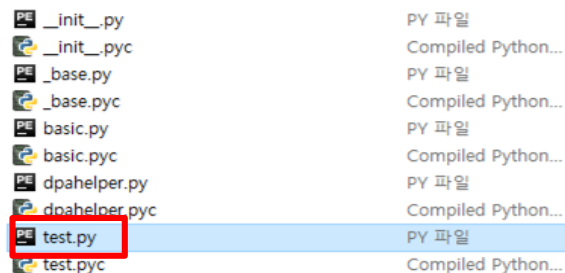
```
from chipwhisperer.common.utils.pluginmanager import Plugin
from chipwhisperer.common.utils.parameter import Parameterized, Parameter
```

```
class AcqKeyTextPattern_Base(Parameterized, Plugin):
    _name = "Key/Text Pattern"

    from chipwhisperer.common.utils import util
    from _base import AcqKeyTextPattern_Base
    from chipwhisperer.common.utils.parameter import setupSetParam
```

```
class AcqKeyTextPattern_test(AcqKeyTextPattern_Base):
    _name = "Test"

    def __init__(self, target=None):
        AcqKeyTextPattern_Base.__init__(self)
```



# 모듈 추가 간단한 예시

```
from chipwhisperer.common.utils.parameter import Parameterized, Parameter
class AcqKeyTextPattern_Base(Parameterized, Plugin):
```

```
from _base import AcqKeyTextPattern_Base
from chipwhisperer.common.utils.parameter import import setupSetParam
```

```
class AcqKeyTextPattern_Basic(AcqKeyTextPattern_Base):
    _name = "Basic"

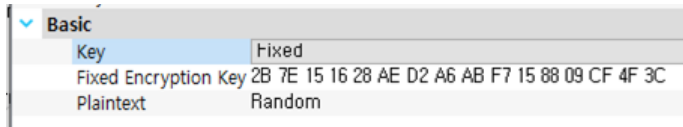
    def __init__(self, target=None):
        AcqKeyTextPattern_Base.__init__(self)

        self.initkey = '2B 7E 15 16 28 AE D2 A6 AB F7 15 88 09 CF 4F 3C'
        self.types = {'test1': False, 'test2': True}
        self.getParams().addChildren([
            {'name': 'test', 'type': 'list', 'values': self.types, 'get': self.getKeyType,
             'set': self.setKeyType, 'action': lambda p: self.findParam("initkey").show(p.getValue())},
            {'linked': ['initkey']}],
        ])
        self.setTarget(target)

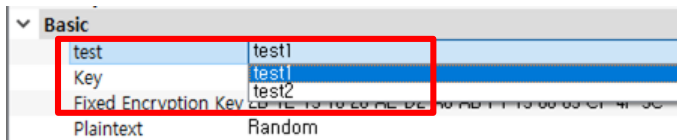
    def getTestType(self):
        return self.initkey

    @setupSetParam("TEST")
    def setKeyType(self, t):
        self.initkey = t
```

1. \_base의 클래스에서 Parameterized 클래스를 상속
2. \_base의 클래스를 확장하여 (다중상속으로 문제가 생길 수 있으므로)
3. \_name 정의
4. self.getParams().addChildren([...]) 호출



Basic	
Key	Fixed
Fixed Encryption Key	2B 7E 15 16 28 AE D2 A6 AB F7 15 88 09 CF 4F 3C
Plaintext	Random



Basic	
test	test1
Key	test1
Fixed Encryption Key	2B 7E 15 16 28 AE D2 A6 AB F7 15 88 09 CF 4F 3C
Plaintext	Random

# 클래스 다이어그램

## Parameterized

software.chipwhisperer.capture.ui.GlitchExplorerDialog.GlitchExplorerDialog  
software.chipwhisperer.common.results.base.ResultsBase  
software.chipwhisperer.capture.scopes.picoscope\_interface.picoscopes.PicoScopeBase  
software.chipwhisperer.common.results.scatterplot.ScatterPlot  
software.chipwhisperer.common.traces.\_base.TraceContainer  
software.chipwhisperer.analyzer.attacks.algorithmsbase.AlgorithmsBase  
software.chipwhisperer.common.utils.tracesource.PassiveTraceObserver  
software.chipwhisperer.analyzer.attacks.models.base.ModelsBase  
software.chipwhisperer.capture.acq\_patterns.\_base.AcqKeyTextPattern\_Base  
software.chipwhisperer.capture.scopes.visascope\_interface.\_base.VisaScope  
software.chipwhisperer.capture.targets.smartcard\_readers.\_base.ReaderTemplate  
software.chipwhisperer.capture.ui.GlitchExplorerDialog.TuningParameter  
software.chipwhisperer.capture.scopes.cwhardware.ChipWhispererGlitch.ChipWhispererGlitch  
software.chipwhisperer.capture.scopes.\_OpenADCInterface.ClockSettings  
software.chipwhisperer.capture.scopes.cwhardware.ChipWhispererLite.CWLiteUSB  
software.chipwhisperer.common.scripts.base.UserScriptBase  
software.chipwhisperer.capture.scopes.cwhardware.ChipWhispererSAD.ChipWhispererSAD  
software.chipwhisperer.capture.scopes.openadc\_interface.oadc\_serial.OpenADCInterface\_Serial  
software.chipwhisperer.analyzer.attacks.\_base.AttackBaseClass  
software.chipwhisperer.capture.scopes.cwhardware.ChipWhispererDecodeTrigger.ChipWhispererDecodeTrigger  
software.chipwhisperer.analyzer.utils.attackscriptgen.AttackScriptGen  
software.chipwhisperer.capture.scopes.\_OpenADCInterface.GainSettings

software.chipwhisperer.capture.scopes.base.ScopeTemplate  
software.chipwhisperer.analyzer.utils.TraceExplorerDialog.TraceExplorerDialog  
software.chipwhisperer.common.api.CWCoreAPI.CWCoreAPI  
software.chipwhisperer.analyzer.utils.TraceExplorerScripts.PartitionDisplay.PartitionDisplay  
software.chipwhisperer.capture.targets.spiflash\_programmers.\_base.SPIFlashTemplate  
software.chipwhisperer.capture.scopes.openadc\_interface.naeusbchip.OpenADCInterface\_NAEUSBChip  
software.chipwhisperer.capture.scopes.\_OpenADCInterface.HWInformation  
software.chipwhisperer.capture.scopes.cwhardware.ChipWhispererDigitalPattern.ChipWhispererDigitalPattern  
software.chipwhisperer.capture.targets.smartcard\_protocols.\_base.ProtocolTemplate  
software.chipwhisperer.capture.scopes.cwhardware.ChipWhispererExtra.CWExtraSettings  
software.chipwhisperer.capture.scopes.\_OpenADCInterface.TriggerSettings  
software.chipwhisperer.analyzer.utils.Partition.Partition  
software.chipwhisperer.analyzer.utils.TraceExplorerScripts.TextDisplay.TextDisplay  
software.chipwhisperer.capture.targets.simpleserial\_readers.\_base.SimpleSerialTemplate  
software.chipwhisperer.capture.scopes.\_qt.OpenADCQt  
software.chipwhisperer.capture.scopes.openadc\_interface.ztex.OpenADCInterface\_ZTEX  
software.chipwhisperer.capture.targets.SAKURAG.ChipWhispererComm  
software.chipwhisperer.capture.scopes.openadc\_interface.ftdi.OpenADCInterface\_FTDI  
software.chipwhisperer.capture.scopes.cwhardware.ChipWhispererExtra.ChipWhispererExtra  
software.chipwhisperer.capture.targets.\_base.TargetTemplate  
software.chipwhisperer.capture.auxiliary.\_base.AuxiliaryTemplate  
software.chipwhisperer.common.ui.PreferencesDialog.GeneralTab

총 44개 클래스

# CW 5.0.1 alpha

- 5.0.1 전체적인 변경 사항
  - front-end 위주의 업데이트
  - 파이썬 2에서 파이썬 3으로 전환
  - GUI와 wiki 튜토리얼이 Jupyter Notebook으로 대체됨  
웹 브라우저에서 파이썬 블록을 실행해 볼 수 있게 됨
  - 파이썬 2.7 특정 코드들을 제외하면 대부분의 API는 4.0 버전과 동일
  - GUI와 관련없는 대부분의 스크립트는 이전 버전과 동일하게 작동
  - Analyzer 지원이 GUI 밖 어디서든지 가능해짐



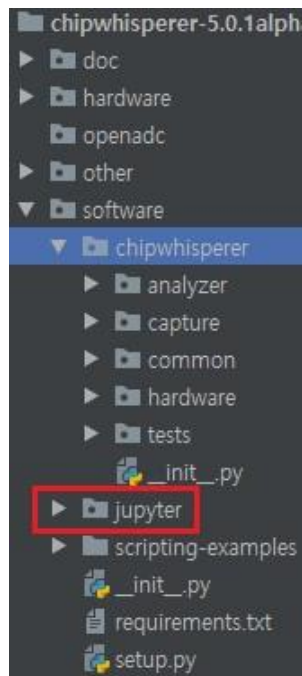
# CW 5.0.1 alpha

- 5.0.1 전체적인 변경 사항
  - 파이썬 API를 이용한 analyzer 및 프로젝트 지원
  - 파이썬 API를 통한 여러 ChipWhisperer 지원 가능
  - 사용자가 Github에 제출한 튜토리얼 이용 가능
  - 캡처 속도 향상

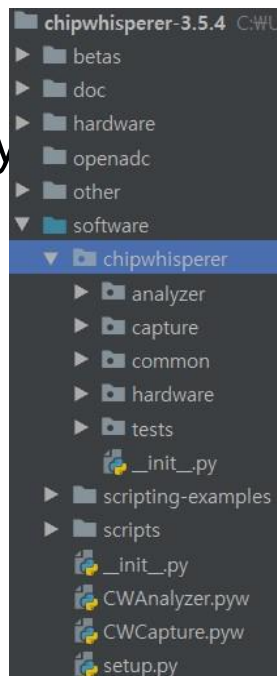
# CW 5.0.1 alpha vs 3.5.4

- /software
- 3.5.4버전에만 존재: CWAnalyzer.pyw
- **5.0.1버전에만 존재: jupyter**
  - CW-5 버전은 UI 요소를 jupyter에 의존

5.0.1 alpha



3.5.4



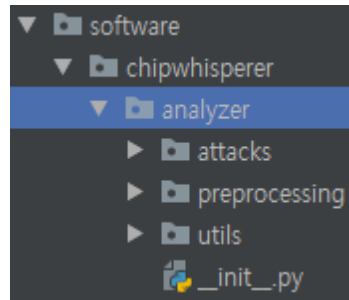
re.py

# CW 5.0.1 alpha vs 3.5.4

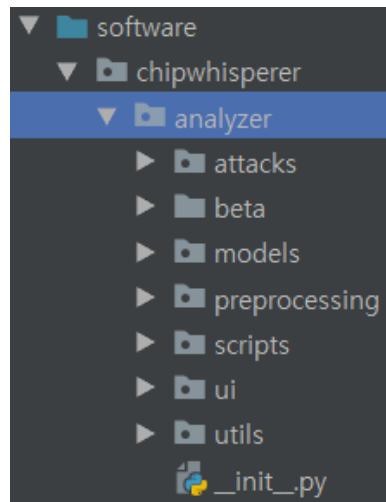
- /chipwhisperer/analyzer
- 3.5.4에만 존재: beta, models, scripts, ui

- 디렉토리 및 파일 구성은 양측 동일
  - 단, utils 디렉토리는 3.5.4의 내용이 풍부
    - 스크립트 편집기, 하이라이터, 파티션 도구, 다이얼로그
    - 파티션 및 텍스트 디스플레이 지원
  - 5.0.1의 utils에서는 UI 지원 요소 배제

5.0.1 alpha



3.5.4

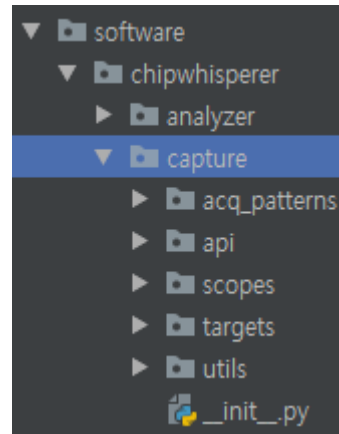


# CW 5.0.1 alpha vs 3.5.4

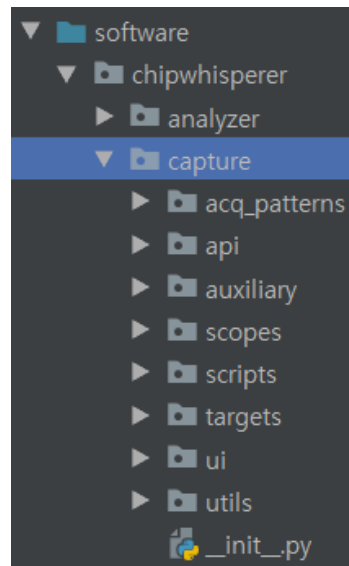
- /chipwhisperer/capture
- 3.5.4에만 존재: auxiliary, scripts, ui,  
/utils/SerialTerminalDialog.py

- **5.0.1에만 존재: /api/aux\_list.py**
  - 3.5.4는 auxiliary에서 관리

5.0.1 alpha



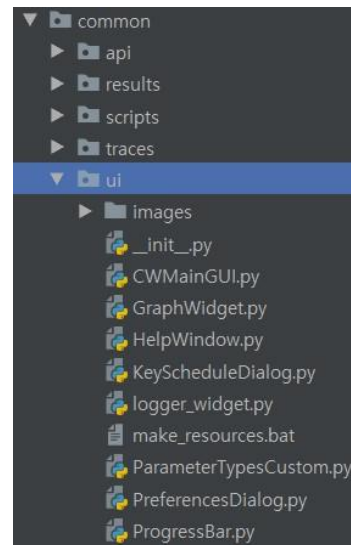
3.5.4



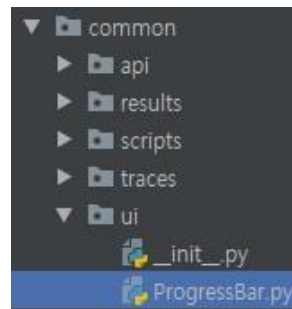
# CW 5.0.1 alpha vs 3.5.4

- /chipwhisperer/common
- 디렉토리 및 파일 구성은 양측 동일
  - 단, results 디렉토리는 예외
    - 3.5.4의 파일 구성이 풍부함
- 5.0.1에서 ui 디렉토리는 common에만 존재
  - 파일 수: **5.0.1 2개 vs 3.5.4 20개**
    - images 디렉토리의 구성요소는 제외한 수치
  - 제공하는 기능이 기존보다 현저히 줄어들음
  - UI 요소를 jupyter에 의존하기 때문으로 추정

3.5.4



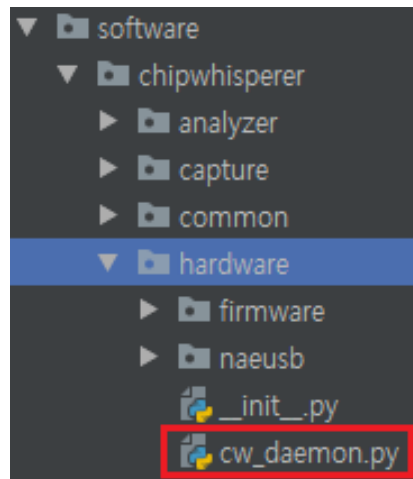
5.0.1 alpha



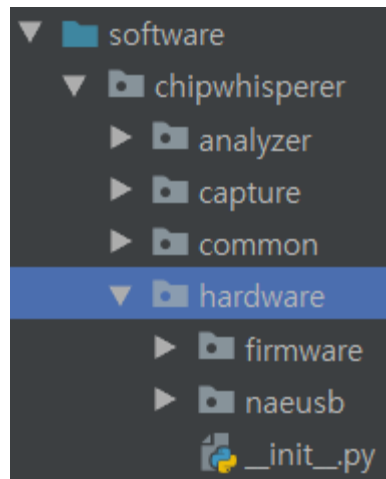
# CW 5.0.1 alpha vs 3.5.4

- /chipwhisperer/hardware
- **5.0.1에만 존재: cw\_daemon.py**
  - 서버와 클라이언트 간 통신
  - jupyter 지원을 위한 것으로 추정
- 그 외 모든 파일 구성 동일

5.0.1 alpha



3.5.4



# CW 5.0.1 alpha vs 3.5.4

- 5.0.1 버전과 3.5.4 버전의 파일 수는 약 2.55배 차이
- 단, 5.0.1은 아직 개발중임을 감안해야 함


5.0.1 alpha


<

3.5.4

chipwhisperer-5.0.1alpha 속성

chipwhisperer-3.5.4 속성

일반	공유	보안	이전 버전	사용자 지정
				
종류:	파일 폴더(.1alpha)			
위치:				
크기:	<u>207MB (217,378,045 바이트)</u>			
디스크 할당 크기:	215MB (226,361,344 바이트)			
내용:	파일 4,871, 폴더 912			

일반	공유	보안	이전 버전	사용자 지정
				
종류:	파일 폴더(.4)			
위치:				
크기:	<u>529MB (555,171,320 바이트)</u>			
디스크 할당 크기:	550MB (576,950,272 바이트)			
내용:	파일 12,462, 폴더 1,393			

# CW 5.0.1 alpha vs 3.5.4

- 결론

- UI 지원에 대한 단순화 작업
- UI 관련 파일들 Jupyter로 통합
- auxiliary 관련 파일을 하나의 파일로 종합
- Jupyter 지원을 위한 파일들 추가
- 여러가지 파일들에 대한 통합으로 인해 파일 크기 감소