# PjSIP integration

## How to add Zorg to PJSIP

First, you need:

- PJSIP 1.10. To avoid issues with line endings, get the UNIX (.tar.bz2) package
- the zorg 1.0 for pjsip 1.10 patch, from this site
- the `patch` command-line tool; we used GNU patch 2.5.8

Then:

1. extract the PJSIP package (e.g. `pjproject-1.10.tar.bz2`) to the default directory (e.g. `pjproject-1.10/`)
2. extract the zorg package (e.g. `pjsip-1.10+zorg-1.0.tar.bz2`) to the default directory (e.g. `pjsip-1.10+zorg-1.0/`)
3. copy the `zorg` subdirectory of the zorg package (e.g. `pjsip-1.10+zorg-1.0/zorg`) as `third_party/zorg` under the PJSIP directory (e.g. p `jproject-1.10/third_party/zorg`)
4. apply the `zorg.diff` patch from the zorg package to the PJSIP directory: just run the `patch -p1 --binary < path to zorg.diff` comma nd line from the PJSIP directory.

This should be enough! To actually enable ZRTP, remember to add the following line to your `pj/config_site.h` before building:
`#define PJMEDIA_HAS_ZRTP 1`

### Requirements

#### Windows build

Visual Studio 2005 or later is required to build PJSIP with Zorg.

#### Symbian build

The S60 SDK (either 3rd or 5th edition) and the OpenC/C++ plugin are required to build PJSIP with Zorg. OpenC/C++ is only required to *build* PJSIP with Zorg, **Zorg doesn't actually depend on the OpenC/C++ runtime**.
The OpenC/C++ SDK plugin has a bug: manually patch the SDK as described in the Open C incorrectly implements offsetof knowledge base article.

### Known issues

The patch includes an update to libsrtp; zrtp.org might work with an older version, but it hasn't been tested.
The Windows build environment only supports Visual Studio 2005 and later
The UNIX build environments is currently unsupported.
The Symbian build environment is broken by default, and needs to be patched by hand; see the Symbian build section above for more information.

## Description of the changes

### Build system

The integration adds the following new build-time configuration macros:

- `PJMEDIA_HAS_ZRTP`: define to `1` to enable the ZRTP features described in this section, or to `0` to disable them. Defaults to `0`.
- `PJMEDIA_ZRTP_MASQUERADE`: define to `1` to enable ZRTP masquerading, which masquerades ZRTP packets as media packets, to complete ZRTP handshakes even through media relays which discard ZRTP packets; define to `0` to disable masquerading. Defaults to `1`.

### PJMEDIA

The integration adds a new media (RTP) transport to PJMEDIA, which will automatically try to enable ZRTP security on every stream (unless disabled).

### PJSUA-LIB

All features of the ZRTP transport are available from PJSUA-LIB, either as extensions to the PJSUA call object or as new global functions.

### PJSUA

#### Command line options

New per-account command line options:

```
  --zrtp-use=N        Use ZRTP? 0:disabled, 1:optional, 2:mandatory (def:1)
  --zrtp-id=IN        ZRTP ZID (24 hexadecimal characters, or '*' for random)
  --zrtp-flags=FLAGS  ZRTP flags: (def:cSdO)
                      C/c enable/disable allowclear
                      S/s enable/disable autosecure
                      D/d enable/disable disclose_bit
                      O/o enable/disable discovery_optimization
  --zrtp-cachettl=S   Set TTL of cached ZRTP secrets to S seconds (def:30 days)
  --zrtp-masquerade=N Masquerade ZRTP messages as media packets? 0:no, 1:yes, 2:always (def:0)
```

New global command line options:

```
  --zrtp-global-id=IN Default ZRTP ZID (24 hexadecimal characters, or '*' for random)
  --zrtp-clientid=ID  ZRTP peer id
  --zrtp-cache=FILE   ZRTP secrets cache file
```

Here's what they do:

- `--zrtp-use`: disable, enable or force ZRTP for all calls made by this account
- `--zrtp-id`: override the global ZID for this account
- `--zrtp-flags`: enable or disable the following ZRTP features for this account:
    - C/c: enable or disable `allowclear`; this does nothing, `GoClear` is unimplemented in Zorg. Default: c (disabled)
    - S/s: enable or disable `autosecure`; if enabled, Zorg will actively enter secure mode; if disabled, it will wait for the peer first. Default: S (enabled)
    - D/d: enable or disable `disclose_bit`; enable if you want to pretend that you disclose encryption keys to third parties (our patch doesn't let you to). Default: d (disabled)
    - O/o: enable or disable `discovery_optimization`; enable for a shorter, faster discovery phase (i.e. finding out if the other peer supports ZRTP). Default: O (enabled)
- `--zrtp-cachettl`: ses the TTL (time-to-live) for ZRTP secrets cached by this account to S seconds.
- `--zrtp-masquerade`: disable, enable or force ZRTP masquerading. The values:
    - `0`: do not masquerade ZRTP messages: ZRTP calls will work with any client, but may not work through certain media relays which drop non-RTP traffic
    - `1`: send both standard and masqueraded ZRTP messages: ZRTP calls will work with any client and any media relay
    - `2`: send masqueraded ZRTP messages only: ZRTP calls will only work with another client that supports masquerading
- `--zrtp-global-id`: sets the default ZID for all accounts that don't have one
- `--zrtp-clientid`: sets the ZRTP client id (i.e. user-agent)
- `--zrtp-cache`: sets the name of the ZRTP secrets cache file

### Run-time commands

New commands available while PJSUA is running:

```
| z  Toggle ZRTP secure/clear mode     ZV Set ZRTP SAS Verified flag         |
| Z  Set ZRTP secure/clear mode        Zv Unset ZRTP SAS Verified flag       |
| mz Make ZRTP call                    ms Make SRTP call                     |
```

And here's what they do:

- `z`: toggle between ZRTP-secure and clear mode on the current call, if `allowclear` is enabled; currently unimplemented
- `Z`: interactively set the mode of the current call (ZRTP-secure/clear)
- `ZV`: sets the SAS Verified flag for the current call; the flag will persist for future calls with the same peer, provided the ZIDs stay the same *and* that the peer sets the flag as well
- `Zv`: resets the SAS Verified flag for the current call; the flag will stay reset for future calls with the same peer, provided the ZIDs stay the same
- `mz`: make a call using ZRTP security, disabling SRTP (if present)
- `ms`: make a call using SRTP security, disabling ZRTP (if present)