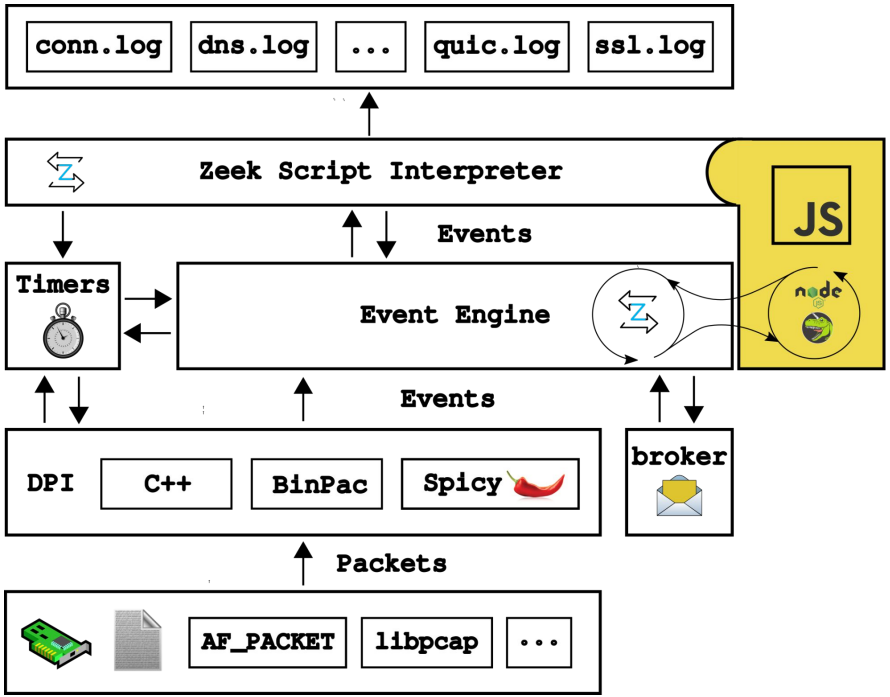
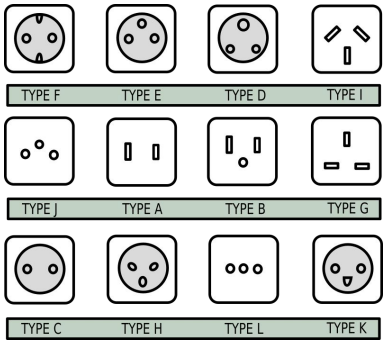
Extending Zeek through JavaScript

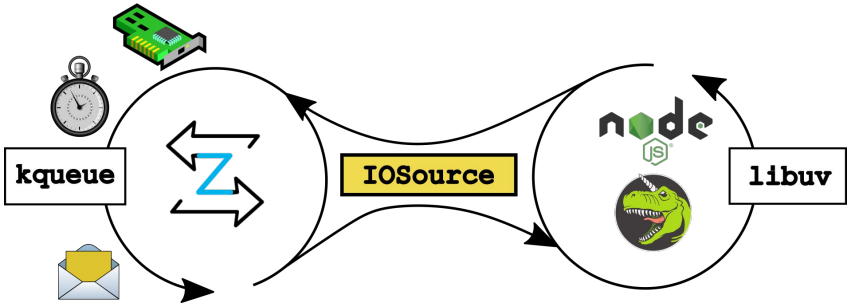
Arne Welzel

corelight
OPEN SOURCE

ZeekJS: Zeek and JavaScript







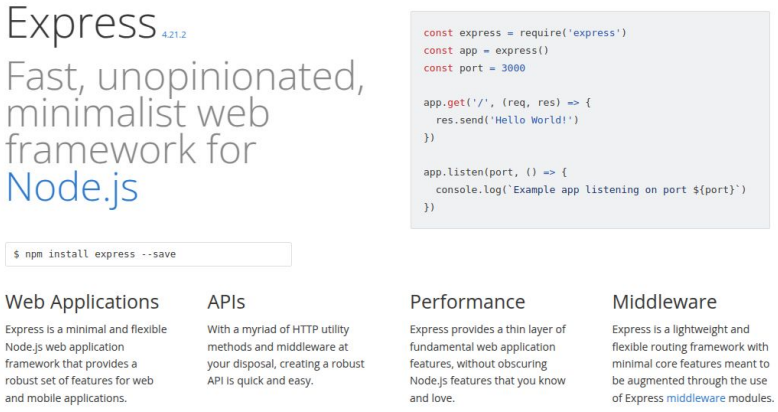
https://archive.fosdem.org/2024/schedule/event/fosdem-2024-2388-zeekjs-javascript-support-in-zeek/

Intro: File Analysis in Zeek

- `Input::add_analysis([$source=file_path, $name=name])`
- Allows to analyze files stored on disk using Zeek's file analyzers
- files.log, pe.log, png.log, …

Intro: File Analysis API

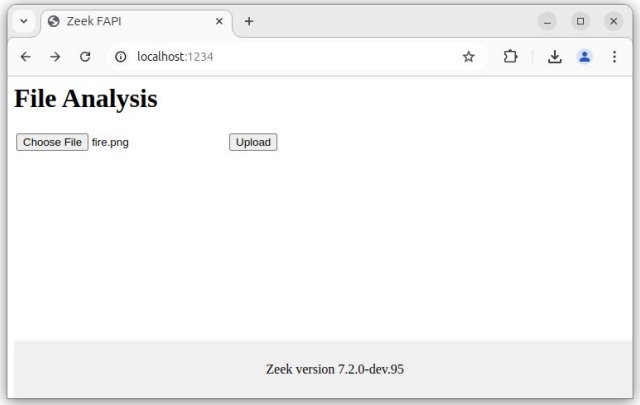
```
$ curl -v -F file=@logo.png http://localhost:1234/upload
{
  "job_id": "JnPc2fJUhkul",
  "logs": {
    "PNG::LOG": [
      {
        "ts": 1740078089.723066,
        "id": "Fz9MuudSefFZR5mh4",
        "chunks": [
          "IHDR",
          "sBIT",
          "pHYs",
          "tEXt",
          "IDAT",
          "IEND"
        ],
        "width": "275",
        "height": "91",
        "colour_type": "truecolour with alpha",
        "bit_depth": "8",
        "interlaced": false
      }
    ],
...
```



https://expressjs.com

Express API Skeleton

```
// npm install expressjs express-fileupload pug
const express = require("express");
const fileUpload = require("express-fileupload");

const app = express();
app.use(fileUpload());

app.get("/", (req, res) => {
  // ...
});

app.post("/upload", (req, res) => {
  // ...
});

app.listen(1234, "127.0.0.1", () => {
  console.log("Server is running");
});
```



The global `zeek` Object

- Register event and hook handlers
- Queue events, invoke hooks and functions
- Record field selection (&log)
- Access to Zeek-script variables
- Explicit type conversion for any

```
// hello.js
zeek.on("zeek_init", () => {
 console.log("Hello Zeek!", zeek.invoke("zeek_version"));
});

$ zeek hello.js
Hello Zeek! 7.0.5
```

Access to Zeek side variables - `zeek.global_vars`

```
# api.zeek
module FAPI;

export {
   const listen_host: string = "127.0.0.1" &redef;
   const listen_port: port = 1234/tcp &redef;
   const work_dir: string = "./work" &redef;
}
```

```
// http.js
const host = zeek.global_vars["FAPI::listen_host"];
const port = zeek.global_vars["FAPI::listen_port"].port;
const work_dir = zeek.global_vars["FAPI::work_dir"];
...
app.listen(port, host, () => {
   console.log(`Server is running on http://${host}:${port}`);
});
```

HTTP POST Handler

```
app.post("/upload", (req, res) => {

  let job_id = zeek.invoke("unique_id", ["J"]);
  let job_dir = work_dir + "/" + job_id;
  let file_path = job_dir + "/file";
  let file_name = req.files.file.name;

  fs.mkdirSync(job_dir);

  req.files.file.mv(file_path, (err) => {
    let j = new job.Job(res, job_id, job_dir, file_path, file_name);
    job.queueJob(j);
  });
});
```

Invoking Zeek functions - `zeek.invoke()`

```
exports.queueJob = (j) => {
  // ...
 let analysis_desc = {
   source: j.file_path,
   name: j.id,
 };
 zeek.invoke("Input::add_analysis", [analysis_desc]);
 return true;
};

exports.deleteJob = (j) => {
 console.log(`JS: ${j.id} - deleting ${j.job_dir}`);
 fs.rmSync(j.job_dir, { recursive: true, force: true });

 // Remove the input stream, else we leak FDs and memory.
 zeek.invoke("Input::remove", [j.id]);
};
```

```
type AnalysisDescription: record {
   source: string;
   reader: Reader &default=...;
   mode: Mode &default=default_mode;
   name: string;
   config: table[string] of string ...
};
```

Handling Events and setting Fields - `zeek.on()`

```
# api.zeek
redef record Files::Info += {
   job_id: string &log &optional;
};
```

```
// handlers.js
zeek.on("file_new", (f) => {
  let j = job.getCurrent();
  f.info.job_id = j.id;

  if (f.source == j.id)
    f.info.filename = j.file_name;

  j.fileNew(f.id);
});
```

```
// Always add the SHA1 and SHA256 analyzers
zeek.on("file_new", (f) => {
  zeek.invoke("Files::add_analyzer",
              [f, "Files::ANALYZER_SHA1"]);
  zeek.invoke("Files::add_analyzer",
              [f, "Files::ANALYZER_SHA256"]);

  // Idea: Register user specified analyzers
  //       from HTTP request
});
```

Type Conversions

	JS
string	string
addr and subnet	string
time and interval	number
int / double	number
count	bigint
record	object wrapping Zeek record value
table	object wrapping Zeek table value
…	…

Registering hook handlers - `zeek.hook()`

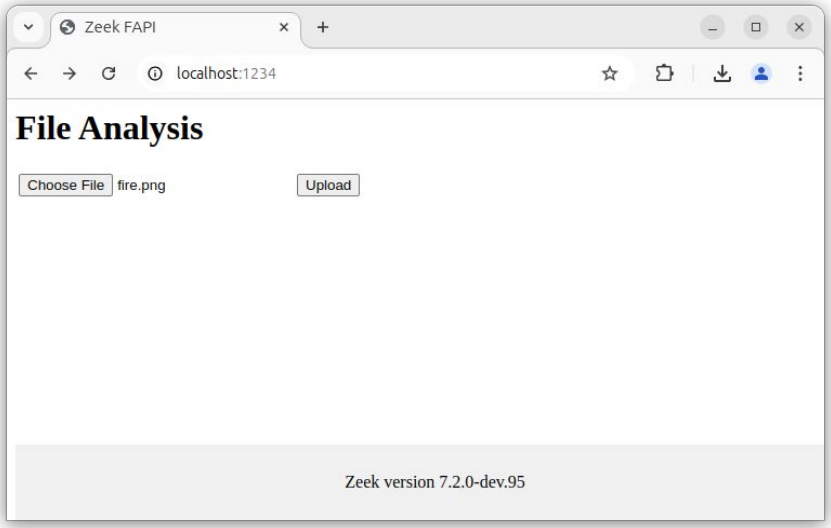
```
zeek.hook("Log::log_stream_policy", (rec, log_id) => {
  let j = job.getCurrent();

  // Ignore logs outside of file processing
  if (!j) return;

  let entry = zeek.flatten(zeek.select_fields(rec, zeek.ATTR_LOG));
  j.addLog(log_id, entry);

  // Return false is like break in hooks: skip logging.
  return false;
});
```

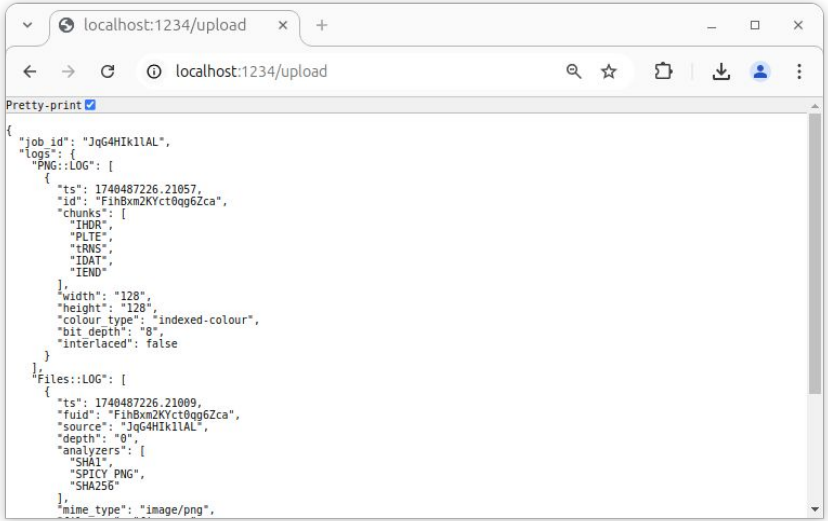
HTTP response

```
zeek.on("file_state_remove", { priority: -2000 }, (f) => {
  let j = job.getCurrent();
  console.log(`JS: ${j.id} file_state_remove ${f.id} ${f.source}`);

  if (j.hasActiveFiles()) return;

  j.res.status(200).json({
    job_id: j.id,
    logs: j.logs,
  });

  job.deleteJob(j);
  job.nextJob();
});
```

Demo

https://github.com/awelzel/zeekjs-file-analysis





Experimental and Sharp Edges

- Spicy and V8 issues due to fiber
  - Alternative stack is troublesome for V8
  - https://github.com/zeek/zeek/issues/4239
- Naive type conversions and APIs
- Performance
- …but potentially very powerful!

Conclusion

- Use JavaScript to embed a HTTP API within Zeek directly
- Invoking Zeek functions
- Handling events and hooks
- Setting fields on records

zeek.JS

Questions