An Undocumented API is a Useless API

API's are interfaces in which the implementations are abstracted, making documentation vital for proper usage. But how do you find accurate documentation for every API? And how do you maintain documentation after updates?

TADAPI is a way for developers to:
- Automatically discover undocumented APIs
- Automatically outline their own APIs
- Easily update API documentation
- Analyze network traffic

TADAPI Automates Documentation

TADAPI discovers API calls by analyzing the network traffic when REST calls are made as a program is in use. The hostname, base URI, and parameters are then extracted from the POST and GET requests, and formatted in a dynamically generated HTML file that is intuitive and deliverable. What better way to produce documentation than to simply use the application?

**Current Methods**
- Manually written during development
- Manually updated after API changes
- Google Search

**TADAPI**
- Creates, finds, and updates documentation for APIs simply by using the application you want to document or discover.

Not Just for APIs

TADAPI was developed modularly so that users can create processors for their own specific purposes. TADAPI will capture the network traffic, and format it. Simply drag and drop the custom processors into the processor folder in order to incorporate it with the application. With processor modularity, the potential uses of TADAPI are limitless.

**Potential uses:**
- Security
- Application Optimization
- Analytics, etc.

**Included Processors:**
- Documentation
- DLP (Data Loss Prevention)

---

1. Starting the MANAGER creates a SNIFFER object which creates a PCAP object.
2. The SNIFFER scrapes network traffic as traffic is generated from application use.
3. The PCAP stores the network traffic scraped by the SNIFFER.
4. The PROCESSOR parses the PCAP file (figure 1), looking for GETS, POSTS, etc. and formats the data into a hash of URLs, parameters, contents, etc. to be used as an input into the specified processor.
5. The selected processor dynamically generates an HTML file (figure 2) from the given hash.