Gopher & Search Engines

GOPHER

- The **Gopher protocol** is a TCP/IP application layer protocol designed for distributing, searching, and retrieving documents over the Internet. The Gopher protocol was strongly oriented towards a menu-document design and presented an alternative to the World Wide Web in its early stages, but ultimately HTTP became the dominant protocol. The Gopher ecosystem is often regarded as the effective predecessor of the World Wide Web.
- Works on port 70.

- It offers some features not natively supported by the Web and imposes a much stronger hierarchy on information stored on it. Its text menu interface is well-suited to computing environments that rely heavily on remote text-oriented computer terminals, which were still common at the time of its creation in 1991, and the simplicity of its protocol facilitated a wide variety of client implementations.

Goals

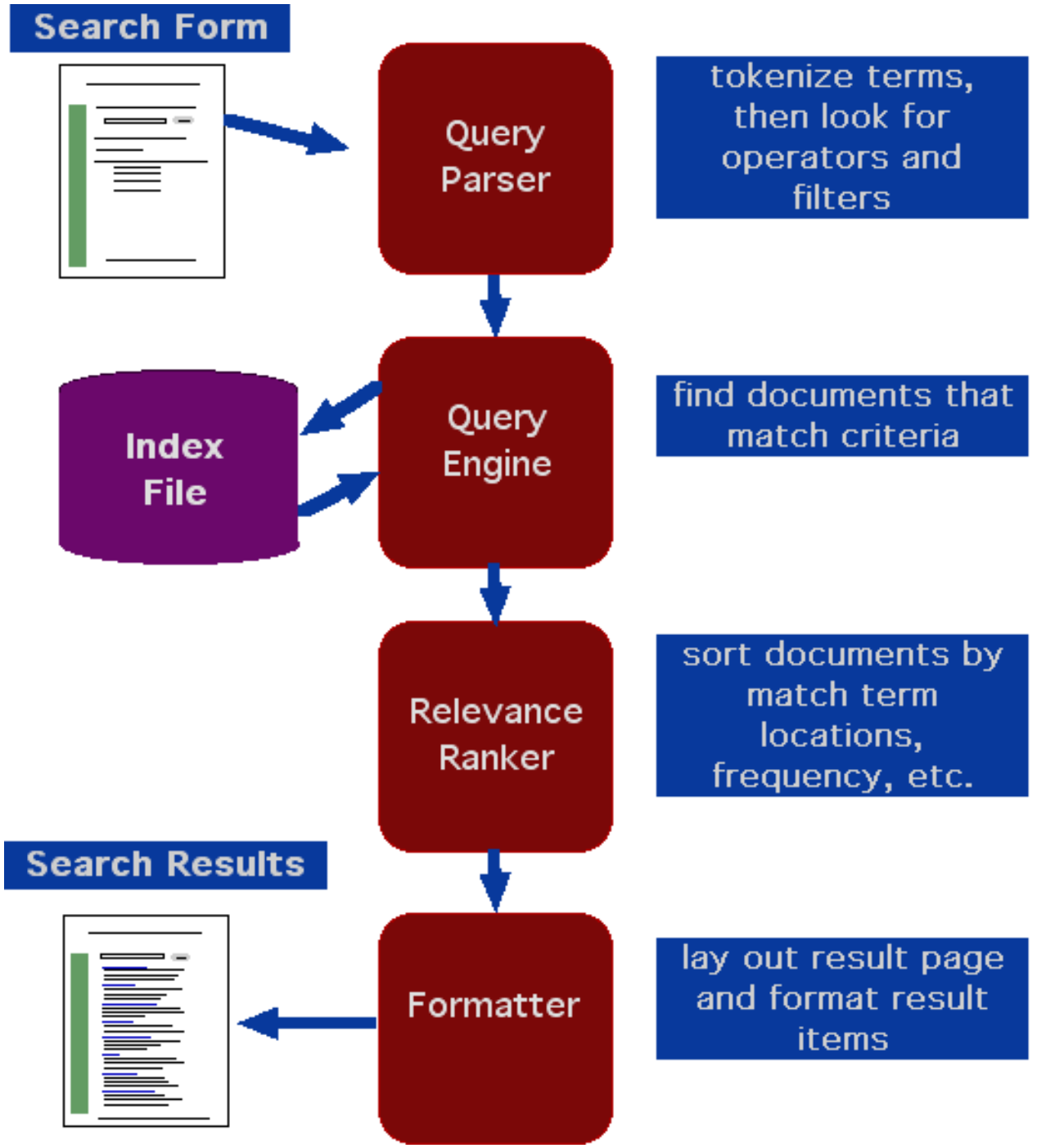
- Its central goals were, as stated in RFC 1436
  - A file-like hierarchical arrangement that would be familiar to users.
  - A simple syntax.
  - A system that can be created quickly and inexpensively.
  - Extending the file system metaphor, such as searches

Search Engines

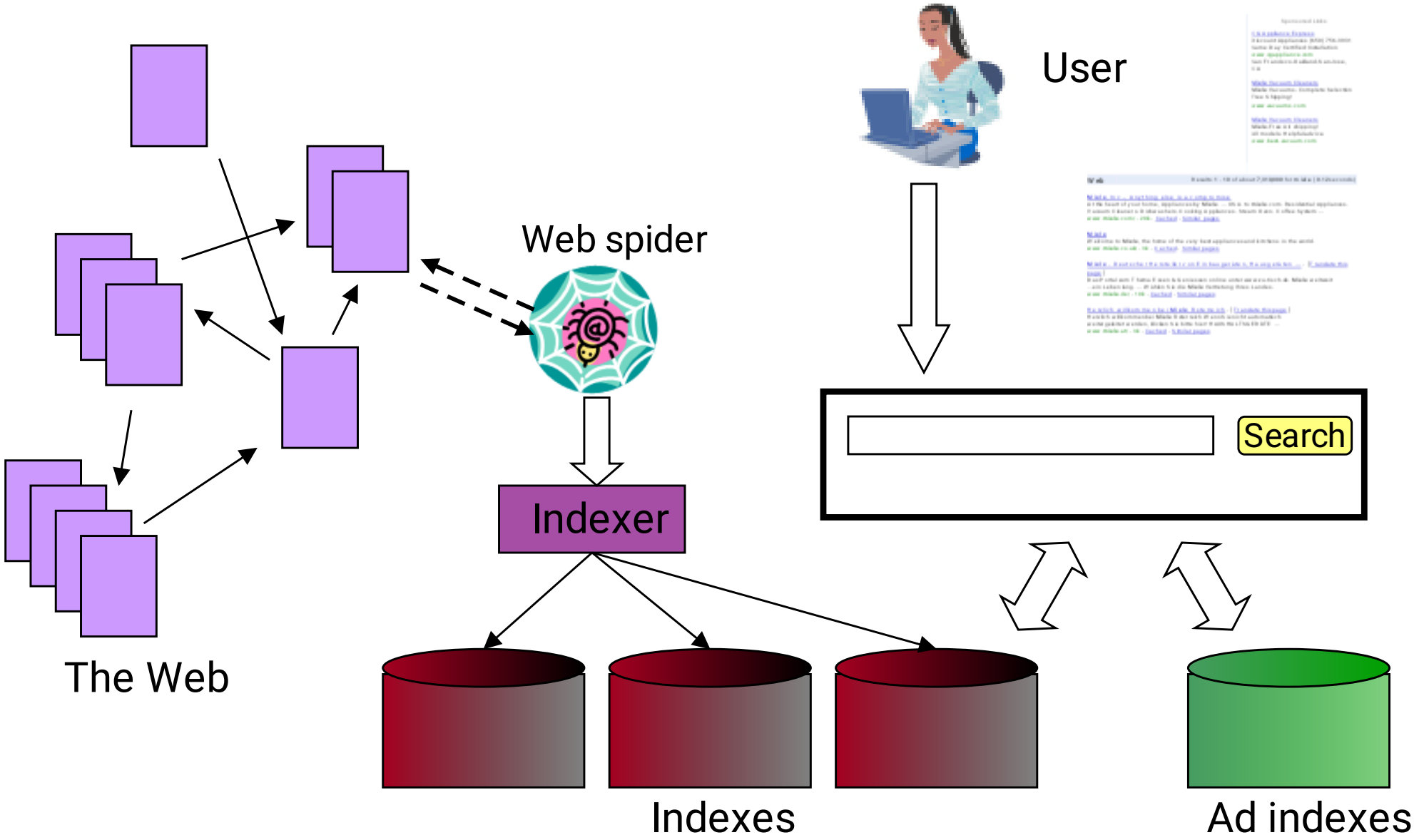
- A **web search engine** is a software system that is designed to search for information on the World Wide Web. The search results are generally presented in a line of results often referred to as search engine results pages (SERPs). The information may be a mix of web pages, images, and other types of files. Some search engines also mine data available in databases or open directories. Unlike web directories, which are maintained only by human editors, search engines also maintain real-time information by running an algorithm on a web crawler.

How web search engines work

- A search engine operates in the following order:
  - Web crawling -A **Web crawler** is an Internet bot that systematically browses the WWW, typically for the purpose of Web indexing. A Web crawler may also be called a **Web spider**, an **ant**, an **automatic indexer**.
  - Indexing
  - Searching


Search Form
Query Parser
tokenize terms, then look for operators and filters
Index File
Query Engine
find documents that match criteria
Relevance Ranker
sort documents by match term locations, frequency, etc.
Search Results
Formatter
lay out result page and format result items

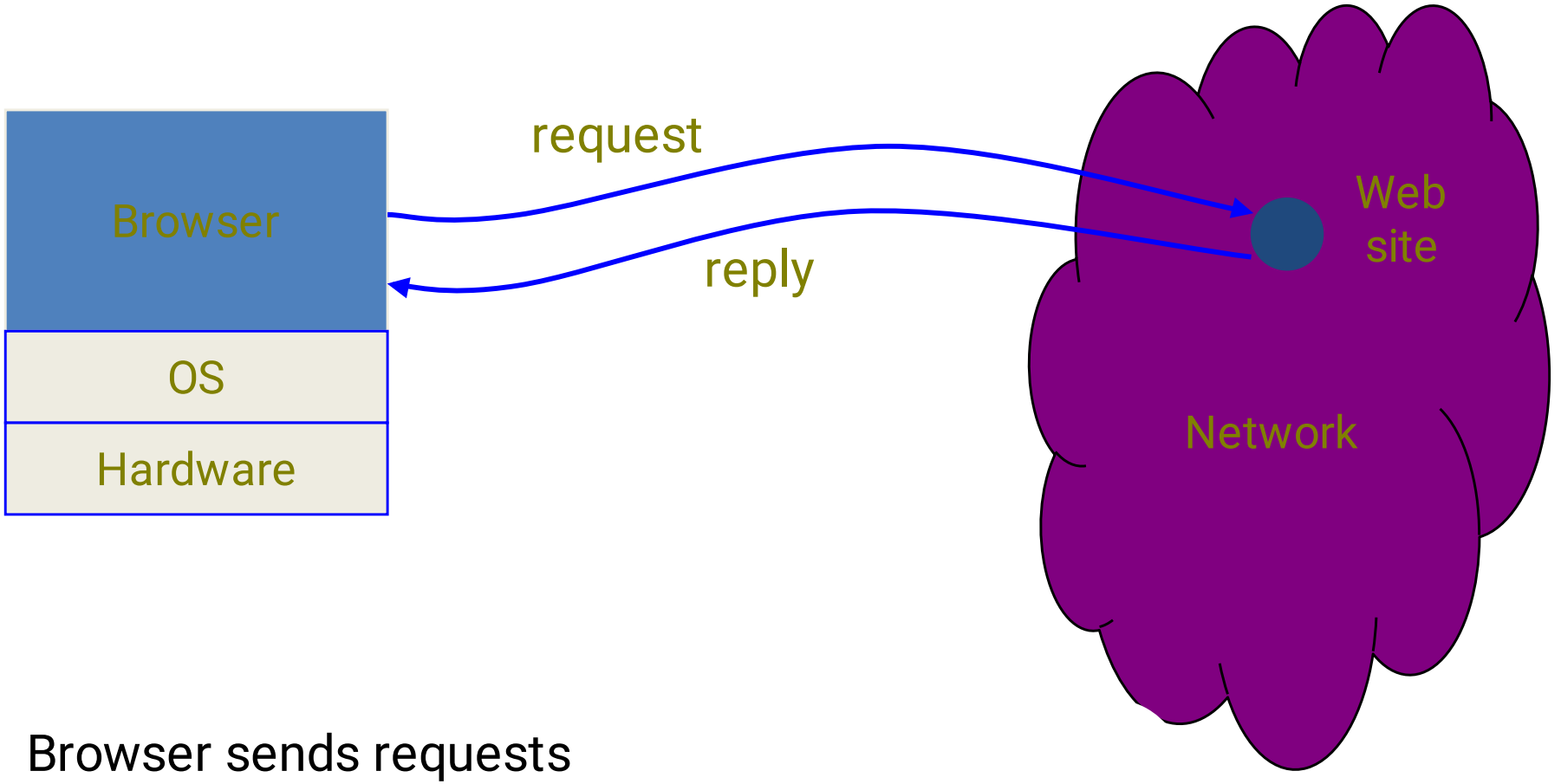
Architecture of a Search Engine



Web Security

- Many sensitive tasks are done through web
  - Online banking, online shopping
  - Database access
  - System administration
- Web applications and web users are targets of many attacks
  - Cross site scripting
  - SQL injection
  - Cross site request forgery
  - Information leakage
  - Session hijacking

Web Browser and Network



- Browser sends requests
- Web site sends response pages, which may include code
- Interaction susceptible to network attacks

Web Security Issues

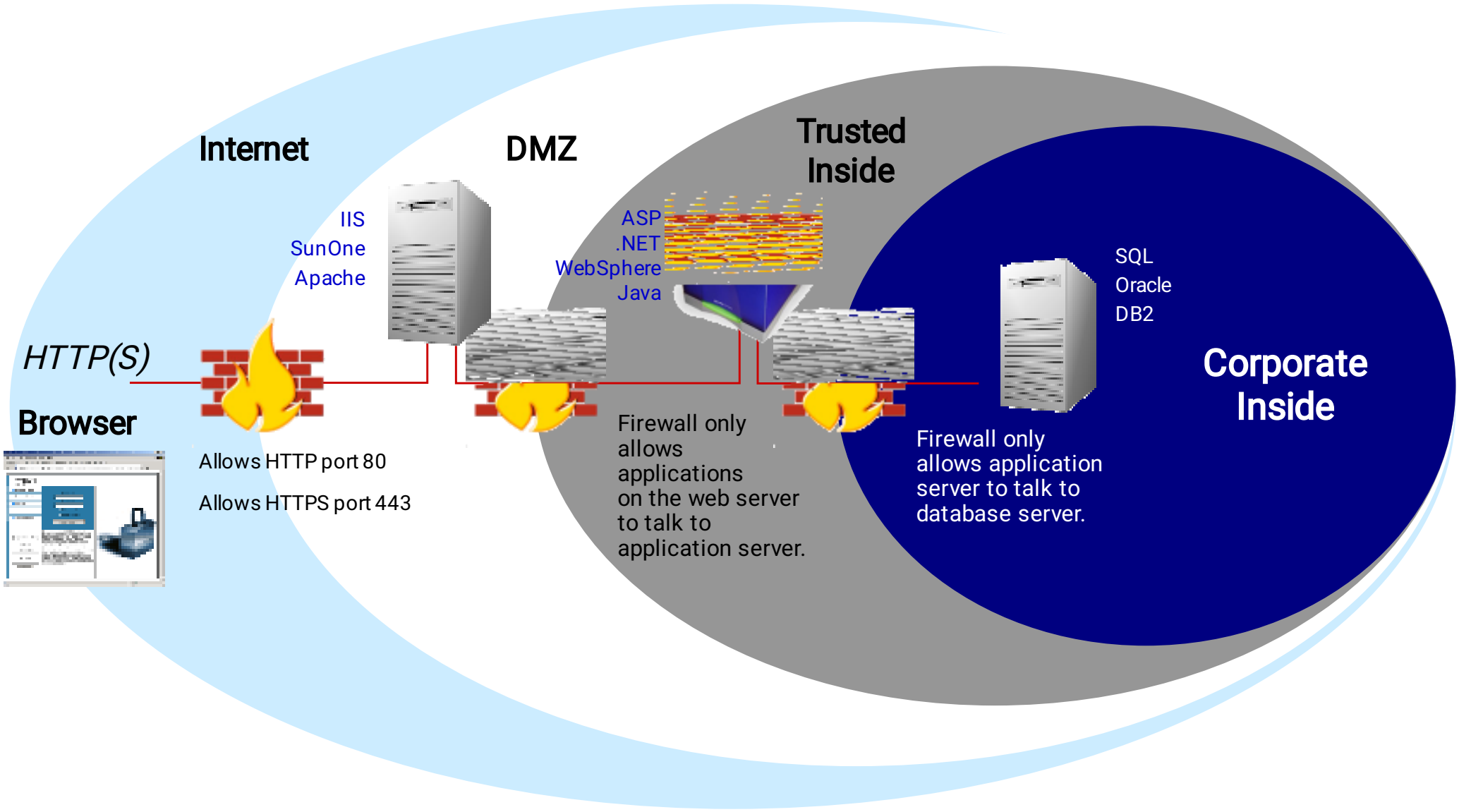
- Secure communications between client & server
  - HTTPS (HTTP over SSL)
- User authentication & session management
  - Cookies & other methods
- Active contents from different websites
  - Protecting resources maintained by browsers
- Web application security
- Web site authentication (e.g., anti-phishing)
- Privacy concerns

Effect of the Attack

- Attacker can execute arbitrary scripts in browser
- Can manipulate any DOM component on victim.com
  - Control links on page
  - Control form fields (e.g. password field) on this page and linked pages.
- Can infect other users: MySpace.com worm.

- More Web Security Issues
  - SQL injection
  - Side channel information leakage
  - Driveby downloads
  - Browser extension security
  - Cookie privacy issues

Web Applications Breach the Perimeter



Web Application Vulnerabilities

- Technical Vulnerabilities
  - Result of insecure programming techniques
  - Mitigation requires code changes
  - Detectable by scanners
- Logical Vulnerabilities
  - Result of insecure program logic
  - Most often to due to poor decisions regarding trust
  - Mitigation often requires design/architecture changes
  - Detection often requires humans to understand the context

How to Secure Web Applications

- Incorporate security into the lifecycle
  - Apply information security principles to all software development efforts
- Educate
  - Issue awareness, Training, etc…

How to Secure Web Applications

- Incorporating security into lifecycle
  - Integrate security into application requirements
  - Including information security professionals in software architecture/ design review
  - Security APIs & libraries (e.g. ESAPI, Validator, etc.) when possible
  - Threat modeling
  - Web application vulnerability assessment tools