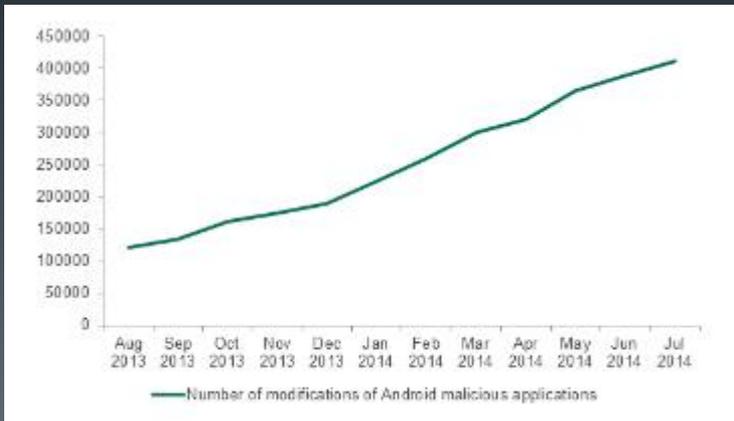
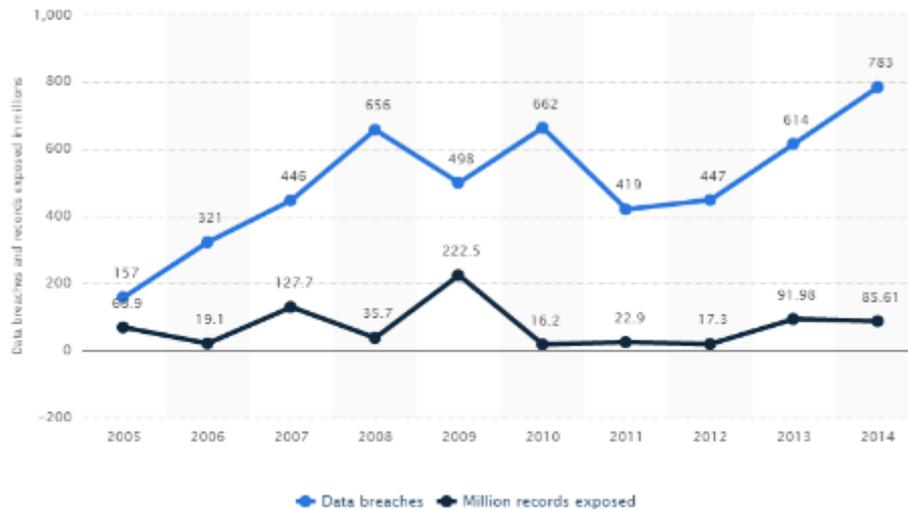


Cyber Crime Trends



Annual number of data breaches and exposed records in the United States from 2005 to 2014 (in millions)

The statistic presents the development of cyber attacks over time. It presents the recorded number of data breaches and records exposed in the United States between 2005 and 2014. In 2014, the number of data breaches in the United States amounted to 783 with more than 85.61 million records exposed.



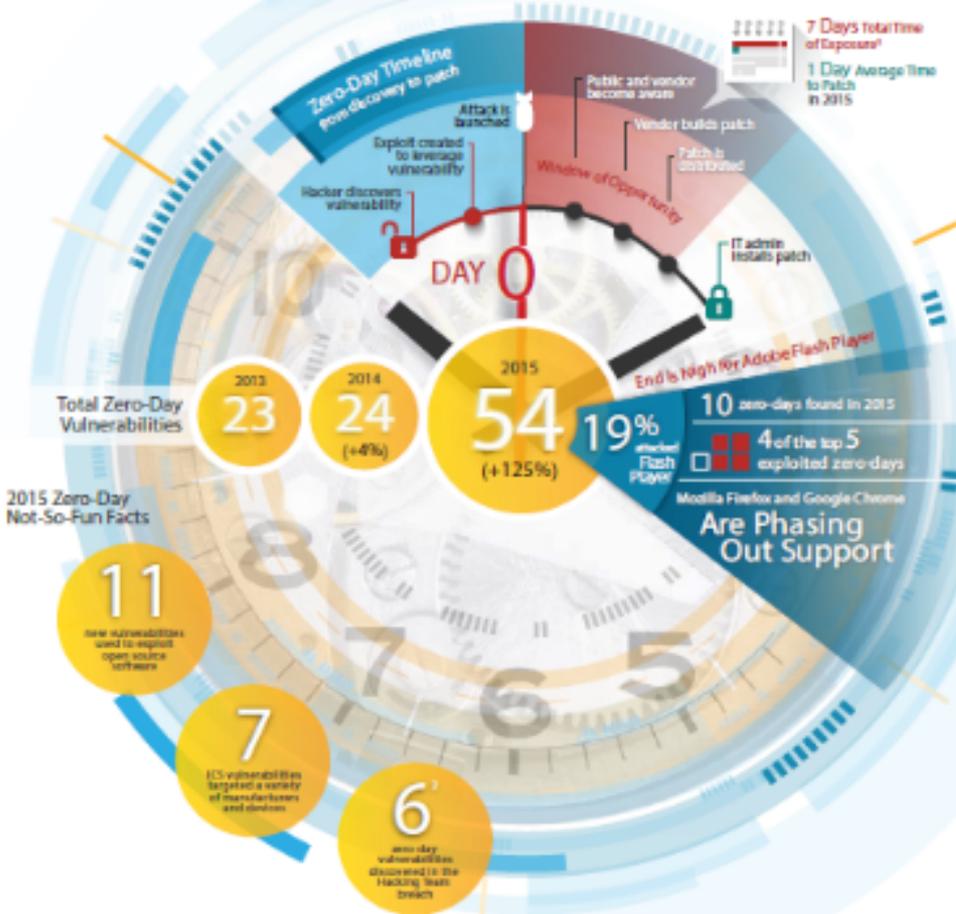
Cyber Crime Trends



A New Zero-Day Vulnerability Discovered Every Week in 2015¹

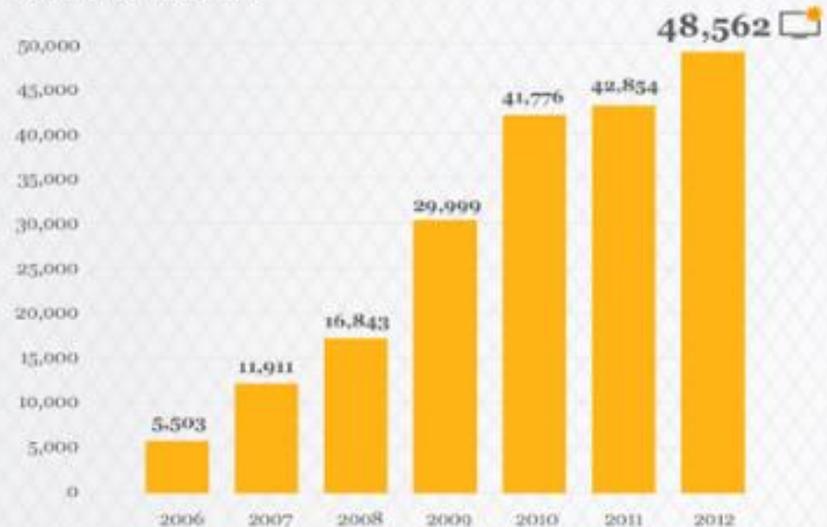
Advanced attack groups continue to profit from previously undiscovered flaws in browsers and website plugins.

In 2015, 54 zero-day vulnerabilities were discovered.



NUMBER OF INCIDENTS REPORTED TO US-CERT FISCAL YEARS 2006 -2012 FROM FEDERAL AGENCIES¹

NUMBER OF INCIDENTS



¹For average, based on 10 vulnerabilities.
²Source: Symantec Security Intelligence Center (SIC).
³Source: Symantec Security Intelligence Center (SIC).
⁴Source: Symantec Security Intelligence Center (SIC).



Cybersecurity Jobs



- According to the Bureau of [Labor Statistics](#), the rate of growth for jobs in information security is projected at 37% from 2012–2022—that’s much faster than the average for all other occupations.
- A [recent study](#) found that 84 percent of cyber security positions require a bachelor's degree or higher.
- [More than 209,000 cyber security jobs in the U.S. went unfilled in 2015](#). Job postings are up 74 percent over the past five years. The demand information security professionals is expected to grow by 53 percent through 2018.
- Forbes: cybersecurity market which is expected to grow from \$75 billion in 2015 to [\\$170 billion by 2020](#).
- [Cisco puts the global figure at one million cybersecurity job openings](#).

Cybersecurity

- Collection of technologies to protect computer systems (networks, computers, programs, data) from unauthorized access, theft, damage and to prevent failure or misdirection of provided services.



Cybersecurity Goals: CIA triad

■ Confidentiality

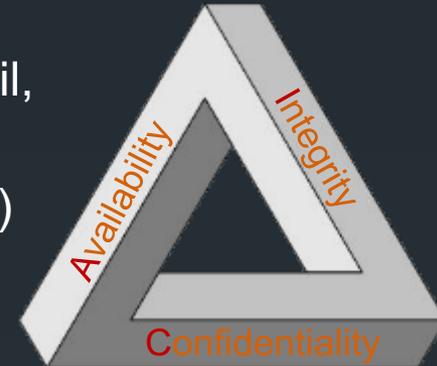
- Communication: How to protect confidentiality of email, web surfing history?
- System: How to control the access (aka authorization)

■ Integrity means no tampering

- File integrity – has a file been tampered with?
- Communication integrity - error detection and correction.
- System integrity – has a computer been compromised?
- Data integrity - the absence of tampering by unauthorized parties.

■ Availability

- Network availability: How to distinguish flash crowd from DDOS attack?
- System availability: Fault-tolerant, graceful degradation, QoS control.



Cybersecurity Goals: AAA triad

■ Authenticity

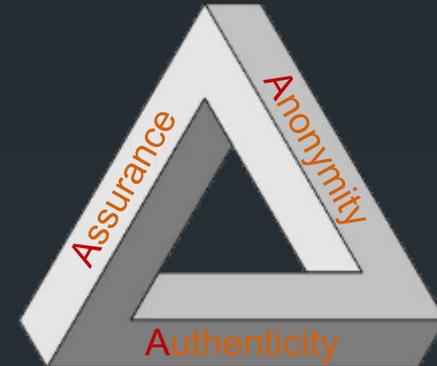
- How to prove who you are to a system (what you have, what you know)?
- How to prove the origin/source of an email?
- Data authenticity - ensuring that data originates from the correct source.

■ Anonymity

- How to remain anonymous (untraceable)?
- When should a system provide anonymity?

■ Assurance

- How digital trust in a system is achieved? (CIA triad – integrity)
- How trust is provided and managed?
- How is trust delegated?



Attacks: Buffer Overflow (Intro)



- Malicious user inputs data stored into an array that exceeds the array bounds to write into adjacent data memory and/or executable code.
- Can result in segmentation faults/access violations, program freezes, invalid output.
- Virulent input beyond the array bounds can over-write program code with malware.
- C and C++ programs are susceptible to [buffer overflow attacks](#). The languages provide no automatic array bounds checking, instead it is the [responsibility of the programmer](#).

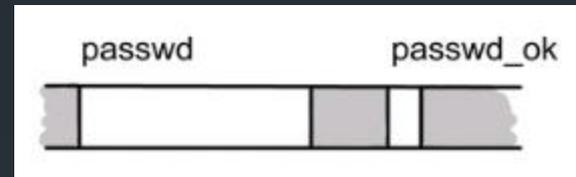
Attacks: Buffer Overflow



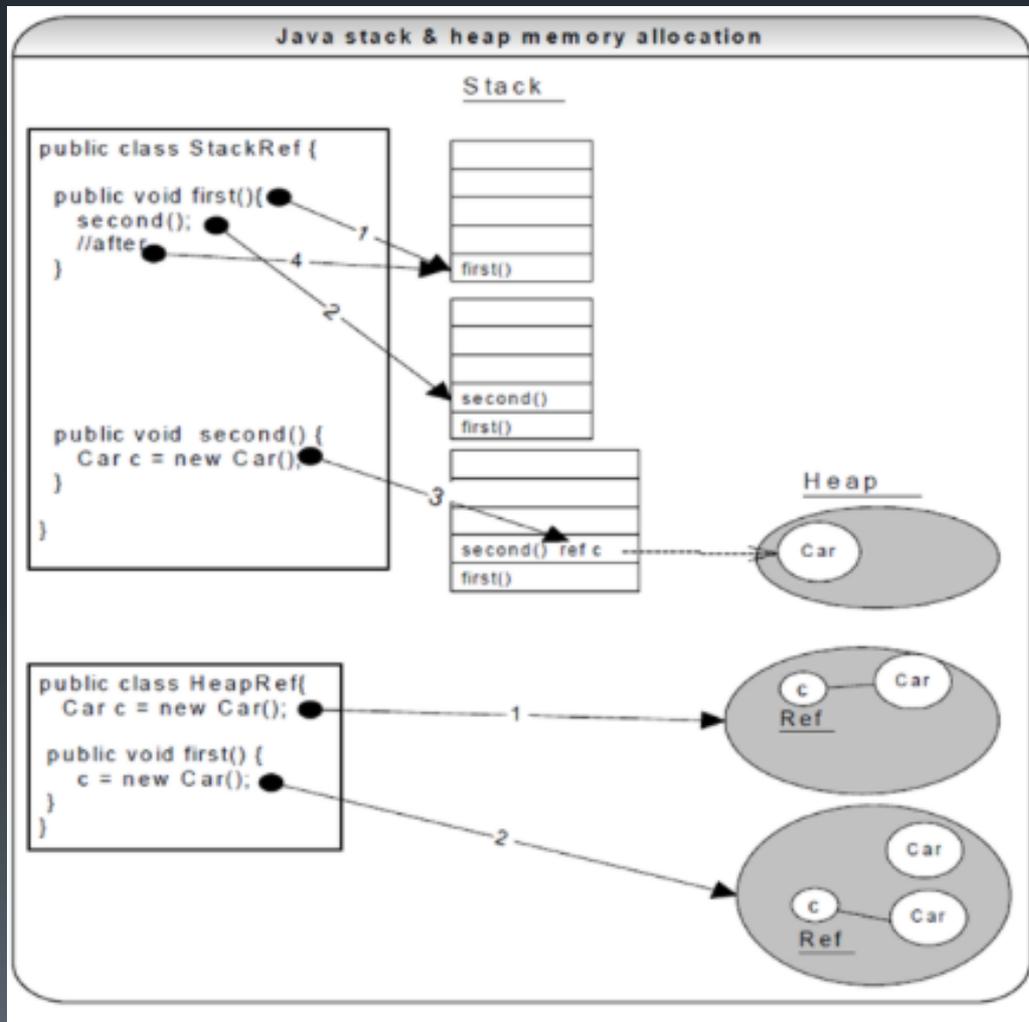
- Buffer overflow attacks do NOT always employ code insertion.
- Consider simple password checking code:
 - If the passwd_ok variable follows the passwd array then an overflow of one value would rewrite the passwd_ok initialization to true
 - The passwd_ok check always passes granting the attacker access regardless of the supplied password value.

```
int main(int argc, char *argv[]) {
char passwd_ok = 0;
char passwd[8];
strcpy(passwd, argv[1]);
if (strcmp(passwd, "niklas")==0)
    passwd_ok = 1;

if (passwd_ok) {
    ...
}
```



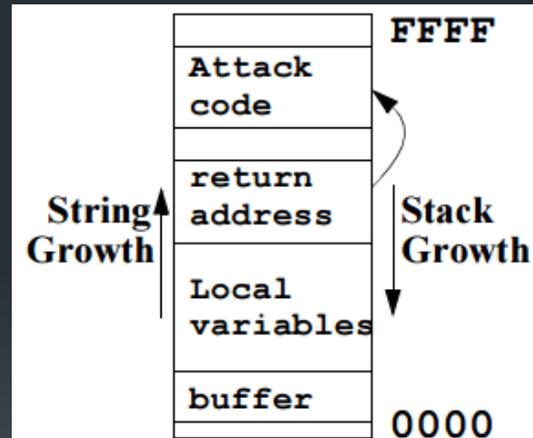
Review: Memory Stack vs Heap



Attacks: Buffer Overflow (Stack)



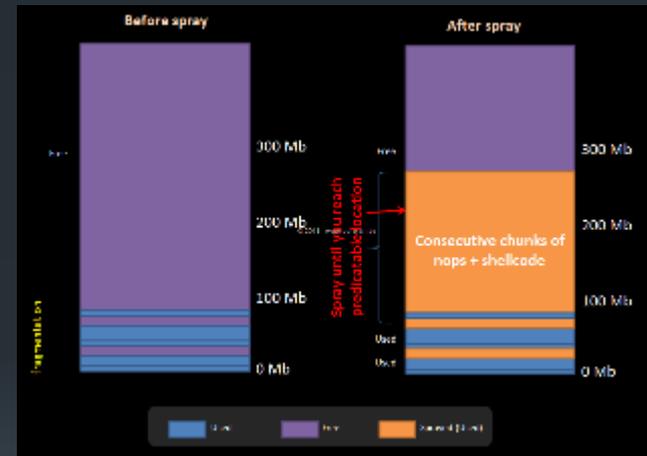
- Attempts to over-write the runtime stack frame return address to point to hack supplied code: termed *stack smashing*.
- All arrays in Java are automatically checked for array bound overflow throwing an exception.



Attacks: Buffer Overflow (Heap)



- Data stored/buffered in dynamically allocated memory in the heap is overwritten/overflowed to replace actual data with malicious data, non-control-data attack.
- Typical heap attack attempts to overwrite system heap data structures to trick heap management code into writing attacker data into a desired location.
- The attacker supplied data may replace a function pointer address to transfer control to [attacker code/payload](#).
- Over-written heap data could replace a file name pointer stored in the heap to point to a malicious string containing the name to a system file, (e.g., ["/root/.ssh"](#)), to grant attacker remote system access.
- The same Java array bounds checking that prevents stack smashing also prevents heap overflow attacks, ([heap spraying](#)).



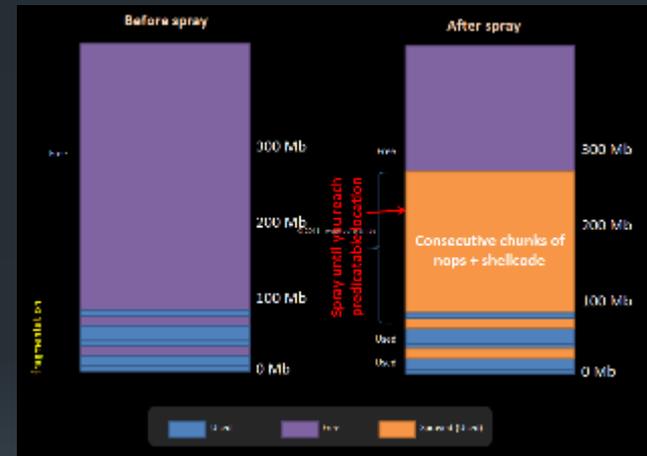
| Address | Contents |
|----------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 0c080018 | 0x1000 bytes: Nops shellcode 0x1000 bytes: Nops shellcode 0x1000 bytes: Nops shellcode 0x1000 bytes: Nops shellcode ... 0x1000 bytes: Nops shellcode |
| 0c090018 | 0x1000 bytes: Nops shellcode 0x1000 bytes: Nops shellcode 0x1000 bytes: Nops shellcode 0x1000 bytes: Nops shellcode ... 0x1000 bytes: Nops shellcode |
| 0c0a0018 | 0x1000 bytes: Nops shellcode 0x1000 bytes: Nops shellcode 0x1000 bytes: Nops shellcode 0x1000 bytes: Nops shellcode ... 0x1000 bytes: Nops shellcode |
| 0c0b0018 | 0x1000 bytes: Nops shellcode 0x1000 bytes: Nops shellcode 0x1000 bytes: Nops shellcode 0x1000 bytes: Nops shellcode ... 0x1000 bytes: Nops shellcode |
| 0c0c0018 | 0x1000 bytes: Nops shellcode 0x1000 bytes: Nops shellcode 0x1000 bytes: Nops shellcode 0x1000 bytes: Nops shellcode ... 0x1000 bytes: Nops shellcode |
| 0c0d0018 | |

0x0c0c0c0c

Defense: Buffer Overflow (Heap)



- Address Space Layout Randomization ([ASLR](#)): randomly locates base address of the executable, positions of the stack, heap & libraries.
- Guard Pages: OS places [guard pages](#) between heap buffers, any access aborts process.
- Data Execution Prevention (DEP): OS prevents the execution of code from the heap.
- Testing: programmers check that input data does not exceed the size of arrays/buffers.
- Eliminate the use of library code that does NOT check their argument lengths, (e.g. C: strcpy, sprintf, etc.), replacing with safe library alternatives, (strncpy, snprintf).
- Use a VM language (e.g., Java, C#) that provides automatic bounds checking.



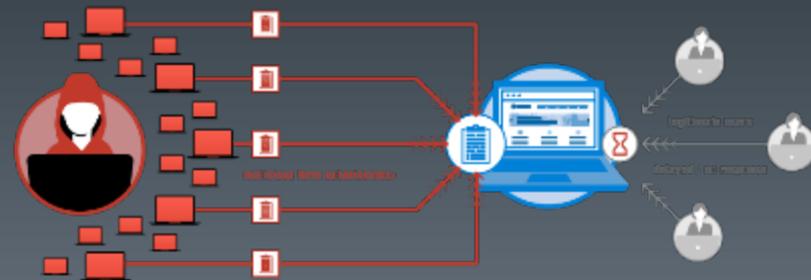
| Address | Contents |
|----------|-----------------------------------------------------------------------------------------------------------------------------------------------------------|
| 0c080018 | 0x1000 bytes Nops shellcode 0x1000 bytes Nops shellcode 0x1000 bytes Nops shellcode 0x1000 bytes Nops shellcode ... 0x1000 bytes Nops shellcode |
| 0c090018 | 0x1000 bytes Nops shellcode 0x1000 bytes Nops shellcode 0x1000 bytes Nops shellcode 0x1000 bytes Nops shellcode ... 0x1000 bytes Nops shellcode |
| 0c0a0018 | 0x1000 bytes Nops shellcode 0x1000 bytes Nops shellcode 0x1000 bytes Nops shellcode 0x1000 bytes Nops shellcode ... 0x1000 bytes Nops shellcode |
| 0c0b0018 | 0x1000 bytes Nops shellcode 0x1000 bytes Nops shellcode 0x1000 bytes Nops shellcode 0x1000 bytes Nops shellcode ... 0x1000 bytes Nops shellcode |
| 0c0c0018 | 0x1000 bytes Nops shellcode 0x1000 bytes Nops shellcode 0x1000 bytes Nops shellcode 0x1000 bytes Nops shellcode ... 0x1000 bytes Nops shellcode |
| 0c0d0018 | |

0x0c0c0c0c

Denial of Service (DOS) Attacks



- **DOS**: Attacker attempts to make a machine or network resource unavailable to users.
- Usual attack floods server or resource with fake requests to overload systems to prevent legitimate requests being fulfilled.
 - Distributed DOS (**DDOS**): requests are sent from many machines, (Botnet: zombie computers that receive/execute remote commands without owner's knowledge).
 - DRDOS (Distributed Reflected DOS): Forged requests are sent to many servers with the source address of the request spoofed to be the target under attack.
- Degradation of Service: a form of DOS not intended to disrupt service, but to slow service.
 - Can appear as legitimate increased traffic flow, (flash crowd).
- DOS attacks attempts to consume a resource, (bandwidth, memory, CPU, disk space, etc.) until it is exhausted.



DOS Defenses



- All DOS attacks can **NOT** be prevented.
- Ingress/Egress Filtering: verification method for checking that network packets are actually from their source before the packets are processed (granted entry/exit).
 - Routers block packets from sending networks if the originating address is not within the sending network.
 - Prevents IP address spoofing.
- **Blacklist/Whitelist**: access from blacklisted countries is prevented & access from whitelisted countries is always allowed.
- **Intrusion Prevention System (IPS)** software/devices that detect & attempt to stop the attack.
 - IPS may block network traffic from triggering IP addresses or ignore suspicious packets, (may result in false positives.)
- **Load Balancer**: system is configured to detect when thresholds of resource consumption is reached. Load balancing is employed to spread the workload over multiple servers.
- **Software Hardening**: code systems that can detect and react to DOS attacks. (*The focus of the Cybersecurity labs.*)

DOS Resource Cache Attack



Attacker: assume the attacker can control 80% of the request workload sent to the resource (database) cache.

Attacker may attempt to degrade performance for legitimate users by sending a workload that increases likelihood of cache misses to occur. Such a workload would request keys only once, trying to use up capacity in the cache.

Rate limiting. To avoid compromising the database's availability, each user's rate of database access is limited

Receives value ←

Sends key →

Cache

Caches a limited number of requested (key, value) pairs in memory using a data structure.
Limited capacity.



Database

Legitimate Users: their workload has some locality: the same key is accessed multiple times, for instance during a session.

Cache hit: Key that is requested is in the cache, it is looked up and the corresponding value is returned.

Cache miss: Key that is requested is *not* in the cache. It is retrieved from the database instead and added to the cache if possible. The database is slower than the cache, thus a database penalty is incurred in the overall latency.