

Software Engineering in Industrial Practice (SEIP)

Dr. Ralf S. Engelschall

ARCHITECTURE FUNDAMENTALS

Back of the Envelope Calculation

TECHNISCHE UNIVERSITÄT MÜNCHEN

AF 16.5

Intellectual Content: Version 1.1.0 (2021-11-23) Authored 2011-2021 by Dr. Ralf S. Engelschall
Graphical Illustration: Version 1.3.0 (2023-10-10) Copyright © 2012-2022 Dr. Ralf S. Engelschall <http://engelschall.com> All Rights Reserved
Unauthorized Reproduction Prohibited. Licensed to Technische Universität München (TUM) for reproduction in Computer Science lecture contexts only.

Specification (Example)

Customer: Twitter Inc.
Business: MicroBlogging

Use-Cases 1/3 (profile):
- user can register an account
- user can "follow" other users
- user can create lists of users he follows

Use Cases 2/3 (send):
- user can send tweets
- tweets are based on words, each either a text "example", tag "#example", user reference "@example" or URL http://example.com
- tweets are either public broadcast or personal/direct messages
- user can re-tweet a message of others

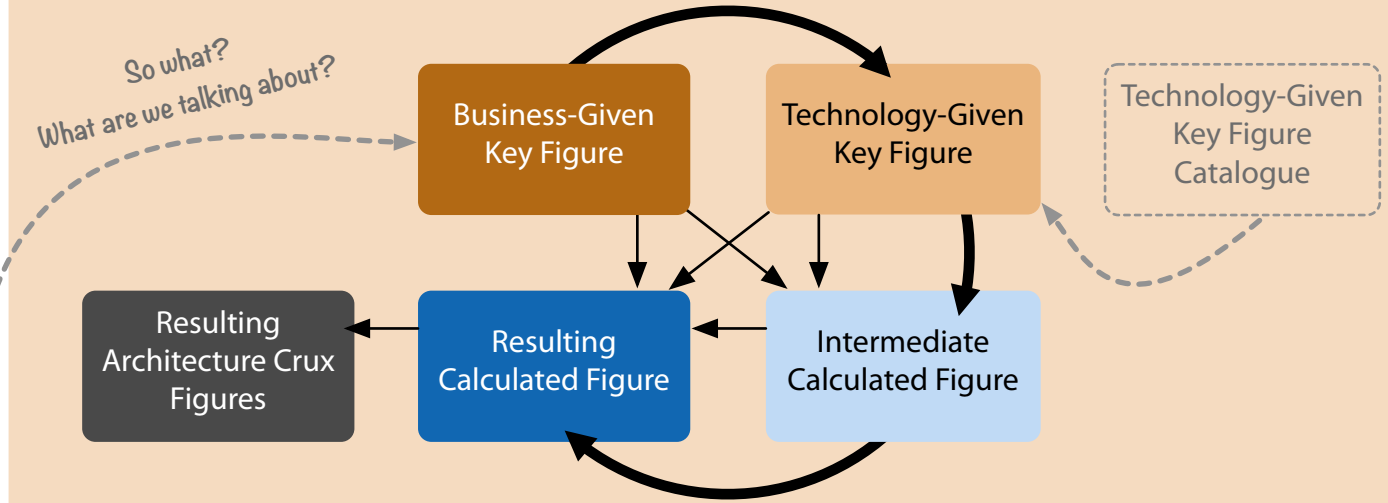
Use Cases 3/3 (query):
- user can view timeline (chronological tweets of others he follows)
- user can search for tweets (by keyword "foo", tag "#foo", or user "@foo")
- user can view tag cloud

Frontends/Clients:
- mobile app (iOS, Android)
- desktop app (Windows, Mac OS X)
- web app
- embedded web widget (query use cases only)

Current Demand (as of 2012):
- 140M user profiles
- 400M tweets/day
- 6393 tweets/second peak
- 140 characters/tweet
- 30K queries/second
- 300 GB/hour data in total
- 4,4 tweets/day/user on average
- 103,4 follower/user
- < 5s tweet-write-to-read-delay

Future Demand:
- quadratic user and traffic growth

Conversion & Normalization



Calculation (Example)

Twitter Information

6.393	tweets/second peak
400.000.000	tweets/day (write)
2.592.000.000	queries/day (read)
6,5	factor read/writes

140	chars/tweet
4.630	tweets/second (write)
30.000	queries/second (read)
10	tweets/query
4,4	tweets/day/user
2,4	tweets/day (M. Fowler)
0,8	tweets/day (R. Engelschall)
621.880.000	tweets/day (average) total
64,3%	users are active at all
277.778	users/minute active

Traffic Bandwidth

350%	overhead HTTP+TCP+IP+Ethernet
2,2	MB/s (write)
140,2	MB/s (read)
142,4	MB/s

10000	Mbps	1250	MB/s
1000	Mbps	125	MB/s
100	Mbps	12,5	MB/s
10	Mbps	1,25	MB/s

Storage Requirements (static)

140.000.000	user profiles
52.000	chars/users for profile
6,62	TB profile (total)
103,4	follower/user
14.474.600.000	user follow links
32	bytes/link
0,42	TB links (total)

Storage Hardware Requirements

0,3	TB/disk (15K rpm)
8,0	disks/server
2,4	TB/server
8,2	server/month (new)

Storage Requirements (dynamic)

300	GB/hour data in total
214	TB/month data in total
200%	overhead storage
1265,9	KB/s tweets
104,3	GB/day tweets
3,1	TB/month tweets
200	chars/log entry
6763,6	KB/s log
557,3	GB/day log
16,6	TB/month log
9,2%	ratio business data
90,8%	ratio infrastructure data

Computing Hardware Requirements

2000	requests/sec (read) AS performance
100	requests/sec (write) As performance
15,0	servers for writes
46,3	servers for reads

ARCHITECTURE FUNDAMENTALS

Weighted Decision Matrix

TECHNISCHE UNIVERSITÄT MÜNCHEN

Weighted Decision Matrix

		A_1	A_2	…	A_n
C_1	w_1	$E_{1,1}$	$E_{1,2}$	…	$E_{1,n}$
C_2	w_2	$E_{2,1}$	$E_{2,2}$	…	$E_{2,n}$
…	…	…	…	…	…
C_m	w_m	$E_{m,1}$	$E_{m,2}$	…	$E_{m,n}$
		R_1	R_2	…	R_n

$C_{i=1..m}$: Criteria i

$A_{j=1..n}$: Alternative j

$w_{i=1..m}$:in [1/4, 1/2, 1, 2, 4]: Weighting i

$E_{i,j}$ in { -2, -1, 0, +1, +2 }: Evaluation i,j

$R_{j=1..n} = SUM_{i=1..m}(w_i * E_{i,j})$: Rating j

$R_{best} = MAX_{j=1..n}(R_j)$: Best Rating

Light-Weight Alternative:
qualitatively cherry-picking major positive/negative backing criterias

	A_1			A_2			…	A_n		
+	C_1	C_4	C_8	C_1	C_2	C_7	…	C_2	C_9	C_8
-	C_2	C_3		C_4			…	C_7	C_5	
	Decision for A_{best}									

Best Practice Rules

Rule 1: the alternatives have to be really **reasonably comparable**.

Rule 2: the **best** rating should be a least **10% above** the **second** best rating.

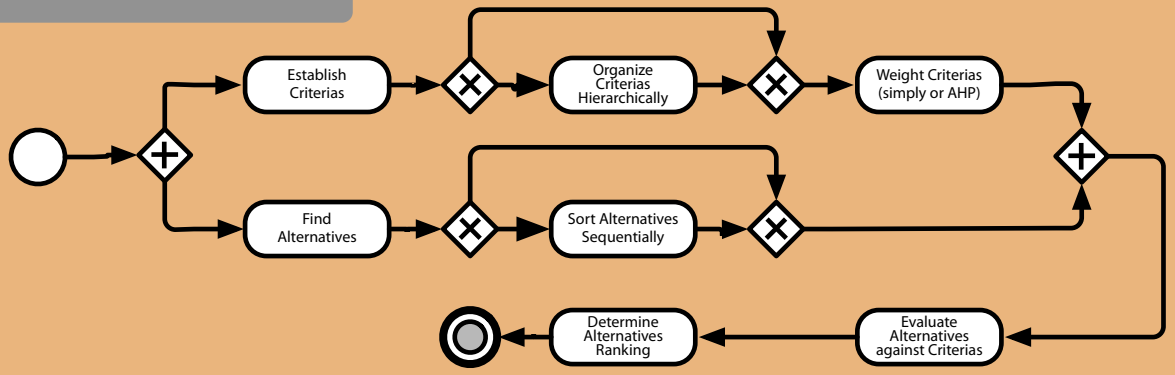
Rule 3: the **best** rating should cover at least **80%** of the **requirements**.

Rule 4: the Weighted Decision Matrices should cover at least all **grand decisions**.

Notice

It's about **subjective** decision **transparency**, not about **objective** decision **making**!

Decision Making Process



Standard Criteria Catalogs

Software Selection:
Suitable Functionality
Available Usage Examples
Reasonable Documentation
Reasonable Support
Permissive License
Long-Term Release Track Record
Current Market Momentum

Software Selection (Open Source):
+ Clean Source Code
+ Clean Build Process
+ Open Source License

Software Selection (Library):
+ Non-Invasive Programming Model
+ Orthogonal Application Programming Interface
+ Minimum/No Dependencies
+ Non-Copyleft Open Source License

Software Selection (Framework):
+ Orthogonal Application Programming Interface
+ Adequate Dependencies
+ Non-Overlapping Scope
+ Non-Copyleft Open Source License

Software Selection (Tool):
+ Clean Deployment Procedure
+ Pleasant Command-Line Interface

Software Selection (Application):
+ Clean Deployment Procedure
+ Pleasant Graphical User Interface

Software Architecture Evaluation:
Meets Functional Requirements
Meets Non-Functional Requirements
Adequate Technology Overhead
Single Dependency Direction
Distance to State of the Art ("modern")
Distance to Most Simple Approach ("adequate")
Distance to Mainstream Approach ("mainstream")
Documented Architecture Decisions ("rationales")
Documented Architecture Views
Documented Architecture Perspectives (NFR)

AF 16.6

Intellectual Content: Version 1.11 (2022-05-20), Authored 2018-2022 by Dr. Ralf S. Engelschall
Graphical Illustration: Version 1.10 (2021-09-26), Copyright © 2018-2021 Dr. Ralf S. Engelschall <http://engelschall.com>, All Rights Reserved
Unauthorized Reproduction Prohibited. Licensed to Technische Universität München (TUM) for reproduction in Computer Science lecture contexts only.

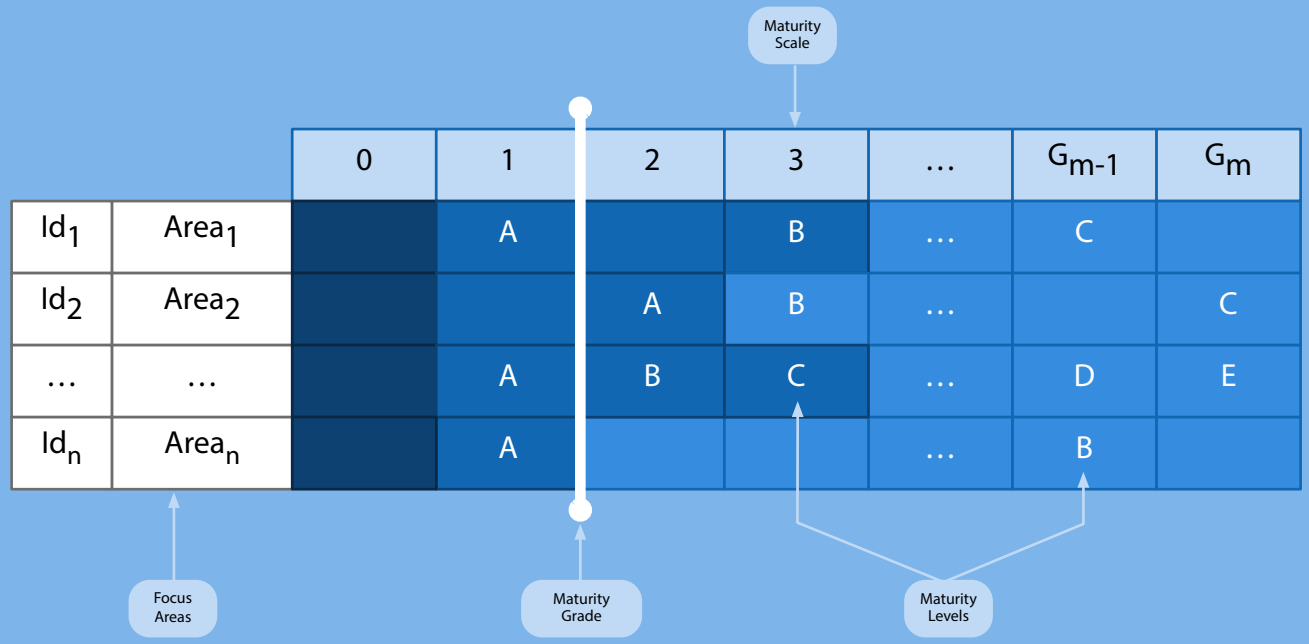
ARCHITECTURE FUNDAMENTALS

Focus Area Maturity Model

TECHNISCHE UNIVERSITÄT MÜNCHEN

AF 16.7

Matrix **Structure**



Focus Area Definition

Id: <unique id of focus area>
Name: <unique name of focus area>

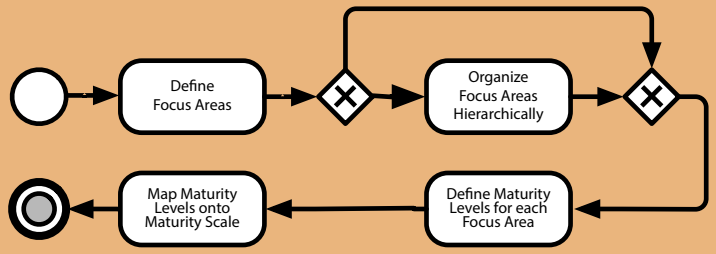
Maturity Level Definition

Id: <unique id of focus area>
Level: <unique letter of maturity level>
Name: <unique name of capability>
Goal: <purpose the capability serves>
Action: <steps how to meet the capability>
Prerequisites: <optional references to Id/Level>
References: <optional external references>

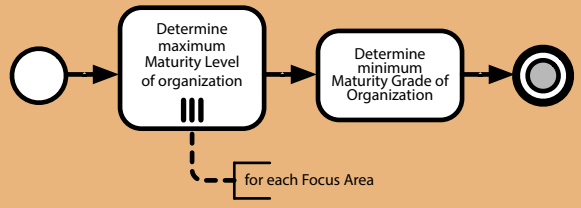
Maturity Level **Prerequisites**

Notice: Maturity Levels are inherently ordered within their Focus Area, but optionally also form a dependency graph by cross-referencing Maturity Levels of other Focus Areas.

Matrix **Design** Process



Maturity **Decision** Process



Maturity Grade **Zero**

Notice: the Maturity Scale always starts with 0, because an organization might not be able to fulfil a Focus Area at all, i.e., it might to not even be on Maturity Level A.

Maturity Grade **Determination**

Determine minimum Maturity Level fulfilled by an organization and project from Maturity Level onto Maturity Scale.

Intellectual Content: Version 1.0.1 (2023-12-01), Authored 2022-2023 by Dr. Ralf S. Engelschall
Graphical Illustration: Version 1.0.0 (2023-05-21), Copyright © 2022 Dr. Ralf S. Engelschall <http://engelschall.com> All Rights Reserved
Unauthorized Reproduction Prohibited. Licensed to Technische Universität München (TUM) for reproduction in Computer Science lecture contexts only.

ARCHITECTURE FUNDAMENTALS

Big Picture (8-D)

TECHNISCHE UNIVERSITÄT MÜNCHEN

AF 17.1

Vote is a portable mobile-first designed application for easily performing anonymous online votings within a small group of people to figure out their opinions or moods.

Votings are created in advance, executed at a certain time, conducted by the users, and then finally reported.

1 Elevator Pitch (RE D)

Name, Purpose, Motivation, Actors, Devices.

Rationale: Roughly describe the purpose and primary motivation.

Format: Prose Abstract

2 Crux Flash (RE D)

Functionality, Cruxes.

Rationale: Roughly describe the functionality and the cruxes.

Format: Prose Abstract

Votings can be quickly accessed by QR-code or URL and are based on one or more questions and corresponding multiple-choice-based answers.

Votings are interactively conducted, and answers are received and reported either asynchronously in batches (offline voting) or even synchronously in real-time (online voting).

Actor Role	Use-Case
User	Register Account
	Recover Account
	Configure Account
	Login Account
	Logout Account
Author	Create Voting
	Grant Voting Access
Supervisor	Execute Voting
	Enable Question
	Display Result
Voter	Vote Question
	Display Result

3 Customer Journey (UX D)

Actor Roles, Use-Cases.

Rationale: Sketch the customer journey through major use-cases.

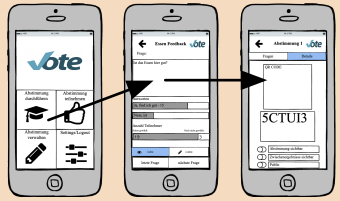
Format: 2xN Table or UML UC Diag.

4 Dialog Storyboard (UX D)

Dialogs, Interaction, Control Flow.

Rationale: Illustrate the major user interface dialogs (or dialog types).

Format: Wireframe Graph Diagram



Quality	Expectation
Cross-Platform Client	yes
Non-Cleartext Password Storage	yes
Minimum Concurrent Voters (people)	50
Maximum Display Result Latency (sec)	1

5 Quality Requirements (RE T)

Qualities, Expectations.

Rationale: List requirements on the major non-functional qualities.

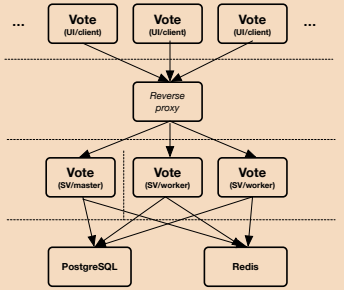
Format: 2xN Table

6 System Architecture (SY T)

Actors, Systems, Zones, Programs.

Rationale: Illustrate the major system architecture components.

Format: Boxes'n'Lines Diagram



Aspect	Amount	Size	Total Size	Unit
Account Data	10.000	256	2.560.000	B
Voting Data	10.000	1.024	10.240.000	B
Server RAM Usage	200	20	4.000	MB

8 Sizing Sketch (SW T)

Aspects, Amounts, Sizes, Total Sizes, Units.

Rationale: Sketch the sizing of major entities and system parts.

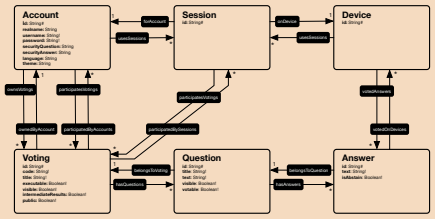
Format: 5xN Table

7 Data Model (SW T)

Entities, Relationships.

Rationale: Model major data entities and their relationships.

Format: UML Class Diagram



(Example) (Method) (Example)

1 Creation Step · RE Requirements Engineering · UX User Experience · SY Systems Architecture · SW Software Architecture · D Domain Scope · T Technology Scope

Intellectual Content: Version 1.03 (2020-07-20), Authored 2019-2020 by Dr. Ralf S. Engelschall
Graphical Illustration: Version 1.03 (2020-07-20), Copyright © 2019-2020 Dr. Ralf S. Engelschall <http://engelschall.com> All Rights Reserved.
Unauthorized Reproduction Prohibited. Licensed to Technische Universität München (TUM) for reproduction in Computer Science lecture contexts only.

ARCHITECTURE FUNDAMENTALS

Viewpoints & Perspectives

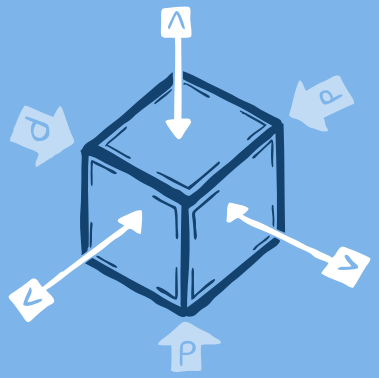
TECHNISCHE UNIVERSITÄT MÜNCHEN

AF 17.2

Development

5

Aspects of the software development process for versioning, building, testing, maintaining, and enhancing the system.



Development Process Viewpoints

Functionality

2

System's functional elements, their responsibilities, interfaces, and primary interactions (control flow).

Information

3

Static data structures and information flows to store, manipulate, manage, and distribute information.

Concurrency

4

Concurrency structure of the system and mapping of functional elements to concurrency units (processes, threads, transaction scopes).

Application Architecture Viewpoints

Context

1

Relationships, dependencies, and interactions between the system and its run-time environment (people, systems, external entities).

Application Overview Viewpoints

Deployment

6

Required technical environment and mapping of software elements to runtime environment that will execute them.

Operation

7

Aspects to operate, administer, update, upgrade and support the system when running in its production environment.

Operation Process Viewpoints

Evolution & Change

Ability of the system to be flexible in the face of the inevitable change that all systems experience over time.

Regulation & Compliance

Ability of the system to conform to local and international laws, quasi-legal regulations, company policies, and other rules and standards.

Internationalization & Localization

Ability of the system to be independent from and adaptable to any particular language, country, or cultural group.

Usability & Accessibility

Ability of the system to allow people to effectively interact with the system and also to be even used by people with disabilities.

Performance & Scalability

Ability of the system to predictably execute within its mandated performance profile and to handle increased processing volumes.

Availability & Resilience

Ability of the system to be fully or partly operational when required and to effectively handle failures.

Constraints & Resources

Ability of the system to be designed, built, deployed, and operated within known constraints around people, budget, time, and materials.

Security & Recoverability

Ability of the system to reliably control and audit who can perform what actions on what resources and to detect and recover from failures.

Cross-Cutting Perspectives

Intellectual Content: Version 1.04 (2019-09-15) Authored 2011-2019 by Dr. Ralf S. Engelschall based on Nick Rozanski, Eoin Woods: "Software Systems Architecture", 2nd ed. 2011, ISBN 978-0-321-71833-4
Graphical Illustration: Version 1.08 (2018-09-15) Copyright © 2011-2018 Dr. Ralf S. Engelschall <http://engelschall.com> All Rights Reserved
Unauthorized Reproduction Prohibited. Licensed to Technische Universität München (TUM) for reproduction in Computer Science lecture contexts only.

ARCHITECTURE FUNDAMENTALS

Security Basics

TECHNISCHE UNIVERSITÄT MÜNCHEN

AF 18.1

Information Protection Requirements

Confidentiality / **Vertraulichkeit** (germ.)

Information and systems are available to authorized users only.

Informationen und Systeme stehen nur autorisierten Benutzern zur Verfügung.



Integrity / **Integrität** (germ.)

Correctness of information and functionality of systems is given.

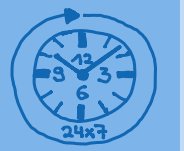
Die Korrektheit der Informationen und die Funktionsweise von Systemen ist gegeben.



Availability / **Verfügbarkeit** (germ.)

Information and systems are available to authorized users without hindrance.

Informationen und Systeme stehen autorisierten Benutzen ungehindert zur Verfügung.



Information Protection Assessment

	Confidentiality			Integrity & Availability
	Military:	Gov. (germ.):	Industry:	Recoverable within…
very high	top secret	streng geheim	(secret)	4 hours
high	secret	geheim	(strictly confidential)	1 day
normal	confidential	vertraulich	confidential	1 week
low	(restricted)	nur für den Dienstgebrauch	internal	2 week
none	(public)	(öffentlich)	public	N.A.

Privacy by Design Aspects

TRY Processing Transparency

Full transparency of all person-related data processing actions and the rationales for the processing.

DIF Differential Privacy

Introduced randomness into the results of queries on underlying confidential or person-related data to ensure ambiguity in the deduced underlying data.

CST Processing Consent

Explicit consent from the data owner for the data processing, including the right to contradict the consent at any later time.

STR Secure Storage

Secure storage of all person-related information to prevent data loss or exposure while the information is at rest.

MIN Data Minimization

Minimization of the gathered, processed and stored person-related data.

TRS Secure Transfer

Secure transfer of all person-related information to prevent data loss or exposure while the information is transferred.

PSN Data Pseudonymization

Substitute person-related information with artificial identifiers (pseudonyms) while still keeping information for backtracking to the original information.

ERA Right to Erasure

Ensure the data owners, at any time, have the right and operational means to lookup and optionally erase all their person-related information.

ANO Data Anonymization

Substitute person-related information with artificial identifiers without keeping any information for backtracking to the original information.

PRT Right to Portability

Ensure the data owners, at any time, have the right and operational means to lookup and optionally export all their person-related information.

AGG Data Aggregation

Always report person-related statistical information only as aggregated numbers which each does not cover less than about 10 involved persons.

NOT Breach Notification

Transparently provide notifications in case of any protection breach of confidential or person-related information.

Intellectual Content: Version 1.0.0 (2018-10-13), Authored 2018 by Dr. Ralf S. Engelschall
Graphical Illustration: Version 1.0.0 (2018-10-13), Copyright © 2018 Dr. Ralf S. Engelschall <http://engelschall.com> All Rights Reserved
Unauthorized Reproduction Prohibited. Licensed to Technische Universität München (TUM) for reproduction in Computer Science lecture contexts only.

ARCHITECTURE FUNDAMENTALS

Security Attacks

TECHNISCHE UNIVERSITÄT MÜNCHEN

AF 18.2

Attack Strategies

UND Undercover
Directly attack a particular victim system by infiltrating it through an undercover operation.

DRT Direct
Directly attack a particular victim system by leveraging the surprise effect.

MIM Man-In-The-Middle
Attack a particular victim system by pretending un unrelated foreign third-party system.

SCR Scarecrow
Indirectly attack a particular victim system through a scarecrow system.

TRD Third-Party
Attack a particular victim system with an unrelated foreign third-party system.

SPR Spreading
Spread out globally and attack as much victim systems as possible.

PCH Passive Couch
Passive couch and attack all victim systems coming around by coincidence.

Attack Approaches

SOC Social Engineering
Coaxing passwords and other valuable information from unsuspecting users through seemingly innocent conversation.

BRF Brute Force
Cracking passwords and other credentials by trying every word in a dictionary and even every possible combination of characters.

PHS Phishing
Acquire information, such as usernames, passwords, and credit card details, by masquerading as a trustworthy entity.

RAC Race Condition
Exploit the fact that the processing result of a target is dependent on the sequence or timing of other uncontrollable events.

SPO Spoofing
Forge or imitate an object, with the intention to induce someone to accept it as genuine.

RPL Replay
Repeat a previously wiretapped valid data transmission to cause a target to (re)accept it as coming from the original sender.

TRJ Trojan Horse
Trick someone to install a malware program without noticing, by pretending a harmless program.

BOT Bots
Use software programs which pretend to be a human user but execute actions a magnitude faster.

INJ Code Injection
Introduce foreign code into a computer program to change its execution by exploiting its problematic processing of invalid input data.

IAB Interface Abuse
Intentionally improper and not expected use of a networking or programming interface in order to trick a target into a advantageously state.

SNF Sniffing
Passive wiretapping of a foreign communication association.

HIJ Hijacking
Active wiretapping and seizing control of a previously established communication association.

FLD Flooding
Spread out a large amount of data to targets in order to exceed that data which can be accommodated by the targets.

TRP Trap
Establish a harmless looking context which allows to catch a target once it enters the context.

OVF Overflow
Send an extraordinarily large amount of input data to a target in order to exceed that data which can be accommodated by the target.

CRK Cracking
Modifying code to disable or remove features in order to circumvent copy protections, license checks, etc.

Attack Goals

PES Privilege Escalation
Gain elevated access to and perform unauthorized actions on resources that are normally protected from an application or user.

IDT Identity Spoofing
Masquerade as a foreign entity to leverage from their security credentials.

IGN Information Gain
Gain access to protected information.

ICR Information Corruption
Corrupt or even destroy protected information.

MEX Malware Execution
Execute malicious software programs.

CTL Control Takeover
Take over the control of a target system.

DOS Denial of Service
Services are overwhelmed with large number of dummy requests in order to effectively make the service unavailable for regular requests.

Attack = Strategy x Approach x Goal

Intellectual Content: Version 1.02 (2014-03-01), Authored 2012-2014 by Dr. Ralf S. Engelschall
Graphical Illustration: Version 1.2.1 (2020-06-23), Copyright © 2012-2020 Dr. Ralf S. Engelschall <http://engelschall.com>, All Rights Reserved.
Unauthorized Reproduction Prohibited. Licensed to Technische Universität München (TUM) for reproduction in Computer Science lecture contexts only.

ARCHITECTURE FUNDAMENTALS

Security Defenses

TECHNISCHE UNIVERSITÄT MÜNCHEN

AF 18.3

Defenses at Network Protocol Level

SNP Secure Network Protocol
Use a standardized, proven to be secure, network protocol to protect from Man-in-the-Middle, Undercover, Spoofing, Hijacking, and Replay.
SSL/TLS

VAL Input Validation
Use smart validation and canonicalization of all inputs to prevent problems during further processing.
JSON Schema

FTE Faked Temporary Errors
Fake temporary protocol errors to protect from Spreading and flooding Bots.
SMTP Greylisting

ESC Output Escaping
Use strict escaping of problematic characters in all outputs.
HTML Entity

TMS Timestamping
Use timestamps or virtual clocks to sequence communication and protect from Replay.
OAUTH

SIG Digital Signature
Use a cryptographically strong message digest to digitally sign an artifact in order to allow receivers to verify its origin.
RSA+SHA1

PRX Protocol Proxy
Use application-level, filtering proxies in the communication to protect from Code Injection and Interface Abuse.
PGPproxy

ENC Data Encryption
Use a cryptographically strong cipher to encrypt an artifact in order to allow privacy in communication.
AES

PID Peer Identification
Use peer identification to distinguish between different peers.
SMTP

PKI Public-Key Infrastructure
Use private/public keys and a supporting certificate authority infrastructure to leverage from asynchronous cryptography.
X.509

PAU Peer Authentification
Use cryptographically strong peer identification to protect from Man-In-The-Middle, Spoofing and Hijacking.
SSL/TLS

OOB Out-of-band Fingerprint
Use message digest based fingerprints to check the integrity of artifacts by transferring the fingerprints via out-of-band communication.
SSH FP

TFA Two-Factor Authentication
Use two different types of security credentials (which are required at the same time) to authenticate a peer.
RSA Tokens

WHT Peer Whitelisting
Use a strict and binding list of peer identifiers to restrict communication to the set of listed peers.
ACL

SID Unspecific Errors
Use less specific and common errors in responses to security-related operations to not indirectly leak sensitive information.
Login

ZNP Zero Knowledge Protocol
Use an interactive method where one peer proves to another peer that a (usually mathematical) statement is true, without revealing anything other than the veracity of the statement.
SRP

Defenses at Software Application Level

EDU User Education
Use explicit security education for end users to prevent Social Engineering based attacks to succeed.
Letters from Bank Institutes

THR Service Throttling
Use exponential delays to throttle service requests per single communication peer to protect from Denial-of-Service attacks.
Postfix smtpd

SPP Strong Password Policy
Use a strict password policy to ensure that passwords are as strong as possible to protect from dictionary lookup based Brute Force attacks.
Windows Domain Policies

FOV Service Failover
Use automatic failover of services to fallback systems to prevent Denial-of-Service attacks.
Load Balancer

SLT Salted Hashing
Use a random input (salt) to message digests which are stored side-by-side with the resulting hash value in order to prevent dictionary lookup based Brute Force attacks.
Unix crypt

ATO Atomic Operation
Use atomic operations, which either succeed or fail as a whole, to ensure consistency and protect from Race Condition attacks.
Unix 5/12b I/O

TTC Turing Test Completion
Use an Turing Test which a peer has to interactively complete in order to distinguish between a human user and a Bot attack.
CAPTCHA

SND Execution Sandboxing
Use a constrained and controlled run-time environment to restrict the execution of operations.
JVM

MAC Mandatory Access Control
Use system-level enforced constraints which restrict the ability to access or generally perform some sort of operation on an object or target.
BSD MAC

SEP Process Separation
Use separated processes to restrict privileged operations to dedicated constrained processes.
Chrome

RBA Role-Based Authorization
Use authentication based on distinct user roles to fine-granularly control access to resources and operations.
Linux RBAC

VER Data Versioning
Use versioning of data to prevent data losses.
MVCC

BET Bounded Execution Time
Use maximum run-times to bound the execution times of all major operations.
Apache MaxClient

ENV Sane Environment
Use clean and sane run-time environments to ensure maximum independence from the environment and to ensure deterministic results.
Unix "env –i"

BRU Bounded Resource Usage
Use maximum run-time resource allocations to bound the overall resource usage.
Unix ulimit

PST Prepared Statements
Use prepared statements and variable bindings for all backend database communication to protect from Code Injection attacks.
Java PreparedStatement

Intellectual Content: Version 1.02 (2012-10-17) Authored 2012 by Dr. Ralf S. Engelschall
Graphical Illustration: Version 1.04 (2020-06-23) Copyright © 2012-2020 Dr. Ralf S. Engelschall <http://engelschall.com> All Rights Reserved.
Unauthorized Reproduction Prohibited. Licensed to Technische Universität München (TUM) for reproduction in Computer Science lecture contexts only.