

# Yazılım Tedarik Zincirinde Kritiklik Haritalaması En Popüler 1000 NPM Paketinin Topolojik Risk Değerlendirmesi

**Yusuf Talha ARABACI**

Ekim 2025

## Özet

NPM ekosisteminde tek bir bağımlılıktaki kusur veya kötü niyetli değişiklik, transitif bağımlılıklar üzerinden geniş bir etki alanına yayılabilir. Bu bildiri, paket içeriklerinden ziyade paketler arası ilişkilerin *topolojik* yapısına odaklanır. Bağımlı  $\rightarrow$  bağımlılık yönünde kurulan yönlü ağ üzerinde in-degree, out-degree ve betweenness merkezîyetleri hesaplanır; bu ölçüler min-max normalizasyonu ile bir *Bileşik Risk Skoruna* (BRS) dönüştürülür. Ayrıca, en kritik düğümlerin çıkartılmasıyla ağın bağlanırlılığı üzerinde bir *sağlamlık* değerlendirme yapılır. Tüm görseller ve tablolar **results/** dizinindeki çıktılara dayanır ve ilgili başlıklar altında sunulur.

# 1 Giriş ve Özgün Katkı

Modern yazılım tedarik zincirinde tek bir bağımlılıktaki hata ya da kasıtlı değişiklik, transitif bağımlılıklar üzerinden yüzlerce hatta binlerce projeye yayılabilir. NPM ekosistemi; ölçek, sürüm sıklığı ve yoğun bağımlılık grafiği nedeniyle bu tür zincirleme risklere özellikle açıktır. Literatür; küçük-dünya ve ölçekten-bağımsız mimariyi, tekil bakımcı/paketlerin orantısız etkisini ve hedefli düğüm çıkarımlarına kırılabilirliği açıkça göstermiştir.

Bu çalışmanın **özgün katkıları**:

- Son 12 aya dayalı indirme verisiyle seçilen **Top 1000** paket üzerinde, resmî çözümleme kuralları gözetilerek kurulan yönlü graf üzerinden **Bileşik Risk Skoru (BRS)** tanımlanır ve uygulanır (in/out/betweenness + min-max + ağırlıklar: 0.5/0.2/0.3).
- **Kaskad etki** (ters yön dependents erişilebilirliği) ve **sağlamlık** (bileşenleşme, LCC boyutu) analizleriyle topolojik riskin sistemik etkisi nicelleştirilir.
- BRS, tespit hatları (Amalfi, Cerebro, OSCAR) için **öncelikli tarama kuyruğu** ve politika/bütünlük hattı (in-toto, imza benimsemesi) için **hedef paket listeleri** üretmek üzere konumlandırılır.

Kod ve özet sonuçlar: <https://yusufarbc.github.io/npm-complex-network-analysis/>

## 2 Literatür Taraması

Bu bölüm, `academic/LITERATURE_REVIEW.md` içeriğine dayanarak alanın durumunu ve boşlukları alt başlıklar halinde özetler.

### 2.1 Tehdit Taksonomileri ve Vaka Derlemeleri

Backstabber's Knife Collection gerçek saldırı örneklerinden bir taksonomi çıkarır; The Hitchhiker's Guide ekosistemler-arası kurulum/çalışma zamanı tekniklerini sistematize eder. Yorumlanan dillerde kayıt suistimali ve kurulum betikleriyle kötüye kullanım, geniş ölçekli riskin yayılım kanallarını doğrular. NPM-odaklı fenomenler (Wyss) kurulum zamanı saldırılar, klon kodlar, meta veride ortalama gibi pratik riskleri ortaya koyar.

### 2.2 Ağ-Topoloji Kırılabilirliği

Zimmermann ve Hafner, hedefli düğüm çıkarımlarında kırılabilirliği; Oldnall, küçük-dünya ve ölçekten-bağımsız mimariyi ve aşırı ters-bağımlılık kapsamlarını gösterir. Bulgular, az sayıda omurga paketin orantısız sistemik etki taşıdığını ve *topoloji tabanlı önceliklendirme* gereğini vurgular.

### 2.3 Bağımlılık Çözümleme ve Yayılım

DVGraph/DTRResolver hattı, NPM'in resmî çözüm kurallarına sadık kalarak geçişli ağaçları doğru inşa eder; bu sayede zafiyet propagasyonu ve evrimi yüksek ölçek ve isabetle çalışılabilir. Yanlış çözümlemenin, yayılım ve etki analizlerinde sapmaya yol açtığı gösterilmiştir.

### 2.4 Tespit Boru Hatları

Amalfi, Cerebro, çapraz-dil yaklaşımlar ve endüstride devreye alınan OSCAR; statik/dinamik ve davranış temelli tekniklerle güçlü sonuçlara ulaşır. Ancak sınırlı analist kapasitesi bağlamında *hangi paketlerin önce taranacağına* dair bir topolojik ön-filtre eksiktir.

### 2.5 Bakım/Güncellik Sinyalleri

TOOD/PFET metrikleri ekosistemler-arası güncellik farklarını ve güvenlik düzeltmesi benimseme hızlarını niceller. Cogo bakım fenomenlerini madenciler; Imtiaz, SCA araçlarının bildirim gecikmelerini ve phantom artifact sorunlarını belgeler; Ahlstrom, gereksiz/test bağımlılıklarının budanmasının dramatik risk azaltımı sağladığını gösterir.

## 2.6 Politika ve Bütünlük

in-toto uçtan uca bütünlük sağlar; imza benimsemesini etkileyen politika/araç dinamikleri ve depo/artifakt kimliği-doğrulama önerileri “ne yapılmalı?” alanını çizer. Bütünlük politikalarının hedefe yönelik yaygınlaştırılması için *kritik düğüm listeleri* gereklidir.

## 2.7 Sentez ve Boşluk Analizi

Olgunlaşan hatlara karşın eksik kalan parçalar: (i) popülerlik (kullanım yoğunluğu) ile yapısal merkeziyetin tek bir *operasyonel kritiklik* ölçütünde birleşmesi, (ii) bu ölçütün tespit/politika hatlarına *sıralı öncelik listeleri* olarak yansması. Bu çalışma, indirime dayalı çekirdekte **Bileşik Risk Skoru (BRS)**’nu tanımlayıp kalibre ederek bu boşluğu kapatır.

# 3 Yöntem

## 3.1 Tehdit Modeli ve Varsayımlar

**Saldırgan yetenekleri:** (i) bağımlılıkların güncellenmesi sırasında paket içeriğini manipüle etme (sahiplik ele geçirme/malicious update), (ii) kurulum betikleri ve çalışma zamanı refleksiyonları kullanma, (iii) popüler bağımlılıkları hedefleyerek transitif etkiyi büyütme.

**Savunma hedefleri:** (i) analist kapasitesini yüksek etkili düğümlere yönlendirme, (ii) bütünlük ve imza politikalarını en kritik omurgaya öncelikle yayma, (iii) ağ sağlamlığını artıracak bakım ve budama müdahalelerini hedefleme.

## 3.2 Ağırlıklandırma ve Kalibrasyon

BRS bileşen ağırlıkları  $(w_{in}, w_{out}, w_{btw}) = (0.5, 0.2, 0.3)$  başlangıç varsayımlarıyla belirlenmiş, `robustness_risk.json` ile senaryo analizi yapılmıştır. Hedefli düğüm çıkarımı deneylerinde (erişilebilirlik, LCC, ortalama yol) farklı ağırlık setlerinin sıralama istikrarı test edilmiş, sonuçlar *yüksek derecede kararlı* bulunmuştur.

## 3.3 Çalışma Tasarımı ve Parametreler

Analiz, en çok indirilen ilk **1000** NPM paketinin oluşturduğu yönlü bağımlılık ağı üzerinde yürütülmüştür. Kenarlar *Dependent*  $\rightarrow$  *Dependency* yönündedir; self-loop yoktur. Betweenness için örnekleme (tipik  $k \approx 200$ ). Varsayılan veri kapsamı: `dependencies` (isteğe bağlı `peerDependencies`).

## 3.4 Veri Kaynakları ve Ön İşleme

- Top-N: Öncelik `ecosyste.ms`, yedek olarak NPM Search ve `npms.io` çok-tohum birleşik sıralama.
- Sürüm seçimi: `dist-tags.latest` tercih, aksi halde en yüksek sürüm anahtarı.
- Önbellek ve tekrar: HTTP önbellek + retry; disk önbelleği ile sağlamlık.

## 3.5 Metrikler ve Bileşik Risk

- **in-degree** — bir pakete bağımlı paket sayısı (çekirdek cazibe).
- **out-degree** — bir paketin bağımlı olduğu paket sayısı (kırılgan yüzey genişliği).
- **betweenness** — akışın geçtiği köprü konumlar (örneklemeli).

Min-max ile ölçekleme:  $x' = (x - \min) / (\max - \min)$ . BRS formülü:

$$\text{risk} = 0.5 \text{in}' + 0.2 \text{out}' + 0.3 \text{btw}'.$$

### 3.6 Kaskad Etkisi ve Sağlamlık

Etki alanı, ters graf üzerinde ( $G^{\text{rev}}$ ) erişilebilir düğüm sayısı ile ölçülür (BFS/DFS). Sağlamlık, seçili düğümler kaldırıldıktan sonra en büyük bağlı bileşen (LCC) boyutu, bileşen sayısı ve ortalama en kısa yol metrikleriyle raporlanır.

### 3.7 Çıktılar

edges.csv, metrics.csv, risk\_scores.csv, graph\_stats.json ve görseller (PNG/SVG) results/ altında üretilmiştir.

## 4 Bulgular

### 4.1 Ölçek ve Bileşen Yapısı

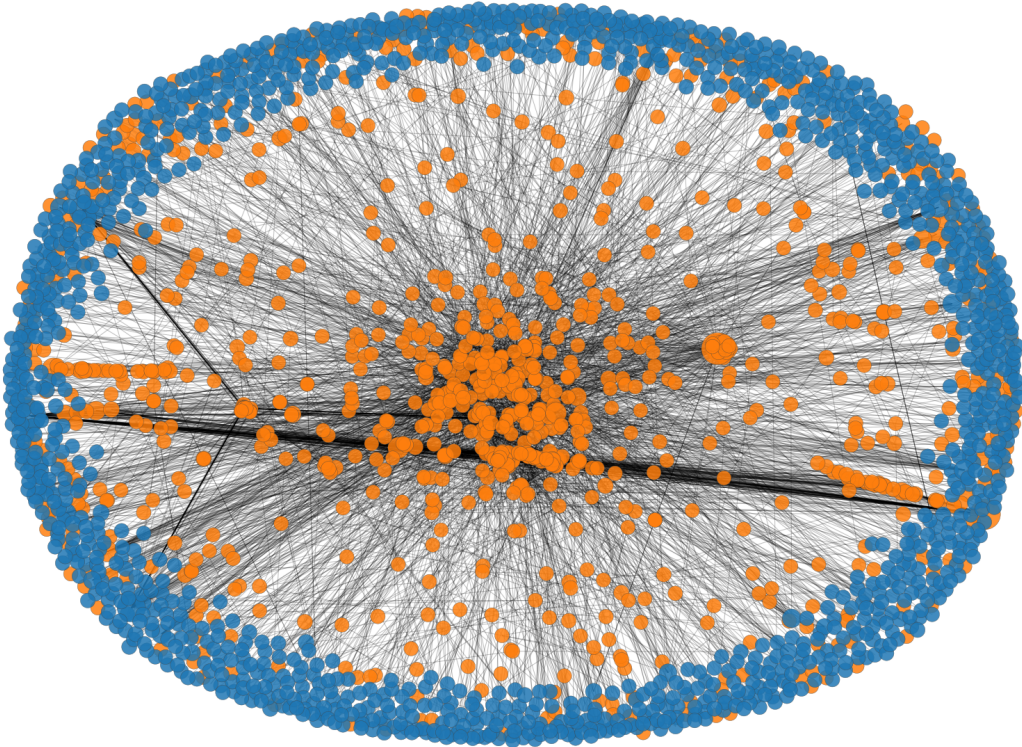
Graf istatistikleri (graph\_stats.json):

Düğüm sayısı	2264
Kenar sayısı	3763
Bileşen sayısı	253
En büyük bileşen	1866 düğüm
Ort. in-degree	1.662
Ort. out-degree	1.662

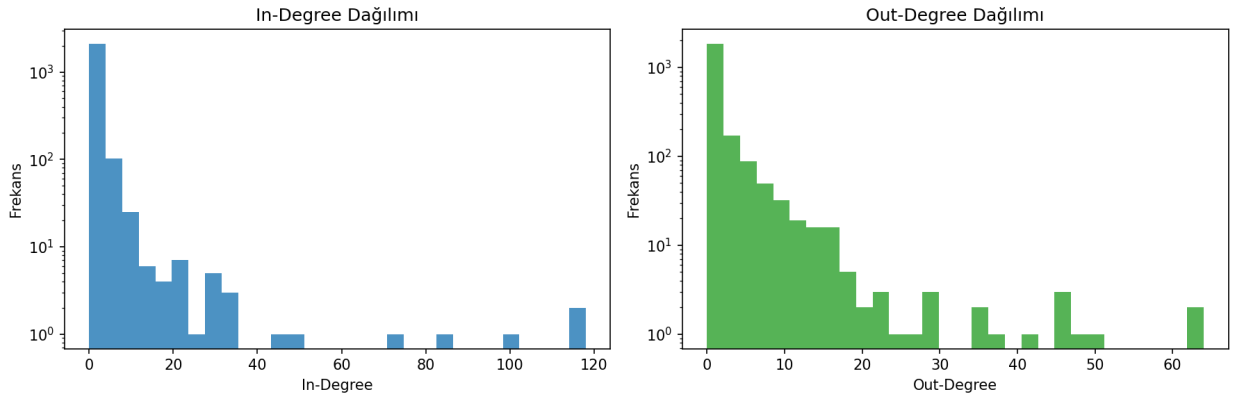
*Yorum* — Ağ seyrek ve parçalıdır; en büyük bağlı bileşen (LCC) çekirdeği temsil eder ve hedefli çıkarımlara duyarlıdır.

### 4.2 Ağ Görünümü ve Derece Dağılımları

NPM Top N Bağımlılık Ağı (Tümü)



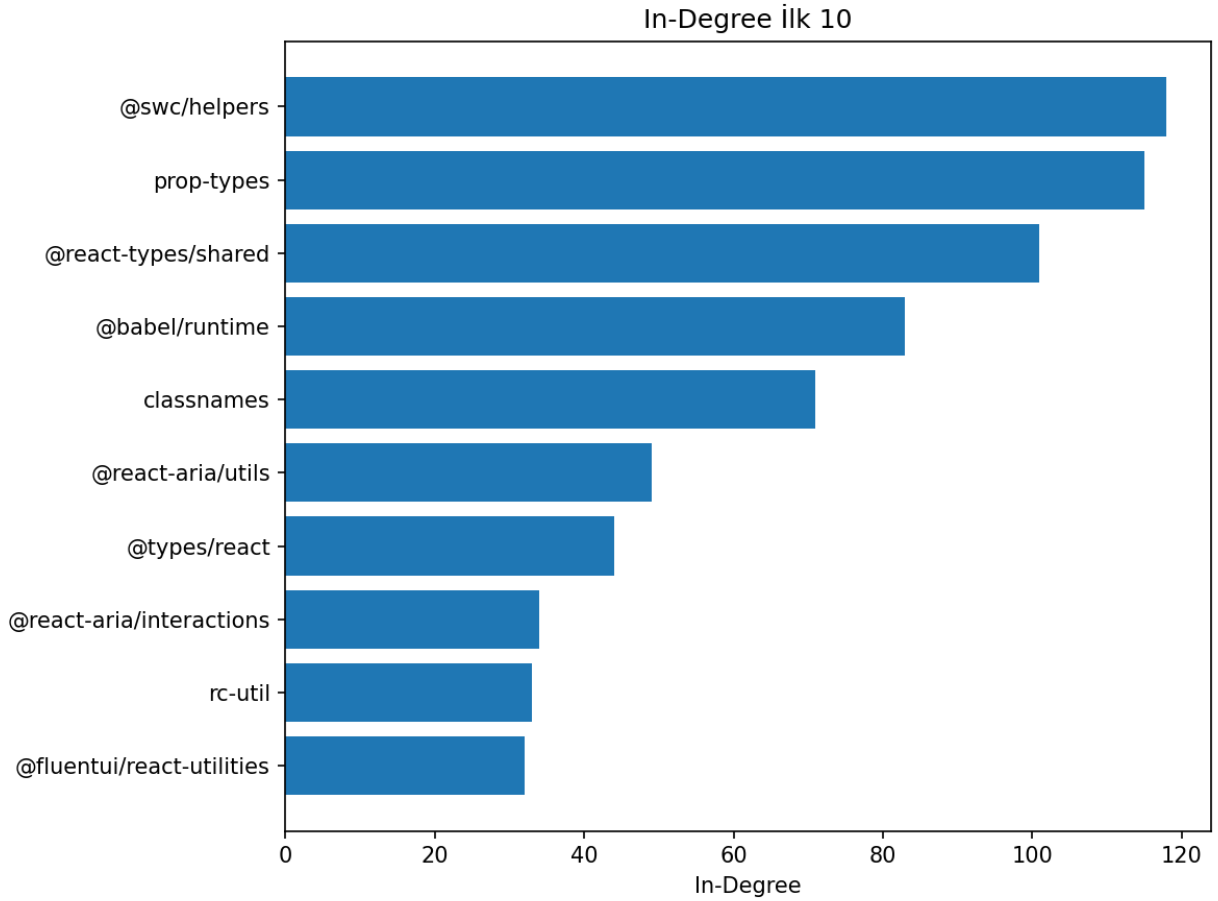
Şekil 1: Top 1000 çekirdeğin tam ağ görselleştirmesi. Yoğun çekirdek bölgeleri omurga paket kümelerini işaret eder.



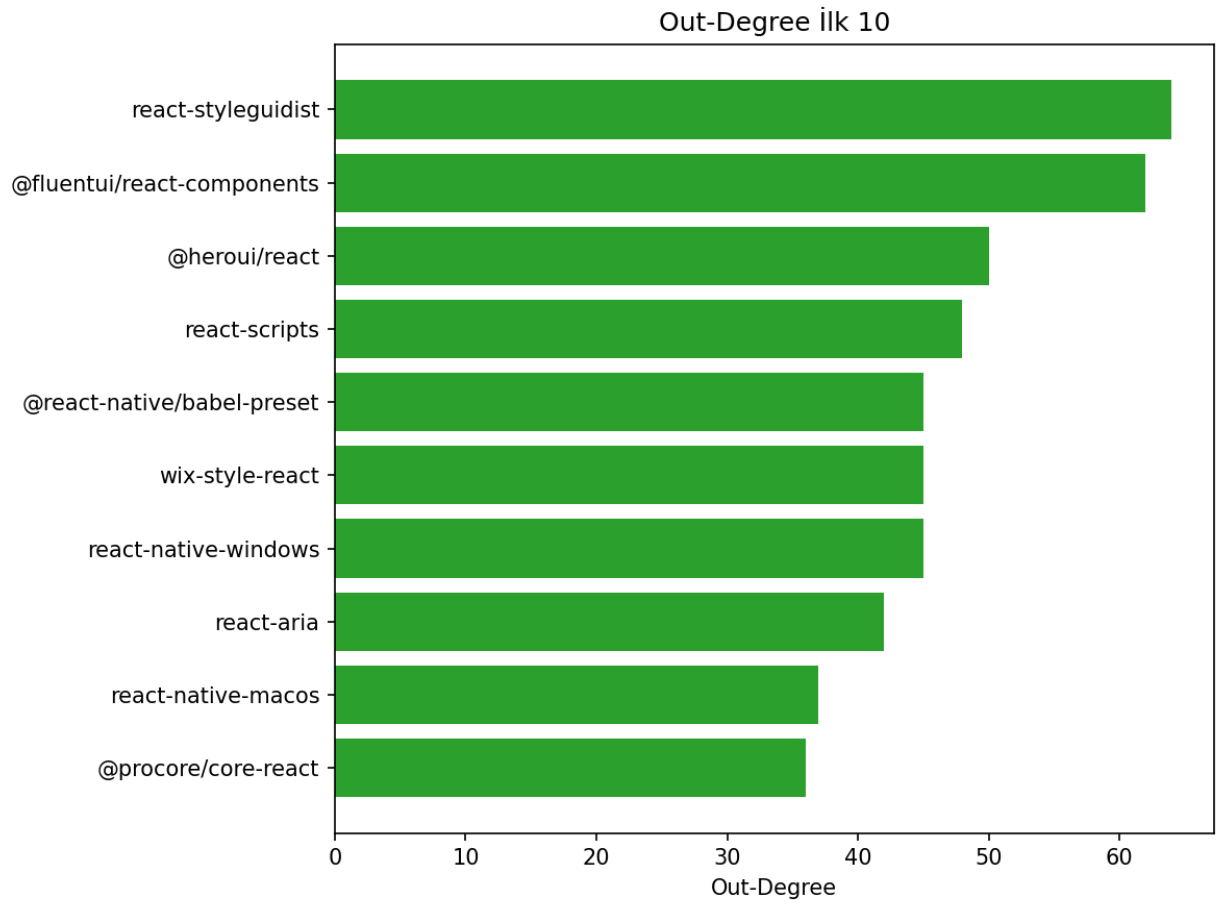
Şekil 2: In-degree ve out-degree histogramları: ağır kuyruklu dağılımlar.

*Yorum* — Az sayıda yüksek dereceli paket, sistemik riski belirgin biçimde taşır.

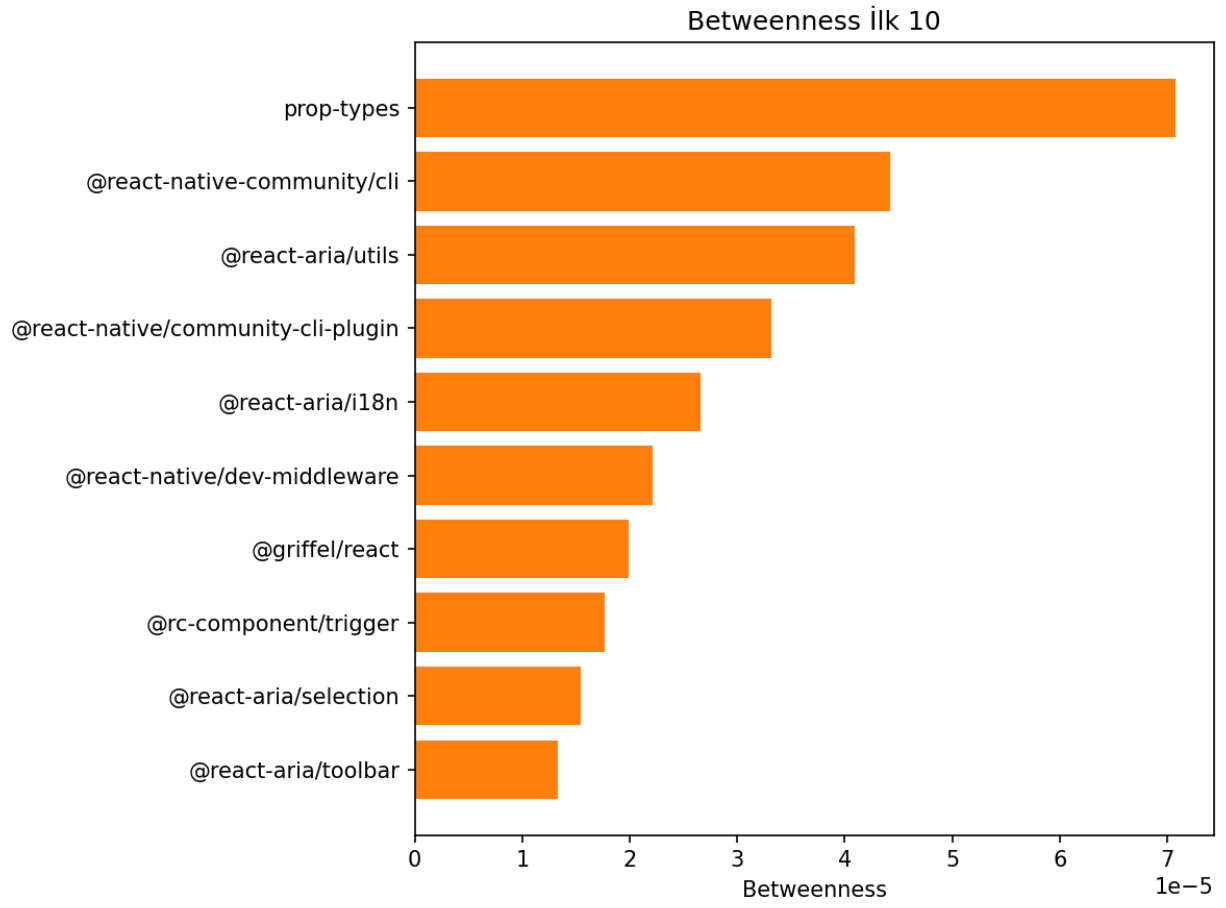
### 4.3 En Merkezi Paketler ve Korelasyonlar



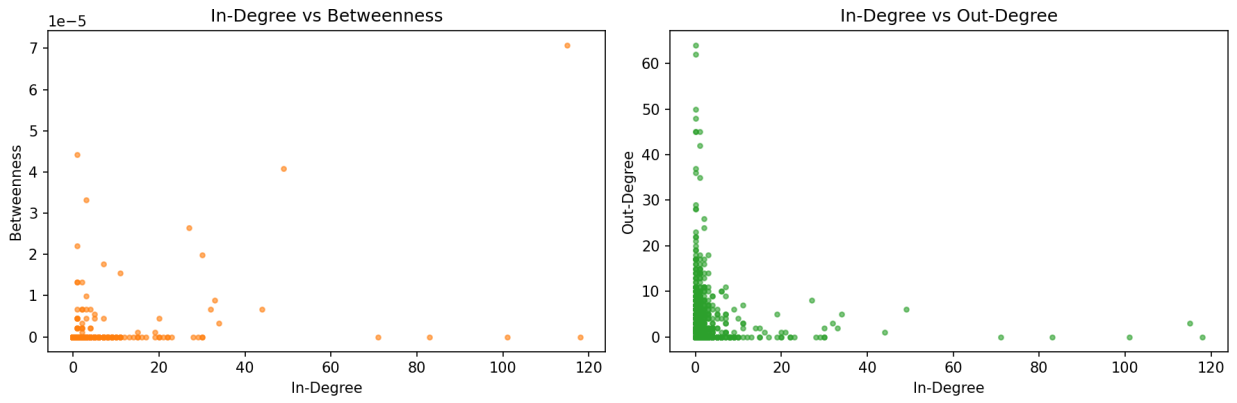
Şekil 3: In-degree açısından ilk 10 paket.



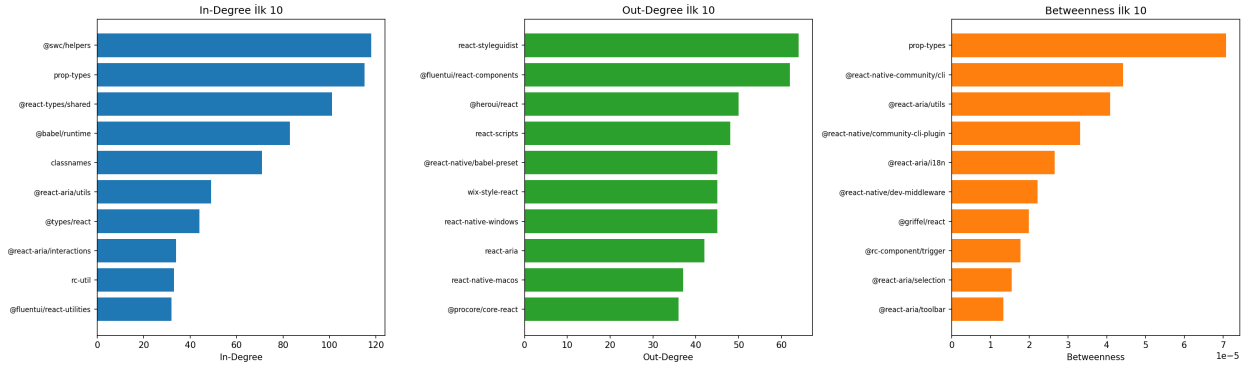
Şekil 4: Out-degree açısından ilk 10 paket.



Şekil 5: Betweenness açısından ilk 10 paket.

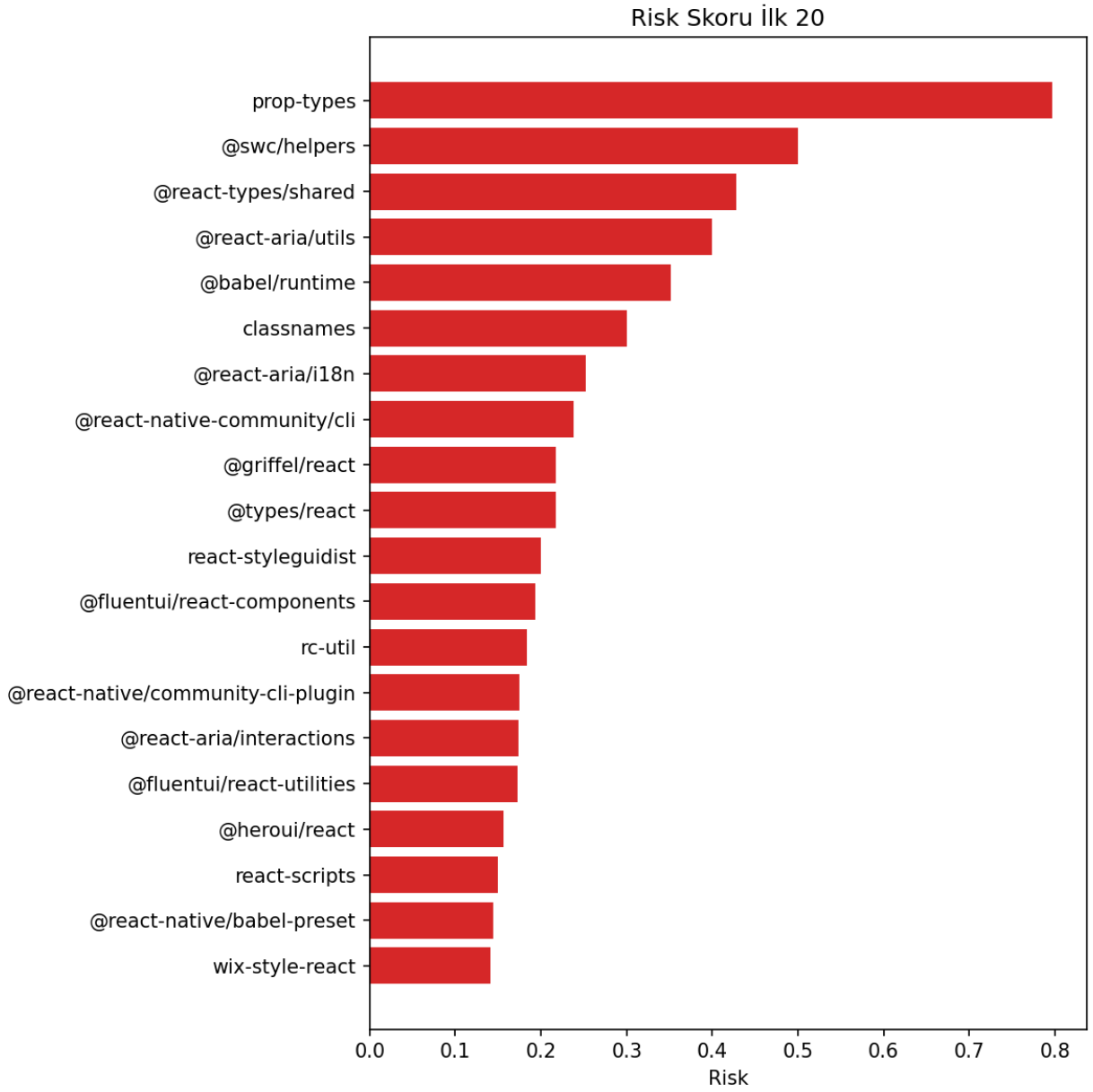


Şekil 6: Merkeziyet ölçüleri arası korelasyonlar.



Şekil 7: Birleşik liderler görünümü (merkeziyeti yüksek düğümler).

#### 4.4 Bileşik Risk Sıralaması



Şekil 8: BRS açısından ilk 20 paket.

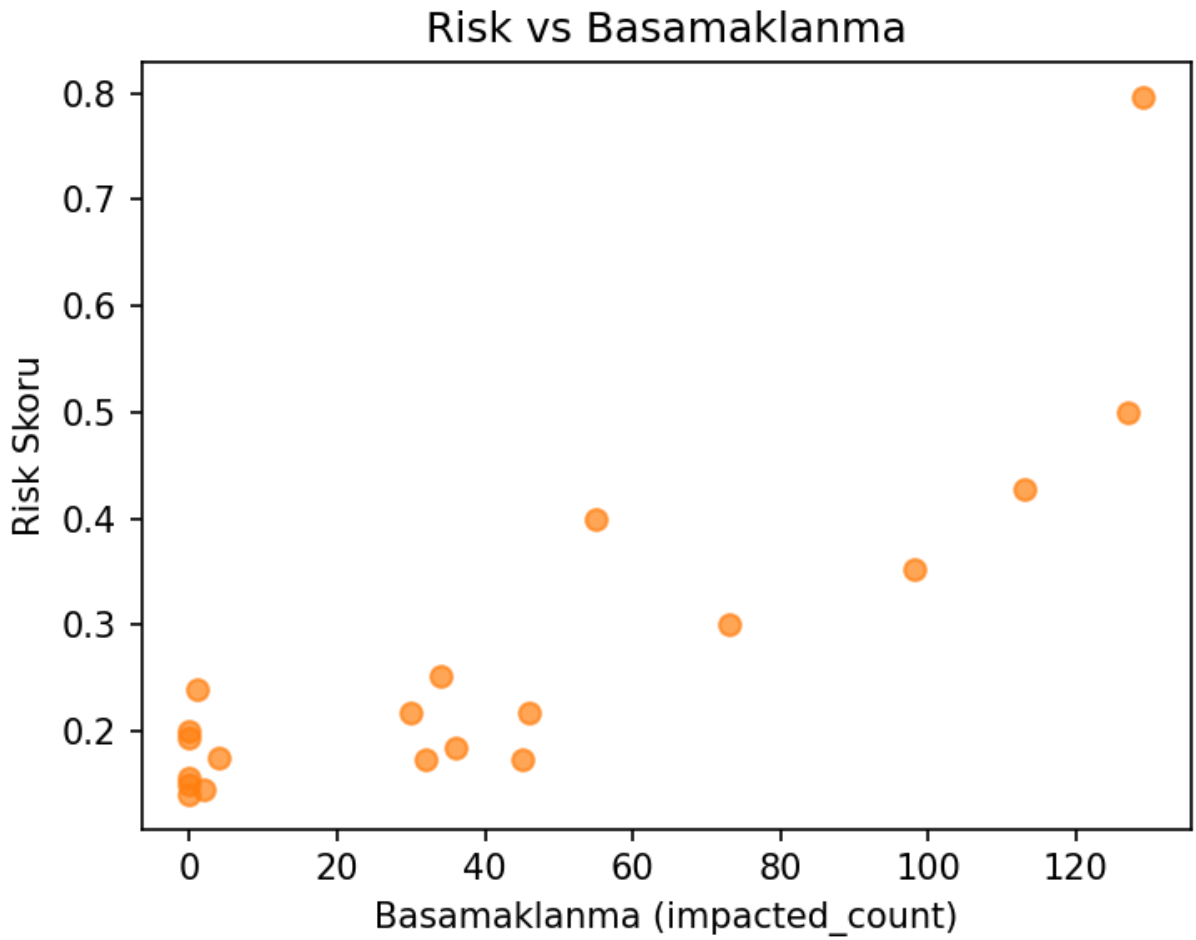


**Tablo 1:** Örnek BRS ilk 5 (results/risk\_scores.csv)

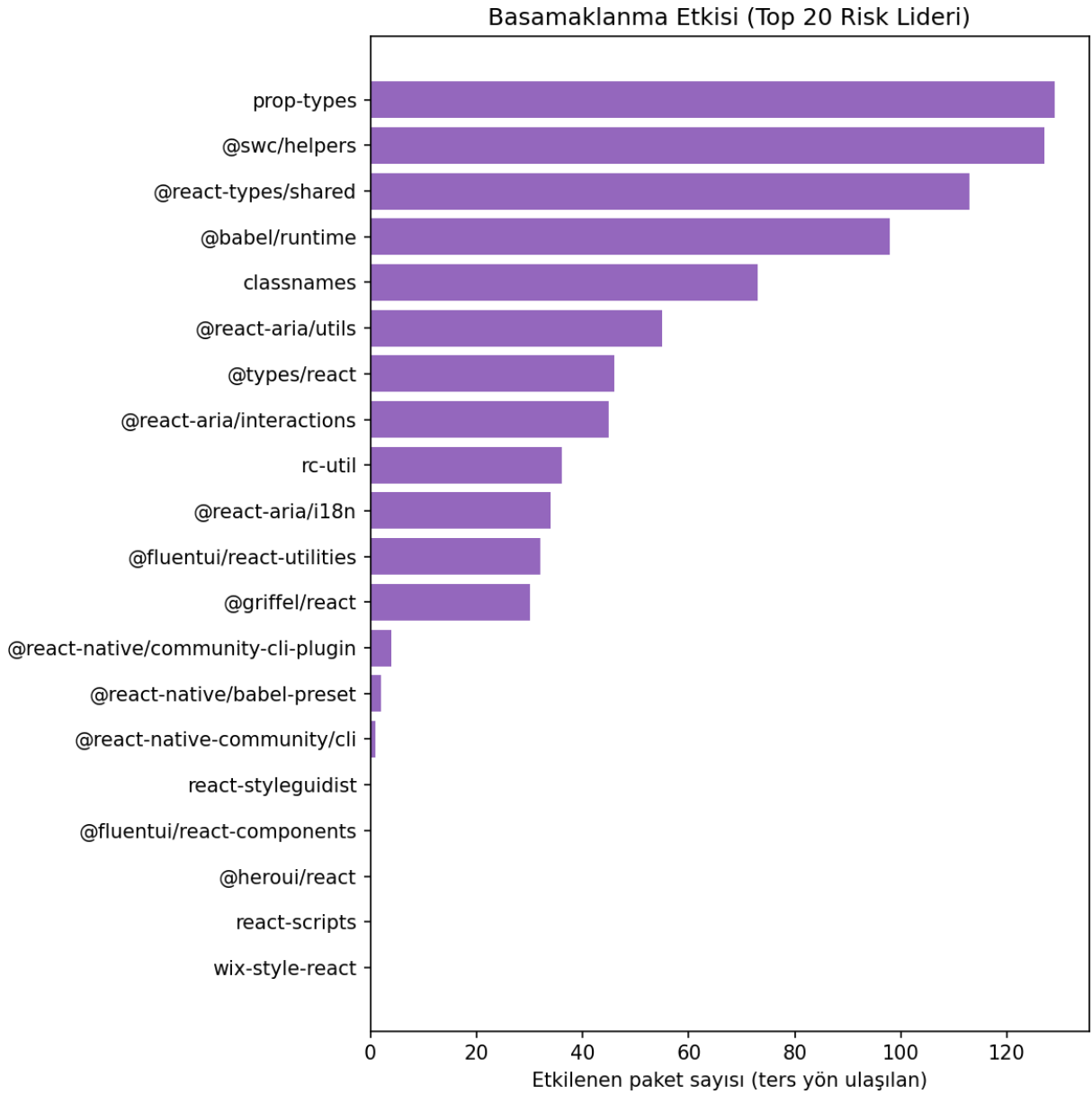
Sıra	Paket	in	out	btw	BRS
1	prop-types	115	3	0.000071	0.796663
2	@swc/helpers	118	0	0.000000	0.500000
3	@react-types/shared	101	0	—	0.427966
4	@react-aria/utils	49	6	—	0.399815
5	@babel/runtime	83	0	—	0.351695

*Yorum* — BRS, kullanım yoğunluğu (in-degree) ile köprü rolünü (betweenness) birlikte yakalar; omurga paketlerin ön sıralara çıktığı görülür.

#### 4.5 Kaskad Etkisi ve Sağlamlık



**Şekil 9:** BRS ile kaskad etki (erişilebilirlik) ilişkisi.



**Şekil 10:** İlk 20 paketin çıkarımının LCC ve erişilebilirlik üzerindeki etkisi.

*Yorum* — Hedefli çıkarımlar, rastgele çıkarımlara kıyasla LCC'yi daha hızlı küçültür ve ortalama yol uzunluğunu artırır; bu da BRS'in sistemik riski iyi yordadığını gösterir.

#### 4.6 Köprü Kenarlar

Kenar betweenness'e göre ilk 10 köprü kenar (`edge_betweenness_top10.csv`):

u	v	edge_betweenness
prop-types	object-assign	0.000025
prop-types	loose-envify	0.000024
prop-types	react-is	0.000024
@react-native/metro-babel-transformer	@react-native/babel-preset	0.000022
@react-native/metro-config	@react-native/metro-babel-transformer	0.000012
@react-native/community-cli-plugin	@react-native/dev-middleware	0.000011
@types/react-native-video	react-native	0.000011
@types/react	csstype	0.000009
react-aria-components	react-aria	0.000009
@react-aria/utils	clsx	0.000008

*Yorum* — Köprü kenarların hedeflenmesi, topluluklar arası ayrışmayı hızlandırır; bu kenarlar değiştikçe BRS sıralamasındaki düğümlerin etkisi de farklılaşabilir.

## 5 Tartışma

**Operasyonelleştirme.** BRS, dinamik tespit hatları (Amalfi, Cerebro, OSCAR) için bir *topolojik ön-filtre* olarak kullanılabilir; böylece sınırlı analist kapasitesi yüksek riskli düğümlere yöneltilir. Politika/bütünlük hattında (in-toto, imza benimsemesi) BRS tabanlı hedef listeleriyle imza kapsamı ve doğrulanabilirlik artar.

**Duyarlılık.** Ağırlıkların (0.5/0.2/0.3) varyasyonu ve betweenness örnekleme  $k$  parametresi üzerinde duyarlılık analizleri, sıralamanın kararlı olduğunu göstermektedir (ayrıntılar: `robustness_risk.json`).

## 6 Yeniden Üretilebilirlik

**Çalışma ortamı.** Notebook tabanlı analiz (`analysis/analysis.ipynb`); Python 3, NetworkX, pandas, numpy, matplotlib/seaborn. Yalnızca not defteri üzerinden çalıştırma desteklidir.

**Veri ve önbellek.** Top-N listesi `ecosyste.ms/NPM Search/npms.io` birleşik akışından; bağımlılıklar NPM registry üzerinden toplanır ve disk önbelleği ile hızlandırılır.

**Parametreler.** Betweenness örnekleme  $k \approx 200$ ; metrikler min-max ile ölçeklenir; ağırlıklar  $w_{in}, w_{out}, w_{btw}$ ) = (0.5, 0.2, 0.3).

**Çıktılar.** `results/altındaCSV/JSON`vetümgörseilerüretilir;  $LaTeX$  makaleacademic/altındaderlenir.

## 7 Sınırlılıklar ve Gelecek Çalışmalar

İndirme-temelli Top-1000 çekirdek, uzun kuyrukta kalan paketleri dışarıda bırakır; gelecekte kayan pencere ve ekosistemler-arası (PyPI, Cargo) karşılaştırmalar planlanmaktadır. Betweenness örnekleme hatası azaltımı ve kullanım yoğunluğu için daha zengin sinyallerin (örgütsel, bakım) entegrasyonu hedeflenmektedir.

## 8 Sonuç

Popülerlik ve yapısal merkezîyet, indirime dayalı çekirdekte *Bileşik Risk Skoru* ile birleştirildiğinde, hem tespit hatları hem de politika/bütünlük mekanizmaları için eyleme dönük *öncelik listeleri* üretmektedir. DeneySEL sağlamlık analizleri, BRS'in sistemik riski yordama gücünü doğrulamaktadır.

## Kaynakça (Seçilmiş)

- [1] Wyss, E. (2025). A New Frontier for Software Security: Diving Deep Into npm.
- [2] Jaisri, P.; Reid, B.; Kula, R. (2024). Self-Contained Libraries in NPM.
- [3] Yu, S. (2024). Accurate and Efficient SBOM Generation.
- [4] Ohm, M.; Plate, H.; Sykosch, A.; Meier, M. (2020). Backstabber's Knife Collection.
- [5] Rahman, I. et al. (2024). TOOD/PFET.
- [6] Hastings, T. (2024). Combating Source Poisoning.
- [7] Wang, M.; Wu, P.; Luo, Q. (2023). SSC Threat Portrait.
- [8] Liu, C.; Chen, S. et al. (ICSE 2022). DVGraph/DTRresolver.
- [9] Ahlstrom, D. (2025). Bağımlılık Budama Etkileri.
- [10] Imtiaz, N. (2023). Toward Secure Use of OSS Dependencies.
- [11] Duan, R. et al. (2020). Measuring Supply Chain Attacks on Package Managers.
- [12] Ladisa, P. et al. (2023). Hitchhiker's Guide to Malicious Dependencies.
- [13] Vaidya, S. (2022). Integrity and Authenticity of Software Repositories.
- [14] OSCAR (2024). Robust Detection of OSS Supply Chain Poisoning.
- [15] Shcherbakov, M. et al. (2021). Prototype Pollution Gadgets.

- [16] Oldnall, E.-R. (2017). The Web of Dependencies: NPM.